FireSIGHT System Database Access Guide

Version 5.4
November 11, 2016

Cisco Systems, Inc.
www.cisco.com
Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

CONTENTS

Introduction

The FireSIGHT System® database access feature allows you to query intrusion, discovery, user activity, correlation, connection, vulnerability, and application and URL statistics database tables on a Cisco Defense Center, using a third-party client that supports JDBC SSL connections.

You can use an industry-standard reporting tool such as Crystal Reports, Actuate BIRT, or JasperSoft iReport to design and submit queries. Or, you can configure your own custom application to query Cisco data under program control. For example, you can build a servlet to report intrusion and discovery event data periodically or refresh an alert dashboard.

Note that you can connect to multiple Defense Centers with a single client, but you must configure access to each one individually.

When deciding which appliance or appliances to connect to, keep in mind that querying the database on a Cisco appliance reduces available appliance resources. You should carefully design your queries and submit them at times consistent with your organization's priorities.

For more information, see the following sections:

Major Changes for Database Access in Version 5.4

If you are upgrading your FireSIGHT System deployment from Version 5.3.1 to Version 5.4, please note the following changes, some of which may require you to update your queries.

Modified Fields for Version 5.4

The `file_name` fields in `fireamp_event` and `file_event` can contain UTF-8 characters.

Modified Tables for Version 5.4

The table below lists changes to database access tables in Version 5.4.

Table 1-1 Summary of Changes to Tables in Version 5.4

Table	Description of Changes
application_host_map, page 6-5	Deprecated `application_tag_id` field
connection_log, page 7-1	Added the following fields: • `access_control_policy_uuid` • `cert_valid_start_date` • `cert_valid_end_date` • `network_analysis_policy_name` • `network_analysis_policy_UUID` • `ssl_actual_action` • `ssl_cipher_suite` • `ssl_expected_action` • `ssl_flow_flags` • `ssl_flow_messages` • `ssl_flow_status` • `ssl_issuer_common_name` • `ssl_issuer_country` • `ssl_issuer_organization` • `ssl_issuer_organization_unit` • `ssl_policy_action` • `ssl_policy_name` • `ssl_policy_reason` • `ssl_rule_action` • `ssl_rule_name` • `ssl_serial_number` • `ssl_server_name` • `ssl_subject_common_name` • `ssl_subject_country` • `ssl_subject_organization` • `ssl_subject_organization_unit` • `ssl_url_category` • `ssl_version`

Table 1-1 Summary of Changes to Tables in Version 5.4 (continued)

Table	Description of Changes
si_connection_log, page 7-16	Added the following fields: • access_control_policy_uuid • cert_valid_start_date • cert_valid_end_date • network_analysis_policy_name • network_analysis_policy_UUID • ssl_actual_action • ssl_cipher_suite • ssl_expected_action • ssl_flow_flags • ssl_flow_messages • ssl_flow_status • ssl_issuer_common_name • ssl_issuer_country • ssl_issuer_organization • ssl_issuer_organization_unit • ssl_policy_action • ssl_policy_name • ssl_policy_reason • ssl_rule_action • ssl_rule_name • ssl_serial_number • ssl_server_name • ssl_subject_common_name • ssl_subject_country • ssl_subject_organization • ssl_subject_organization_unit • ssl_url_category • ssl_version

Table 1-1 Summary of Changes to Tables in Version 5.4 (continued)

Table	Description of Changes
file_event, page 10-1	Added the following fields: • cert_valid_start_date • cert_valid_end_date • ssl_issuer_common_name • ssl_issuer_country • ssl_issuer_organization • ssl_issuer_organization_unit • ssl_serial_number • ssl_subject_common_name • ssl_subject_country • ssl_subject_organization • ssl_subject_organization_unit
fireamp_event, page 3-2	Added the following fields: • cert_valid_start_date • cert_valid_end_date • ssl_issuer_common_name • ssl_issuer_country • ssl_issuer_organization • ssl_issuer_organization_unit • ssl_serial_number • ssl_subject_common_name • ssl_subject_country • ssl_subject_organization • ssl_subject_organization_unit
intrusion_event, page 4-1	Added the following fields: • access_control_policy_UUID • network_analysis_policy_name • network_analysis_policy_UUID

Prerequisites

You must fulfill the prerequisites listed in the following sections before you can use the database access feature:

- Licensing, page 1-13
- FireSIGHT System Features and Terminology, page 1-13
- Communication Ports, page 1-13

Licensing

You can query the external database with any Cisco license installed. However, certain tables are associated with licensed features. These tables are only populated with data if the appropriate Cisco license is installed and your deployment is properly configured to generate the data. If you query these tables and the associated Cisco license is not installed, you retrieve no results. For more information about licensing, see Understanding Licensing in the *FireSIGHT System User Guide*.

FireSIGHT System Features and Terminology

To understand the information in this guide, you should be familiar with the features and nomenclature of the FireSIGHT System, and the function of its components. You should be familiar with the different types of event data these components generate. Note that you can frequently obtain definitions of unfamiliar or product-specific terms in the *FireSIGHT System User Guide*. The user guide also contains additional information about the data in the fields documented in this guide.

Communication Ports

The FireSIGHT System requires the use of specific ports to communicate internally and externally, between appliances, and to enable certain functionality within the network deployment.

After you enable database access on the Defense Center, the system uses ports 1500 and 2000 for the connection that carries JDBC traffic between the client and the appliance.

Client System

On the computer that you want to use to connect to the Cisco database, you must install Java software, also known as the Java Runtime Environment (JRE) or the Java Virtual Machine (JVM). You can download the latest version of Java from `http://java.com/`.

You must download and unzip a package from the Defense Center that contains the JDBC driver files you will use to connect to the database. The package also contains executable files used to install an SSL certificate for encrypted communication with the Defense Center, and other source files for these utilities.

You should also understand how to change applicable system settings on your computer, such as environment variables.

Query Application

To query the Cisco database, you can use commercially available reporting tools such as Actuate BIRT, JasperSoft iReport, or Crystal Reports, or any other application (including custom applications) that supports JDBC SSL connections. This guide provides the information you need to connect to the

database, including the JDBC URL, driver JAR files, driver class, and so on. However, you should refer to your reporting tool documentation for detailed instructions on how to configure a JDBC SSL connection.

Cisco also provides a sample command-line Java application named RunQuery, which you can use to test your database connection, view the schema, and run basic ad hoc queries manually. The RunQuery source code is also a reference for setting up the database connection in a custom Java application. The RunQuery source code is included in the ZIP package that you download from the Defense Center.

RunQuery is a sample client only, **not** a fully featured reporting tool. Cisco **strongly** recommends against using it as your primary method of querying the database. For information on using RunQuery, refer to the README file included in the ZIP package.

Note that the database access feature uses only the following JDBC functionalities:

- database metadata, which includes information such as schema, version, and supported features
- SQL query execution

Database access does not use any other JDBC functionality, including stored procedures, transactions, batch commands, multiple result sets, or insert/update/delete functions.

Database Queries

To query the database, you should know how to construct and execute `SELECT` statements on single tables and on multiple tables using join conditions.

To assist you, this guide contains information on supported MySQL query syntax, the Cisco database schema, allowed joins, and other important query-related requirements and limitations.

Where Do I Begin?

After you have met the prerequisites described in Prerequisites, page 1-12, you can begin configuring your client system to connect to a Defense Center.

Setting Up Database Access, page 2-1 explains how to configure the appliance to allow access, how to configure your client system to connect to the appliance, and how to configure your reporting application to connect to the appliance. It also contains some basic query instructions and information on supported MySQL syntax.

The rest of the guide contains schema and join information for the database and sample queries, and is split into the following chapters:

- Schema: System-Level Tables, page 3-1 contains schema and join information for system-level tables such as the audit log and health events.
- Schema: Intrusion Tables, page 4-1 contains schema and join information for intrusion-related tables.
- Schema: Statistics Tracking Tables, page 5-1 contains schema and join information for application, URL, and user statistics tables.
- Schema: Discovery Event and Network Map Tables, page 6-1 contains schema and join information for tables that contain discovery event and network map information, that is, information on your network assets.
- Schema: Connection Log Tables, page 7-1 contains schema and join information for tables that contain connection event and connection summary event information.

- Schema: User Activity Tables, page 8-1 contains schema and join information for tables that contain user discovery and identity data.
- Schema: Correlation Tables, page 9-1 contains schema and join information for correlation-related tables, including white list events and violations and remediation status data.
- Schema: File Event Tables, page 10-1 contains schema and join information for the table that contains file events.

Setting Up Database Access

To obtain read-only access to the database, you must first configure the appliance to allow access. Then, you must configure your client system to connect to the appliance by downloading the JDBC driver and accepting the SSL certificate from the appliance you want to access. Finally, you must configure your reporting application to connect to the appliance.

Note Before you set up database access, you should make sure you have fulfilled the prerequisites described in Prerequisites, page 1-12.

For more information, see the following sections:

Deciding Which Appliance to Access

You can connect to multiple appliances with a single client, but you must configure each appliance individually to allow access.

To decide which appliance or appliances to connect to, keep in mind that the available data on the appliance depends on the licenses you have installed, and on your configuration of the FireSIGHT System.

The following list outlines some of the specific reasons why a query may not return results:

- Your query is too specific. For example, you may need to adjust the time range or IP address range of a query.
- Not all of the fields for an event may be populated, depending on the network traffic that caused an event to be generated. For example, not all connection events contain payload information.

- The appliance does not have the appropriate license. For example, network discovery and user identity-related events are not logged to the database unless you have installed the appropriate feature licenses.
- You have not configured the FireSIGHT System to log the event type you are querying. For example:
  - Logging intrusion events, discovery-related events, and health events requires that you apply the appropriate policy.
  - Logging network discovery events and host input events is configurable in the system policy. Note that logging is enabled by default.
  - Logging user identity data requires that you configure network discovery.
  - Transmitting packet data for intrusion events requires you to enable that option when adding a managed device to the Defense Center.
  - Generating and logging correlation events, compliance white list events, and remediation status events requires that you add rules or responses to an active correlation policy.
  - To log connection events, you must enable logging of connections in your access control rules and for the default action in your access control policy. Your managed devices are not receiving network traffic that causes events to be generated.
  - Database limits are set to zero in the system policy on the appliance you are querying.
  - Your managed devices are not receiving network traffic that causes events to be generated.

For more information on how events are generated and logged, see the *FireSIGHT System User Guide*.

Creating a Database User Account

License: Any

To configure access to the FireSIGHT System database, you must first create a user account and assign it the External Database User permission. You can grant this permission by assigning the account either a Cisco-predefined user role that includes the External Database User permission or a custom user role created by your organization that includes the External Database User permission. See the *FireSIGHT System User Guide* for information on creating the user account and viewing the permissions in a given user role.

Tip Users assigned the predefined Administrator role have the External Database User permission by default.

External Database users who are locally created and authenticated can change their passwords in the Defense Center web interface. See the *FireSIGHT System User Guide* for information on changing passwords. The following table describes some of the options for locally created users to regulate passwords and account access.

Table 2-1 User Account Password Options

Option	Description
Use External Authentication Method	Select this option if you want this user's credentials to be externally authenticated. **Note** If you select this option for the user and the external authentication server is unavailable, that user can log into the web interface but cannot access any functionality.
Maximum Number of Failed Logins	Enter an integer, without spaces, that determines the maximum number of times each user can consecutively fail login attempts before the account is locked. The default setting is five tries; use `0` to allow an unlimited number of failed logins.
Minimum Password Length	Enter an integer, without spaces, that determines the minimum required length, in characters, of a user's password. The default setting is `8`. A value of `0` indicates that no minimum length is required.
Days Until Password Expiration	Enter the number of days after which the user's password expires. The default setting is `0`, which indicates that the password never expires.
Days Before Expiration Warning	Enter the number of warning days users have to change their password before their password actually expires. The default setting is `0` days. ⚠ **Caution** The number of warning days must be **less than** the number of days before the password expires.
Force Password Reset on Login	Select this option to force users to change their passwords the first time they log in.
Check Password Strength	Select this option to require strong passwords. A strong password must be at least eight alphanumeric characters of mixed case and must include at least one numeric character and one special character. It cannot be a word that appears in a dictionary or include consecutive repeating characters.
Exempt from Browser Session Timeout	Select this option if you do not want a user's login sessions to terminate due to inactivity. Users with the Administrator role cannot be made exempt.

Note that you can externally create and authenticate External Database users, in which case the appliance retrieves user credentials from an external repository, such as an LDAP directory server or RADIUS authentication server. You manage password settings for these users on the external server. For detailed information on external authentication, see the *FireSIGHT System User Guide*.

Enabling Database Access on the Defense Center

License: Any

After you create an External Database user, you must configure the Defense Center to allow access to the database on the appliance. You must also configure a database access list on the appliance and add all host IP addresses that will query the external database.

To enable database access:

Access: Admin

Step 1 On the Defense Center, select **System > Local > Configuration**.

Step 2 Click **Database**.

The Database Settings menu appears.

Step 3 Select the **Allow External Database Access** check box.

The **Access List** field appears.

Step 4 Type the fully qualified domain name (FQDN), or IPv4 address, of the Defense Center in the **Server Hostname** field, depending on your third-party application requirements. You cannot use an IPv6 address as you cannot use an IPv6 address to install a certificate.

If you type an FQDN, you must make sure that the client can resolve the FQDN of the Defense Center. If you type an IP address, you must make sure that the client can connect to the Defense Center using the IP address.

Step 5 To add database access for one or more IP addresses, click **Add Hosts**.

An **IP Address** field appears in the **Access List** field.

Step 6 In the **IP Address** field, you have the following options, depending on the IP addresses you want to add:

- an exact IPv4 address (for example, `192.168.1.101`)
- an exact IPv6 address (for example, `2001:DB8::4`)
- an IP address range using CIDR notation (for example, `192.168.1.1/24`)
- For information on using CIDR in the FireSIGHT System, see IP Address Conventions in the *FireSIGHT System User Guide*.
- **any**, to designate any IP address

Step 7 Click **Add**.

The IP address is added to the database access list.

Step 8 Optionally, to remove an entry in the database access list, click the delete icon ( ).

Step 9 Click **Save**.

The database access settings are saved.

Step 10 Continue with the procedure in the next section, Downloading the JDBC Driver.

Downloading the JDBC Driver

License: Any

After you create an External Database user and configure the Defense Center to allow database access, download the JDBC driver to the client system. You must use this JDBC driver to connect to the database.

To download the JDBC driver:

Access: Admin

Step 1 On the Defense Center, select **System > Local > Configuration**.

Step 2 Click **Database**.

The Database Settings menu appears.

Step 3 Next to **Client JDBC Driver**, click **Download** and follow your browser's prompts to download the **client.zip** package.

Step 4 Unpack the ZIP package. Note the location.
Make sure you preserve the file structure of the package.
The driver, along with other files, is packaged in a ZIP file (**client.zip**). The package contains the following directories:

- **bin**, which contains a sample client called RunQuery, as well as the executable files you use to install the certificate for encrypted communication between your client and the Defense Center
- **lib**, which contains JDBC driver JAR files
- **src**, which contains source code for the executable files in the **bin** directory

Step 5 Continue with the procedure in the next section, Installing the Client SSL Certificate.

Installing the Client SSL Certificate

Once you have downloaded the JDBC driver, use the Cisco-provided program named InstallCert to accept and install the SSL certificate from the Defense Center. Your client system and the Defense Center communicate securely with SSL certificate authentication. When you accept the certificate, your computer adds it to the keystore (**jssecacerts**) in the **security** directory of the currently running JRE:

```
$JAVA_HOME/jre[version]/lib/security
```

The following represent common locations of the keystore for computers running Microsoft Windows and UNIX, respectively:

- `C:\Program Files\Java\jre[version]\lib\security\jssecacerts`
- `/var/jre[version]/lib/security/jssecacerts`

Note If the Java query application you plan to use to access the database access function uses a different JRE, you must copy the keystore to the **security** directory of the other JRE.

To install the SSL certificate using InstallCert:

Step 1 On your computer, open a command line interface.

Step 2 At the command prompt, change to the **bin** directory created when you unpacked the ZIP package.

Step 3 To install the Defense Center's SSL certificate, type the following and press Enter:

```
java InstallCert defense_center
```

where *defense_center* is either the FQDN or the IP address of the Defense Center. InstallCert does not support IPv6 addresses. If you are on an IPv6 network, you must use a resolvable hostname.

Output similar to the following example from a computer running Microsoft Windows appears:

```
Loading KeyStore C:\Program Files\Java\jre6\lib\security...
Opening connection to defensecenter.example.com:2000...
Starting SSL handshake...
```

```
Subject GENERATION=server, T=vjdbc, O="Cisco, Inc.",
...
...
```

You are prompted to view the certificate.

Step 4 Optionally, view the certificate.

You are prompted to accept the certificate.

Step 5 Accept the certificate.

The certificate is accepted, and output similar to the following example from a computer running Microsoft Windows appears:

```
Added certificate to keystore 'C:\Program Files\Java\jre6\lib\security\jssecacerts'
using alias 'defensecenter.example.com-1'
```

If you plan to use Crystal Reports, note the location of the keystore (`jssecacerts`). You will need this information later.

Step 6 You have the following options:

- If you have a third-party application, continue with the procedure in the next section, Connecting to the Database Using a Third-Party Application, page 2-6.
- If you have a custom application, continue with the procedure in Connecting to the Database Using a Custom Program, page 2-8.

Connecting to the Database Using a Third-Party Application

After you install the certificate, you can query the database on a Defense Center using any third-party client that supports JDBC SSL connections. The following table lists information you may need to configure a connection between your client and the Defense Center.

Table 2-2 Connection Information for Database Access Clients

Information	Description
JDBC URL	The following JDBC URL identifies the Cisco database so the JDBC driver on your client can establish a connection with it: `jdbc:vjdbc:rmi://defense_center:2000/VJdbc,eqe` where *`defense_center`* is either the FQDN or the IP address for the Defense Center.
JDBC driver JAR files	You must use the following JAR files when you configure a connection to the Cisco database: • `vjdbc.jar` • `commons-logging-1.1.jar` These files are located in the **`lib`** subdirectory where you unpacked the **`client.zip`** file you downloaded and unpacked, as described in Downloading the JDBC Driver, page 2-4.

Table 2-2 Connection Information for Database Access Clients (continued)

Information	Description
JDBC driver class	You must use the following driver class when you configure a connection to the Cisco database: `com.sourcefire.vjdbc.VirtualDriver`
user name and password	To connect to the database on an appliance, use a user account that has the External Database User permission. For more information, see Creating a Database User Account, page 2-2.

The following sections contain tips for connecting to the Cisco database using three popular industry-standard reporting tools. Whether you use one of these tools or another Java-based application, you should refer to the documentation for your reporting tool for detailed instructions on how to create a JDBC SSL connection.

Crystal Reports

The following is valid for installing Crystal Reports 2011 on a 32-bit Windows environment. If you run a 64-bit Windows environment, the filepaths may be different.

To allow Crystal Reports 2011 to connect to the Cisco database, you must:

- Add the JDBC driver JAR files that you downloaded from the Defense Center to the Crystal Reports classpath. Assuming a default installation of Crystal Reports, you can edit the classpath section in the following file:
  `C:\Program Files\SAP BusinessObjects\SAP Business Objects Enterprise XI 4.0\Java\CRConfig.xml`
- Copy the keystore that was created when you installed the client SSL certificate to the appropriate Crystal Reports security directory. Assuming a default installation of Crystal Reports, that directory is:
  `C:\Program Files\SAP BusinessObjects\SAP Business Objects Enterprise XI 4.0\win32_x86\jdk\jre\lib\security`
- Create a new JDBC (JNDI) connection with the Database Expert, using `Cisco` as the database name.

JasperSoft iReport

To allow iReport to connect to the Cisco database, you must:

- Add the JDBC driver JAR files that you downloaded from the Defense Center to the iReport classpath.
- Add a new JDBC driver using the JDBC driver JAR files that you downloaded from the Defense Center. After you add the driver files, iReport should find the correct driver class.
- Create a new database connection using the driver you just created.

Actuate BIRT

To allow BIRT to connect to the Cisco database, you must:

- Add a driver definition using the **Generic JDBC Driver** template.
- Create a new database connection using the **Generic JDBC** profile type.
- Create a data source for reports using the **JDBC Data Source** data source type.

Tip If you cannot select the Cisco driver class when creating a new JDBC data source profile, add the driver using the JDBC driver JAR files you downloaded from the Defense Center.

Connecting to the Database Using a Custom Program

Once you install the certificate, you can enable custom Java report tools to query the Cisco database. Cisco provides a sample Java command line application named RunQuery that establishes the required SSL connection using the JDBC driver provided with your Defense Center. RunQuery retrieves both table records and table metadata. The source code is included in the **src** directory of the ZIP package you downloaded from the Defense Center. See Downloading the JDBC Driver, page 2-4.

Note RunQuery is a sample client only, **not** a fully featured reporting tool. Cisco **strongly** recommends against using it as your primary method of querying the database. For information on using RunQuery, refer to the README file included in the ZIP package.

See the following for more information on connecting to the database using a custom program:

- Sample Code for Custom Java Programs, page 2-8 describes the Java classes and methods that the RunQuery application uses to set up the database connection and submit queries.
- Running the Application, page 2-9 discusses environment requirements necessary for your Java application to execute.

Sample Code for Custom Java Programs

The RunQuery source code uses the functions discussed below. These code samples illustrate one of several possible implementation approaches.

Dynamically setting the SSL provider connection

After you install the SSL security certificate on your client (see Installing the Client SSL Certificate, page 2-5), you can dynamically register your JSSE provider with the following line in your program:

```
Security.addProvider(new com.sun.net.ssl.internal.ssl.
Provider());
```

Initializing the JDBC driver for your program

You can load the JDBC driver class in your Java application using the `Class.forName()` method, as follows:

```
Class.forName("com.sourcefire.vjdbc.VirtualDriver").newInstance();
```

If your program launches from the command line, the user supplies the JDBC class as follows:

```
java -Djdbc.drivers="com.sourcefire.vjdbc.VirtualDriver" program_name ...
```

where *program_name* is the name of your program.

Connecting the program to the database

Your program must obtain a JDBC connection object before it can submit queries. Use the `DriverManager.getConnection` method as follows to establish the connection and get the connection object:

```
Connection conn = DriverManager.getConnection("jdbc:vjdbc:rmi://my_dc:2000/VJdbc,eqe",
"user", "password");
```

where *my_dc* is either the FQDN or the IP address for the Defense Center, `user` is the database access user account name, and *password* is the account password.

Querying the data in the Cisco tables

Create an SQL query object to submit the query and assign the retrieved records to a result set, as follows:

```
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("sql");
```

where *sql* is the SQL query. See Querying the Database, page 2-10 for supported SQL functions.

Producing the results of a table query

With the result set (`rs`) generated by the above query, you can output the fields as follows:

```
while(rs.next())
{
    for(int i=1; i<= md.getColumnCount(); i++)
    {
        System.out.print(rs.getString(i) + " ");
    }
    System.out.print("\n");
}
```

Getting schema information

Your program can list the tables in the database, as follows:

```
DatabaseMetaData metaData = conn.getMetaData();
      ResultSet tables = meta.getTables(null, null, null, null);
      while (tables.next())
      {
System.out.println(tables.getString("TABLE_NAME"));
      }
```

Your program can list a table's columns, as follows:

```
ResultSet columns = metaData.getColumns(null, null, "table_name", null);
```

where *table_name* is the name of the database table.

Running the Application

Before you run your application, you must set the `CLASSPATH` on the client computer to include the current directory and the locations of your application's JAR files.

If you downloaded and unpacked the ZIP package for Database Access as noted in Downloading the JDBC Driver, page 2-4, update the `CLASSPATH` as follows:

To run the application in a Unix environment:

Step 1 Use the following command:

```
export CLASSPATH=$CLASSPATH:.:path/lib/vjdbc.jar:path/lib/commons-logging-1.1.jar
```

where *path* is the directory path where you unpacked the ZIP package downloaded from the Defense Center.

To run the application in a Windows 7 environment:

Step 1 Right-click the Computer icon and select **Properties**.

The **System** window appears.

Step 2 Click **Advanced System Settings**.

The **System Properties** window appears.

Step 3 Select the **Advanced** tab.

Step 4 Click **Environment Variables...**.

The **Environment Variables** window appears.

Step 5 Select the **CLASSPATH** system variable and click **Edit...**.

The **Edit System Variable** window appears.

Step 6 Add the following to the **Variable value:** field:

```
.;path\bin;.;path\lib\vjdbc.jar;.;path\lib\commons-logging-1.1.jar;.;path\lib
```

where *path* is the directory path where you unpacked the ZIP package downloaded from the Defense Center.

Step 7 Click **OK** to save the value.

The Environment Variables window appears.

Step 8 Click **OK** to save the value. You can now run the application.

Querying the Database

The following sections contain information on supported query syntax and other important query-related requirements and limitations:

- Supported SHOW Statement Syntax, page 2-11 describes the supported MySQL **SHOW** statement syntax for querying the Cisco database.
- Supported DESCRIBE or DESC Statement Syntax, page 2-11 describes the supported MySQL **DESCRIBE** statement syntax for querying the Cisco database.
- Supported SELECT Statement Syntax, page 2-12 describes the supported MySQL **SELECT** statement syntax for querying the Cisco database.
- Join Constraints, page 2-13 describes the joins supported for querying the Cisco database and explains how to get information on the specific allowed joins for any table.

- Querying Data Stored in Unfamiliar Formats, page 2-14 describes how to perform queries on data stored in formats that may be unfamiliar (including UNIX timestamps and IP addresses) so your queries are successful and your results appear as expected.
- Limiting Queries for Performance Reasons, page 2-15 contains recommendations on constraining your queries so as not to degrade the performance of the FireSIGHT System.
- Query Tips, page 2-16 contains tips for querying intrusion events across several appliances.

For schema information and allowed joins, see the following chapters:

- Schema: System-Level Tables, page 3-1
- Schema: Intrusion Tables, page 4-1
- Schema: Statistics Tracking Tables, page 5-1
- Schema: Discovery Event and Network Map Tables, page 6-1
- Schema: Connection Log Tables, page 7-1
- Schema: User Activity Tables, page 8-1
- Schema: Correlation Tables, page 9-1

Supported SHOW Statement Syntax

The **SHOW** statement lists all tables in the Cisco database. The following represents the supported MySQL **SHOW** statement syntax you can use when querying the Cisco database:

```
SHOW TABLES;
```

Any **SHOW** statement syntax not listed above is not supported.

Supported DESCRIBE or DESC Statement Syntax

The Cisco database offers limited use of the **DESCRIBE** statement. In the Cisco database, the output of the DESCRIBE statement only lists the names of the columns and the type of data in each column. The following represents the supported MySQL **DESCRIBE** statement syntax you can use when querying the Cisco database:

```
DESCRIBE table_name;
```

The Cisco database also supports the identical command DESC:

```
DESC table_name;
```

Table 2-3 Supported DESCRIBE Statement Syntax

Where...	Is...
table_name	the name of a table you are querying

Any **DESCRIBE** statement syntax not listed above is not supported. In particular, the FireSIGHT System database access feature does not support:

- the INDEX FOR clause
- the TABLE clause

- the PROCEDURE clause

Supported SELECT Statement Syntax

The following represents the supported MySQL SELECT statement syntax you can use when querying the Cisco database:

```
SELECT
[ALL | DISTINCT]
select_expr [, select_expr ...]
FROM table_references
[WHERE where_condition]
[GROUP BY { column_name | position } [ ASC | DESC ], …]
[HAVING where_condition]
[ORDER BY { column_name | position } [ ASC | DESC ], …]
[LIMIT { [offset,] row_count | row_count OFFSET offset}]
```

The following table details the required syntax for the clauses and arguments in the above SELECT statement.

Table 2-4 Supported SELECT Statement Syntax

Where...	Is...
select_expr	{*column_name* [[AS] *alias*] | *function*(...) [[AS] *alias*] | *aggregate_function*(...) [[AS] *alias*]}
column_name	the name of a field you are querying
function	{ABS | CAST | CEILING | CHAR_LENGTH | COALESCE | CONV | CHARACTER_LENGTH | CONCAT | CONVERT | CURRENT_DATE | CURRENT_TIME | CURRENT_TIMESTAMP | EXTRACT | FLOOR | HEX | INET_ATON | INET_NTOA | INET6_ATON | INET6_NTOA | LEFT | LOWER | LPAD | MID | MOD | NULLIF | OCTET_LENGTH | POSITION | RIGHT | ROUND | SUBSTRING | SYSDATE | TIME | TIMESTAMP | TRIM | UPPER}
aggregate_function	{AVG | COUNT | COUNT(DISTINCT) | MAX | MIN | SUM}
table_references	one of: • *table_reference* INNER JOIN *table_reference join_condition* • *table_reference* LEFT [OUTER] JOIN *table_reference join_condition*
table_reference	*table_name* [[AS] alias]
table_name	the name of a table you are querying
join_condition	ON *conditional_expr*

Table 2-4 Supported SELECT Statement Syntax (continued)

Where...	Is...
`conditional_expr`	an equality comparison between fields that are join-compatible; see Join Constraints, page 2-13 for more information
`where_condition`	one of: • `IS NULL` or `IS NOT NULL` • `NOT, !` • `BETWEEN ... AND ...` • `LIKE` • `=, !=, <>, >, >=, <, <=`

If you are not familiar with how supported MySQL syntax is expressed, see the following table for tips.

Table 2-5 MySQL Syntax Format

These symbols...	That is...	Represent...
brackets	`[]`	an optional clause or argument
curly brackets	`{}`	a required clause or argument
pipe	|	a choice between clauses or arguments

Any `SELECT` statement syntax not listed above is not supported. In particular, the FireSIGHT System database access feature does not support:

- `SELECT *`, that is, you must explicitly specify fields
- unions
- subqueries
- the `WITH ROLLUP` modifier to the `GROUP BY` clause
- the `INTO` clause
- the `FOR UPDATE` clause

Join Constraints

The joins you can perform on Cisco database tables are limited, for performance and other practical reasons. Cisco does not allow you to perform joins where the result is not likely to be useful for event analysis.

You can perform only inner joins and left (outer) joins. Nested joins, cross joins, natural joins, right (outer) joins, full (outer) joins, and joins with the `USING` clause are **not** supported.

The schema documentation indicates the supported joins for each table. Joins not listed are not supported. For example, you cannot join the **`compliance_event`** and **`intrusion_event`** tables on IP address fields, even though both tables contain IP address information. In addition, joins on deprecated tables and deprecated fields are not listed.

Querying Data Stored in Unfamiliar Formats

The Cisco database stores some data in formats that may not be display-friendly. The following sections detail how to perform queries on various fields so your queries are successful and your results appear as expected:

IPv6 Addresses

The Cisco database stores IPv6 addresses in binary format. For results in hex notation, use the **`HEX()`** function. To query the database on a specific IPv6 address, use the **`UNHEX()`** function.

For example, the following statement queries the **`connection_log`** table, which contains information on monitored sessions, constraining the query by a specific IPv6 address:

```
SELECT HEX(initiator_ip), HEX(responder_ip), packets_sent, bytes_sent
FROM connection_log
WHERE initiator_ip = UNHEX('20010db80000000000000000000004321');
```

IPv4 Addresses

The Cisco database stores IPv4 addresses in binary format within the same fields as IPv6 addresses. As with IPv6 addresses, use the `HEX()` function for hex notation. The database follows the RFC by filling in bits 80-95 with 1s, which yields an invalid IPv6 address. For example, the IPv4 address 10.5.15.1 would be stored as `00000000000000000000FFFF0A050F01`.

MAC Addresses

The Cisco database stores MAC addresses in binary format. For results in hex notation, use the `HEX()` function.

For example. the following statement queries the **`rna_host_mac_map`** table, which contains information on hosts with MAC addresses that are not identified with an IP address, limiting the query to the first five hosts:

```
SELECT HEX(host_id), HEX(mac_address)
FROM rna_host_mac_map
LIMIT 5;
```

Packet Data

The Cisco database stores packet data for intrusion events in binary format. For results in hex notation, use the `HEX()` function.

For example. the following statement queries the **`intrusion_event_packet`** table to obtain packet data for a particular event:

```
SELECT HEX(packet_data)
FROM intrusion_event_packet
WHERE event_id = 1234;
```

UNIX Timestamps

The Cisco database stores most timestamps as UNIX timestamps, which represent the number of seconds elapsed since 00:00:00 January 1st, 1970 (UTC). For results in your local time, use the **FROM_UNIXTIME()** function.

For example, the following statement queries the **audit_log** table, which keeps a record of all user actions on the web interface of an appliance, and returns up to 25 results:

```
SELECT FROM_UNIXTIME(action_time_sec), user, message
FROM audit_log
LIMIT 0, 25;
```

Keep in mind that all times in the database are in UTC. Although the **CONVERT_TZ()** function is allowed, it only provides results in UTC.

Note that some events have microsecond resolution associated with them. Use the **CONCAT()** and **LPAD()** functions to concatenate the UNIX timestamp and the microsecond increment. For example, the following statement queries the **intrusion_event** table:

```
SELECT CONCAT(FROM_UNIXTIME(event_time_sec)), '.', LPAD (event_time_usec, 6, '0')),
HEX(host_id),
rule_message
FROM intrusion_event
LIMIT 0, 25;
```

To query the database for events with a particular UNIX timestamp, use the **UNIX_TIMESTAMP()** function.

Limiting Queries for Performance Reasons

Although the system limits the joins you can perform on Cisco database tables, it does still allow some expensive queries - queries that may negatively impact the performance of your Defense Center.

Therefore, you should try to limit the result set for large tables. Strategies include:

- constraining queries to a specific time range
- constraining queries by IP address
- using the **LIMIT** clause

Depending on your deployment, querying many tables may require a limited result set. In particular, the following tables can contain up to 100 million events on a DC3000:

- `fireamp_event`
- `intrusion_event`
- `intrusion_event_packet`
- **connection_log** (pre-Version 5.0 name: **rna_flow**)

- **connection_summary** (pre-Version 5.0 name: **rna_flow_summary**)

Queries on network map tables may also be expensive, depending on the number of hosts the system has detected on your monitored network.

Query Tips

The following sections provide tips on ensuring unique results when you build queries that include detection engines or intrusion events.

Device Names

Device names are not necessarily unique across multiple Defense Centers. To ensure uniqueness, include a specific device UUID in your query.

Intrusion Events

To uniquely match an intrusion event across multiple managed devices, include the following fields in your query of the **intrusion_event** table:

- intrusion_event.event_id
- intrusion_event.event_time_sec
- intrusion_event.sensor_uuid

Sample Queries

The following sections contain sample queries that illustrate how you can use the database access feature:

Caution Performing some of these sample queries may be expensive, depending on your deployment. See Limiting Queries for Performance Reasons, page 2-15 for more information.

Audit Records for a User

The following query returns all records in the audit log for a particular user, displaying all timestamps in UTC:

```
SELECT FROM_UNIXTIME(action_time_sec), user, message
```

```
FROM audit_log
WHERE user = 'eventanalyst';
```

Intrusion Events by Priority and Classification

The following query duplicates the Drilldown of Event, Priority, and Classification view in the Events By Priority and Classification workflow. If you have not changed the default Intrusion Events workflow in your user preferences, this is the first page you see when you select **Analysis > Intrusion Events** on the Defense Center web interface:

```
SELECT rule_message, priority, rule_classification, count(*) as Count
FROM intrusion_event
WHERE reviewed="0" GROUP BY rule_message, priority, rule_classification
ORDER BY Count
DESCLIMIT 0, 25;
```

Intrusion Events and Their Associated Policies

The following query lists intrusion events from the specified week. For each event it shows the associated intrusion policy violation and rule classification.

```
SELECT FROM_UNIXTIME(event_time_sec) AS event_time, event_id AS intrusion_event,
intrusion_event_policy_name AS policy, rule_classification AS classification
FROM intrusion_event
WHERE event_time_sec BETWEEN UNIX_TIMESTAMP('2011-10-01 00:00:00') AND
UNIX_TIMESTAMP('2011-10-07 23:59:59')
ORDER BY policy ASC;
```

Lists of Detected Hosts

The following query returns the basic information in the hosts network map for all MAC hosts (hosts without an IP address) detected on your network, along with the hardware vendor for each NIC:

```
SELECT HEX(mac_address), mac_vendor, host_type, FROM_UNIXTIME(last_seen_sec)
FROM rna_mac_host;
```

The following query maps IP addresses to MAC addresses:

```
SELECT HEX(ipaddr), HEX(mac_address), HEX(host_id)
FROM rna_host_ip_map LEFT JOIN rna_host_mac_map on
rna_host_ip_map.host_id=rna_host_mac_map.host_id;
```

List of Detected Servers

The following query joins two related tables to give you a list of the servers detected on your network along with many of their attributes, similar to what you can see in a table view of servers on the Defense Center's web interface:

```
SELECT FROM_UNIXTIME(s.last_used_sec), HEX(s.host_id), s.port, s.protocol, s.hits,
i.service_name, i.vendor, i.version, i.source_type, s.confidence
FROM AS s
LEFT JOIN rna_ip_host_service_info AS i ON (s.host_id = i.host_id AND s.port = i.port AND
s.protocol =
i.protocol);
```

Note that this query left joins the tables on the set of `host_id`, `port`, and `protocol`, as required by Database Access. See rna_host_service Joins, page 6-34 and rna_host_service_info Joins, page 6-39.

Server Vulnerabilities on Your Network

The following query joins two vulnerability-related tables to give you a list of valid server-related vulnerabilities detected for a particular host, along with whether each vulnerability is exploitable across a network:

```
SELECT h.rna_vuln_id, v.title, v.remote
FROM rna_host_service_vulns AS h
LEFT JOIN rna_vuln AS v ON (h.rna_vuln_id = v.rna_vuln_id)
WHERE h.ip_address = INET_ATON('10.10.10.4')
AND h.invalid = 0;
```

Note that this query left joins the tables on **`rna_vuln_id`**, as required by rna_host_service_vulns, page 6-46 and rna_vuln Joins, page 6-56.

Operating System Summary

The following query duplicates the Summary of OS Names page in the Operating System Summary workflow. If you have not changed the default workflow in your user preferences, this is the first page you see when you select **Analysis > Hosts** on the Defense Center web interface, then select **Hosts**:

```
SELECT vendor, product, count(*) AS total
FROM rna_host_os
GROUP BY vendor, product
ORDER BY total DESC;
```

Operating System Vulnerabilities for a Host

The following query joins two vulnerability-related tables to give you a list of valid operating system-related vulnerabilities detected for a particular host, along with whether each vulnerability is exploitable across a network:

```
SELECT h.rna_vuln_id, v.title, v.remote
FROM rna_host_os_vulns AS h
LEFT JOIN rna_vuln AS v ON (h.rna_vuln_id = v.rna_vuln_id)
WHERE h.host_id = UNHEX('9610B6E6F1784DA4B39BEA7A210AAD68')
```

```
AND h.invalid = 0;
```

Note that this query left joins the tables on **rna_vuln_id**, as required by Database Access. See rna_host_os_vulns, page 6-29 and rna_vuln Joins, page 6-56.

Host Violation Count

The following query duplicates the Host Violation Count page in the Host Violation Count workflow. If you have not changed the default Compliance White List Violations workflow in your user preferences, this is the first page you see when you select **Analysis > Correlation > White List Violations** on the Defense Center's web interface.

```
SELECT host_id, HEX(host_id), white_list_name, count(*) AS total

FROM white_list_violation

GROUP BY host_id, white_list_name

ORDER BY total DESC;
```

Schema: System-Level Tables

This chapter contains information on the schema and supported joins for system-level functions, including auditing, appliance health monitoring, malware detection, and logging of security updates.

For more information, see the sections listed in the following table.

Table 3-1 Schema for System-Level Tables

See...	For the table that stores information on...	Version
audit_log, page 3-1	User interactions with the appliance's web interface.	4.10.x+
fireamp_event, page 3-2	FireAMP malware detection and quarantine events.	5.1+
health_event, page 3-8	Health status events for monitored appliances.	4.10.x+
sru_import_log, page 3-10	Rule updates that have been imported on your appliances.	5.0+

audit_log

The **`audit_log`** table contains information on FireSIGHT System users' interactions with the web interface. Keep in mind that the audit log stores records for the local appliance only, not for managed appliances.

For more information, see the following sections:

- audit_log Fields, page 3-1
- audit_log Joins, page 3-2
- audit_log Sample Query, page 3-2

audit_log Fields

The following table describes the database fields you can access in the **`audit_log`** table.

Table 3-2 audit_log Fields

Field	Description
`action_time_sec`	The UNIX timestamp of the date and time the appliance generated the audit record.
`message`	The action the user performed.
`source`	The IP address of the web interface user's host, in dotted-decimal notation.

Table 3-2 audit_log Fields (continued)

Field	Description
subsystem	The menu path the user followed to generate the audit record.
user	The user name of the user who triggered the audit event.

audit_log Joins

You cannot perform joins on the **audit_log** table.

audit_log Sample Query

The following query returns up to the 25 most recent audit log entries, sorted by time.

```
SELECT from_unixtime(action_time_sec)
AS Time, user, subsystem, message, source, count(*)
AS Total
FROM audit_log
GROUP BY source, subsystem, user, message
ORDER BY source DESC;
```

fireamp_event

The **fireamp_event** table contains information on malware events. These events contain information on malware detected or quarantined within a cloud, the detection method, and hosts and users affected by the malware. New fields were added to identify the application which triggered the event, how the event is handled, and to correlate the event with connection, intrusion, and file events.

For more information, see the following sections:

- fireamp_event Fields, page 3-2
- fireamp_event Joins, page 3-8
- fireamp_event Sample Query, page 3-8

fireamp_event Fields

The following table describes the database fields you can access in the **fireamp_event** table.

Table 3-3 fireamp_event Fields

Field	Description
application_id	ID number that maps to the application performing the file transfer.
application_name	Name of the application performing the transfer.

Table 3-3 fireamp_event Fields (continued)

Field	Description
cert_valid_end_date	The Unix timestamp on which the SSL certificate used in the connection ceases to be valid.
cert_valid_start_date	The Unix timestamp when the SSL certificate used in the connection was issued.
client_application_id	The internal identification number for the client application, if applicable.
client_application_name	The name of the client application, if applicable.
cloud_name	The name of the cloud service from which the FireAMP event originated. Each cloud_name value has an associated cloud_uuid value.
cloud_uuid	The internal unique ID of the cloud service from which the FireAMP event originated. Each cloud_uuid value has an associated cloud_name value.
connection_sec	UNIX timestamp (seconds since 00:00:00 01/01/1970) of the connection event associated with the malware event.
counter	Specific counter for the event, used to distinguish among multiple events that happened during the same second.
detection_name	The name of the detected or quarantined malware.
detector_type	The detector that detected the malware. Each detector_type value has an associated detector_type_id. The possible display values and the associated IDs are: • ClamAV — 128 • ETHOS — 8 • SPERO — 32 • SHA — 4 • Tetra — 64
detector_type_id	The internal ID of the detection technology that detected the malware. Each detector_type_id value has an associated detector_type value. The possible display values and the associated types are: • 4 — SHA • 8 — ETHOS • 32 — SPERO • 64 — Tetra • 128 — ClamAV
direction	Value that indicates whether the file was uploaded or downloaded. Can have the following values: • Download • Upload Currently the value depends on the protocol (for example, if the connection is HTTP it is a download).

Table 3-3 fireamp_event Fields (continued)

Field	Description
disposition	The malware status of the file. Possible values include: • CLEAN — The file is clean and does not contain malware. • UNKNOWN — It is unknown whether the file contains malware. • MALWARE — The file contains malware. • UNAVAILABLE — The software was unable to send a request to the Cisco cloud for a disposition, or the Cisco cloud services did not respond to the request. • CUSTOM SIGNATURE — The file matches a user-defined hash, and is treated in a fashion designated by the user.
dst_continent_name	The name of the continent of the destination host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
dst_country_id	Code for the country of the destination host.
dst_country_name	Name of the country of the destination host.
dst_ip_address_v6	This field has been deprecated and will now return null.
dst_ipaddr	A binary representation of the IPv4 or IPv6 address for the destination of the connection.
dst_port	Port number for the destination of the connection.
endpoint_user	The user determined by the Cisco FireAMP agent if the event was detected by the Cisco cloud. This user is not associated with LDAP and does not appear in the discovered_users table.
event_description	The additional event information associated with the event type.
event_id	The internal unique ID of the FireAMP event.
event_subtype	The action that led to malware detection. Each event_subtype value has an associated event_subtype_id value. The possible display values and the associated IDs are: • Create — 1 • Execute — 2 • Move — 22 • Scan — 4

Table 3-3 fireamp_event Fields (continued)

Field	Description
event_subtype_id	The internal ID of the action that led to malware detection. Each event_subtype_id value has an associated event_subtype value. The possible display values and the associated subtypes are: • 1 — Create • 2 — Execute • 4 — Scan • 22 — Move
event_type	The type of FireAMP event. Each event_type value has an associated event_type_id value. The possible display values and the associated IDs are: • Blocked Execution — 553648168 • Cloud Recall Quarantine — 553648155 • Cloud Recall Quarantine Attempt Failed — 2164260893 • Cloud Recall Quarantine Started — 553648147 • Cloud Recall Restore from Quarantine — 553648154 • Cloud Recall Restore from Quarantine Failed — 2164260892 • Cloud Recall Restore from Quarantine Started — 553648146 • FireAMP IOC — 1107296256 • Quarantine Failure — 2164260880 • Quarantined Item Restored — 553648149 • Quarantine Restore Failed — 2164260884 • Quarantine Restore Started — 553648150 • Scan Completed, No Detections — 554696715 • Scan Completed With Detections — 1091567628 • Scan Failed — 2165309453 • Scan Started — 554696714 • Threat Detected — 1090519054 • Threat Detected in Exclusion — 553648145 • Threat Detected in Network File Transfer — 1 • Threat Detected in Network File Transfer (Retrospective) — 2 • Threat Quarantined — 553648143

Table 3-3 fireamp_event Fields (continued)

Field	Description
event_type_id	The internal ID of the FireAMP event type. Each event_type_id value has an associated event_type value. The possible display values and the associated types are: • 553648143 — Threat Quarantined • 553648145 — Threat Detected in Exclusion • 553648146 — Cloud Recall Restore from Quarantine Started • 553648147 — Cloud Recall Quarantine Started • 553648149 — Quarantined Item Restored • 553648150 — Quarantine Restore Started • 553648154 — Cloud Recall Restore from Quarantine • 553648155 — Cloud Recall Quarantine • 553648168 — Blocked Execution • 554696714 — Scan Started • 554696715 — Scan Completed, No Detections • 1090519054 — Threat Detected • 1091567628 — Scan Completed With Detections • 1107296256 — FireAMP IOC • 2164260880 — Quarantine Failure • 2164260893 — Cloud Recall Quarantine Attempt Failed • 2164260884 — Quarantine Restore Failed • 2164260892 — Cloud Recall Restore from Quarantine Failed • 2165309453 — Scan Failed
file_name	The name of the detected or quarantined file. This name can contain UTF-8 characters.
file_path	The file path, not including the file name, of the detected or quarantined file.
file_sha	The SHA-256 hash value of the detected or quarantined file.
file_size	The size in bytes of the detected or quarantined file.
file_timestamp	The creation timestamp of the detected or quarantined file.
file_type	The file type of the detected or quarantined file.
file_type_id	The internal ID of the file type of the detected or quarantined file.
instance_id	Numerical ID of the Snort instance on the managed device that generated the event.
ioc_count	Number of indications of compromise found in the event.
parent_file_name	The name of the file accessing the detected or quarantined file when detection occurred.
parent_file_sha	The SHA-256 hash value of the parent file accessing the detected or quarantined file when detection occurred.
policy_uuid	Identification number that acts as a unique identifier for the access control policy that triggered the event.

Table 3-3 fireamp_event Fields (continued)

Field	Description
retroactive_ disposition	Disposition of the file if the disposition is updated. If the disposition is not updated, this field contains the same value as the disposition field. The possible values are the same as the disposition field.
score	A numeric value from 0 to 100 based on the potentially malicious behaviors observed during dynamic analysis.
security_context	Description of the security context (virtual firewall) that the traffic passed through. Note that the system only populates this field for ASA FirePOWER devices in multi-context mode.
sensor_address	IP address of the device that generated the event.
sensor_id	ID of the device that generated the event.
sensor_name	The text name of the managed device that generated the event record. This field is null when the event refers to the reporting device itself, rather than to a connected device.
sensor_uuid	A unique identifier for the managed device, or 0 if fireamp_event.sensor_name is null.
src_continent_name	The name of the continent of the source host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
src_country_id	Code for the country of the source host.
src_country_name	Name of the country of the source host.
src_ip_address_v6	Field deprecated in Version 5.2. Returns null for all queries.
src_ipaddr	A binary representation of the IPv4 or IPv6 address for the source of the connection.
src_port	Port number for the source of the connection.
ssl_issuer_common_name	Issuer Common Name from the SSL certificate. This is typically the host and domain name of the certificate issuer, but may contain other information.
ssl_issuer_country	The country of the SSL certificate issuer.
ssl_issuer_organization	The organization of the SSL certificate issuer.
ssl_issuer_organization_unit	The organizational unit of the SSL certificate issuer.
ssl_serial_number	The serial number of the SSL certificate, assigned by the issuing CA.
ssl_subject_common_name	Subject Common name from the SSL certificate This is typically the host and domain name of the certificate subject, but may contain other information.
ssl_subject_country	The country of the SSL certificate subject.
ssl_subject_organization	The organization of the SSL certificate subject.
ssl_subject_organization_unit	The organizational unit of the SSL certificate subject.

Table 3-3 fireamp_event Fields (continued)

Field	Description
threat_name	Name of the threat.
timestamp	The FireAMP event generation timestamp.
url	The URL of the source of the connection.
user_id	An internal identification number for the user who last logged into the host that sent or received the file. This user is in the **discovered_users** table.
username	The name of the user who last logged into the host that sent or received the file.
web_application_id	The internal identification number for the web application, if applicable.
web_application_name	Name of the web application, if applicable.

fireamp_event Joins

The following table describes the joins you can perform on the **fireamp_event** table.

Table 3-4 fireamp_event Joins

You can join this table on...	And...
dst_ipaddr or src_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

fireamp_event Sample Query

The following query returns 25 malware events associated with the specified user, sorted by timestamp in ascending order.

```
SELECT event_id, timestamp, src_ipaddr, dst_ipaddr, username, cloud_name, event_type,
event_subtype, event_description, detection_name, detector_type, file_name,
parent_file_name

FROM fireamp_event

WHERE username="username" ORDER BY timestamp ASC

LIMIT 25;
```

health_event

The **health_event** table contains information on health events generated by the FireSIGHT System.

For more information, see the following sections:

- health_event Fields, page 3-9
- health_event Joins, page 3-9
- health_event Sample Query, page 3-9

health_event Fields

The following table describes the database fields you can access in the **health_event** table.

Table 3-5 health_event Fields

Field	Description
description	The description of the condition that caused the associated health module to generate the health event. For example, health events generated when a process was unable to execute are labeled Unable to Execute.
event_time_sec	The UNIX timestamp of the date and time the Defense Center generated the health event.
id	The internal identification number for the event.
module_name	The name of the health module that generated the event.
sensor_name	The text name of the managed device that generated the event record. This field is **null** when the health event refers to the reporting device itself, rather than to a connected one.
sensor_uuid	A unique identifier for the managed device, or zero if sensor_name is null.
status	The health monitor status that has been reported for the appliance identified in sensor_uuid. Values are: • red — Critical status. Limits have been exceeded for at least one health module on the appliance and the problem has not been corrected. • yellow — Warning status. Limits have been exceeded for at least one health module on the appliance and the problem has not been corrected. • green — Normal status. All health modules on the appliance are running within the limits configured in the health policy applied to the appliance. • recovered — All health modules on the appliance are running within the limits configured in the health policy applied to the appliance, including modules that were in a Critical or Warning state. • disabled — Either the appliance is disabled or blacklisted, or is currently unreachable, or has no health policy applied to it. • error — At least one health monitoring module has failed on the appliance and has not been successfully re-run since the failure occurred
units	The unit of measure for results obtained by the health test. For example, **%** (of Disk Usage).
value	The number of units of the result obtained by the health test. For example, the value of 80% is 80.

health_event Joins

You cannot perform joins on the **health_event** table.

health_event Sample Query

The following query returns up to the 25 most recent health events logged within the defined time frame.

```
SELECT module_name, FROM_UNIXTIME(event_time_sec)
```

```
AS event_time, description, value, units, status, sensor_name
FROM health_event
WHERE event_time_sec
BETWEEN UNIX_TIMESTAMP("2011-10-01 00:00:00")
AND UNIX_TIMESTAMP("2011-10-07 23:59:59")
ORDER BY event_time DESC
LIMIT 0, 25;
```

sru_import_log

The **sru_import_log** table contains information on rule update processes that have been run on your appliances. The **sru_import_log** table supersedes the deprecated **seu_import_log** table starting with Version 5.0 of the FireSIGHT System.

For more information, see the following sections:

- sru_import_log Fields, page 3-10
- sru_import_log Joins, page 3-11
- sru_import_log Sample Query, page 3-11

sru_import_log Fields

The following table describes the database fields you can access in the **sru_import_log** table.

Table 3-6 sru_import_log Fields

Field	Description
action	Indicates the action that has occurred for the imported rule update object type: • apply — The Reapply intrusion policies after the Rule Update import completes option was enabled for the import • changed — For a rule update component or rule, the rule update component was modified, or the rule has a higher revision number and the same GID and SID • collision — For a rule update component or rule, import was skipped because its revision conflicts with an existing component or rule on the appliance • deleted — For rules, the rule has been deleted from the rule update • disabled — For rules, the rule has been disabled in a default policy provided by Cisco • drop — For rules, the rule has been set to Drop and Generate Events in a default policy provided by Cisco • enabled — For a rule update, edit, a preprocessor, rule, or other feature provided by the rule update has been enabled in a default policy provided by Cisco • error — For a rule update or local rule file, the import failed • new — For a rule, this is the first time the object has been stored on this appliance
detail	Either a comment string unique for the change applied by the imported rule update to the component or rule, or blank, for a rule that has not changed.
generator_id	The GID for the generator for a rule.
import_time_sec	The UNIX timestamp of the date and time the rule update import was logged.
name	The name of the imported object. For rules, this corresponds to the rule message. For rule update components, this is the component name, such as online help or Snort.
policy	All, indicating that a rule is included in all default policies.
revision	Revision number for a rule.
signature_id	The SID for a rule or set of rules, decoder, or preprocessor.
sru_name	Descriptive name of the rule update.
sru_uuid	A unique identifier for the rule update.
type	Type of imported object in the rule update: update, rule, variable, and so forth.

sru_import_log Joins

You cannot perform joins on the **sru_import_log** table.

sru_import_log Sample Query

The following query returns up to 25 results in descending order, sorted by timestamp.

```
SELECT FROM_UNIXTIME(import_time_sec)
AS time, name, type, action, generator_id, signature_id, revision, policy
FROM sru_import_log
```

```
ORDER BY time DESC
LIMIT 0, 25;
```

Schema: Intrusion Tables

This chapter contains information on the schema and supported joins for intrusion events, the packets that triggered the events, and the associated rule messages.

For more information, see the sections listed in the following table.

Table 4-1 Schema for Intrusion Tables

See...	For the table that stores information on...	Version
intrusion_event, page 4-1	Intrusion events, which include the date, time, type of exploit, and contextual information about the source and target of an attack.	4.10.x+
intrusion_event_packet, page 4-7	The content of the packet or packets that triggered an intrusion event.	4.10.x+
rule_message, page 4-8	Rule messages for intrusion events, including the associated generator ID (GID), signature ID (SID), and version data.	4.10.x+
rule_documentation, page 4-9	Information on rules, including the attack scenarios, affected systems, and information on when the rule was created and by whom.	5.2+

intrusion_event

The **`intrusion_event`** table contains information on possible intrusions identified by the FireSIGHT System. For each possible intrusion, the system generates an event and an associated record in the database, which contains the date, time, type of exploit, access control policy and rule, intrusion policy and rule, and other contextual information about the source and target of the attack.

Tip For packet-based events, a copy of the packet or packets that triggered the event may also be available; see intrusion_event_packet Sample Query, page 4-8.

For more information, see the following sections:

- `intrusion_event Fields, page 4-2`
- `intrusion_event Joins, page 4-6`
- `intrusion_event Sample Query, page 4-7`

intrusion_event Fields

The following table describes the database fields you can access in the **`intrusion_event`** table.

Table 4-2 intrusion_event Fields

Field	Description
`access_control_policy_name`	The access control policy associated with the intrusion policy that generated the intrusion event. Note that the access control policy name and access control rule name combination is unique for a Defense Center.
`access_control_policy_UUID`	The UUID of the access control policy associated with the intrusion policy that generated the intrusion event.
`access_control_rule_id`	The internal identification number of the access control rule associated with the intrusion policy that generated the intrusion event.
`access_control_rule_name`	The name of the access control rule associated with the intrusion policy that generated the intrusion event. Note that the access control rule name is unique within a policy but not across different policies.
`application_protocol_id`	The internal identification number of the application protocol.
`application_protocol_name`	One of: • the name of the application, if a positive identification can be made • `pending` if the system requires more data • blank if there is no application information in the connection
`blocked`	The value indicating what happened to the packet that triggered the intrusion event: • `0` — Packet not dropped • `1` — Packet dropped (inline, switched, or routed deployment) • `2` — Packet that triggered the event would have been dropped, if the intrusion policy had been applied to a device configured in inline, switched, or routed deployment
`client_application_id`	The internal identification number of the client application that was used in the intrusion event.
`client_application_name`	The client application, if available, that was used in the intrusion event. One of: • the name of the application, if a positive identification can be made • a generic client name if the system detects a client application but cannot identify a specific one. • `null` if there is no application information in the connection
`connection_sec`	UNIX timestamp (seconds since 00:00:00 01/01/1970) of the connection event associated with the intrusion event.
`counter`	Number that is incremented for each connection event in a given second, and is used to differentiate among multiple connection events that happen during the same second.
`detection_engine_name`	Field deprecated in Version 5.0. Returns `null` for all queries.
`detection_engine_uuid`	Field deprecated in Version 5.0. Returns `null` for all queries.

Table 4-2 intrusion_event Fields (continued)

Field	Description
dst_continent_name	The name of the continent of the destination host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
dst_country_id	Code for the country of the destination host.
dst_country_name	Name of the country of the destination host.
dst_ip_address	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
dst_ip_address_v6	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
dst_ipaddr	A binary representation of the IPv4 or IPv6 address for the destination host involved in the triggering event.
dst_port	Either: • the destination port number, if the event protocol type is TCP or UDP • the ICMP code, if the event protocol type is ICMP
dst_user_dept	The department of the destination user.
dst_user_email	The email address of the destination user.
dst_user_first_name	The first name of the destination user.
dst_user_id	The internal identification number for the destination user; that is, the user who last logged into the destination host before the intrusion event occurred.
dst_user_last_name	The last name of the destination user.
dst_user_last_seen_sec	The UNIX timestamp of the date and time when the system last reported a login for the destination user.
dst_user_last_updated_sec	The UNIX timestamp of the date and time when the system last updated the destination user's record.
dst_user_name	The user name for the destination user.
dst_user_phone	The telephone number for the destination user.
event_id	The internal identification number for the event. Uniquely identifies an event on the Defense Center.
event_time_sec	The UNIX timestamp of the date and time when the event packet was captured.
event_time_usec	The microsecond increment of the event timestamp. If microsecond resolution is not available, this value is 0.

Table 4-2 intrusion_event Fields (continued)

Field	Description
icmp_code	ICMP code if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.
icmp_type	ICMP type if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.
impact	The impact flag value of the event. Integer values are: • 1 — Red (vulnerable) • 2 — Orange (potentially vulnerable) • 3 — Yellow (currently not vulnerable) • 4 — Blue (unknown target) • 5 — Gray (unknown impact)
instance_id	Numerical ID of the Snort instance on the managed device that generated the event.
interface_egress_name	The name of the interface for the outbound traffic.
interface_ingress_name	The name of the interface for the inbound traffic.
intrusion_event_policy_uuid	A unique identifier for the intrusion policy that triggered the intrusion event.
intrusion_event_policy_name	The intrusion policy that generated the intrusion event.
ioc_count	Number of indications of compromise found in the event.
network_analysis_policy_name	The network analysis policy associated with the intrusion policy that generated the intrusion event.
network_analysis_policy_UUID	The UUID of the network analysis policy associated with the intrusion policy that generated the intrusion event.
priority	The priority for the rule classification associated with the event. Rule priority is set in the user interface.
protocol_name	The text name of the traffic protocol associated with the intrusion event.
protocol_num	The IANA number of the protocol as listed in http://www.iana.org/assignments/protocol-numbers.
reviewed	Whether the intrusion event has been marked as reviewed: • 1 — Reviewed • 0 — Not reviewed
rule_classification	The description of the rule classification associated with the intrusion event, which usually describes the attack detected by the rule that triggered the event. For example: A Network Trojan was Detected.
rule_classification_id	The identification number for the rule classification associated with the intrusion event.
rule_generator	The component that generated the intrusion event. The generator can be either a rules engine, decoder, or preprocessor.
rule_generator_id	The generator ID (GID) of the component named in rule_generator that generated the intrusion event.

Table 4-2 intrusion_event Fields (continued)

Field	Description
rule_message	Explanatory text for the event. For rule-based intrusion events, the message is generated from the rule. For decoder- and preprocessor-based events, the message is hard coded.
rule_revision	The revision number of the rule associated with the intrusion event.
rule_signature_id	The signature ID (SID) for the intrusion event. Identifies the specific rule, decoder message, or preprocessor message that caused the event to be generated.
security_context	Description of the security context (virtual firewall) that the traffic passed through. Note that the system only populates this field for ASA FirePOWER devices in multi-context mode.
security_zone_egress_name	The egress security zone in the intrusion event that triggered the policy violation.
security_zone_ingress_name	The ingress security zone in the intrusion event that triggered the policy violation.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
src_continent_name	The name of the continent of the destination host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
src_country_id	Code for the country of the destination host.
src_country_name	Name of the country of the destination host.
src_ip_address	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
src_ip_address_v6	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
src_ipaddr	A binary representation of the IPv4 or IPv6 address for the source host involved in the triggering event.
src_port	Either: • the source port number, if the event protocol type is TCP or UDP • the ICMP type, if the event protocol type is ICMP
src_user_dept	The department of the source user.
src_user_email	The email address for the source user.
src_user_first_name	The first name of the source user.

Table 4-2 intrusion_event Fields (continued)

Field	Description
src_user_id	The internal identification number for the source user; that is, the user who last logged into the source host before the intrusion event occurred.
src_user_last_name	The last name of the source user.
src_user_last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the source user.
src_user_last_updated_sec	The UNIX timestamp of the date and time the source user's record was last updated.
src_user_name	The user name for the source user.
src_user_phone	The source user's phone number.
vlan_id	The identification number of the innermost VLAN associated with the packet that triggered the intrusion event.
web_application_id	The internal identification number of the web application that was used in the intrusion event, if applicable.
web_application_name	The web application that was used in the intrusion event, if applicable. One of: • the name of the application, if a positive identification can be made • web browsing if the system detects an application protocol of HTTP but cannot identify a specific web application • blank if the connection has no HTTP traffic

intrusion_event Joins

The following table describes the joins you can perform on the **intrusion_event** table.

Table 4-3 intrusion_event Joins

You can join this table on...	And...
application_protocol_id or client_application_id or web_application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id
dst_ipaddr or src_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

intrusion_event Sample Query

The following query returns the 25 most common unreviewed intrusion event results, sorted in descending order based on `Count`.

```
SELECT rule_message, priority, rule_classification, count(*) as Count

FROM intrusion_event

WHERE reviewed="0"

GROUP BY rule_message, priority, rule_classification

ORDER BY Count DESCLIMIT 0, 25;
```

intrusion_event_packet

The **`intrusion_event_packet`** table contains information on content of the packet or packets that triggered an intrusion event. Keep in mind if you prohibited packet transfer from your managed devices to the Defense Center, the **`intrusion_event_packet`** table contains no data.

For more information, see the following sections:

- intrusion_event_packet Fields, page 4-7
- intrusion_event_packet Joins, page 4-8
- intrusion_event_packet Sample Query, page 4-8

intrusion_event_packet Fields

The following table describes the database fields you can access in the **`intrusion_event_packet`** table.

Table 4-4 intrusion_event_packet Fields

Field	Description
`detection_engine_name`	Field deprecated in Version 5.0. Returns `null` for all queries.
`detection_engine_uuid`	Field deprecated in Version 5.0. Returns `null` for all queries.
`event_id`	The identification number for the event. The ID is unique on a given managed device.
`linktype`	An internal key that indicates the format of the packet's outer layer; used by the managed device to correctly decode the packet. Only link type `1` is supported.
`packet_data`	The contents of the packet that triggered the event.
`packet_time_sec`	The UNIX timestamp of the date and time the event packet was captured.
`packet_time_usec`	The microsecond increment of the event timestamp. If microsecond resolution is not available, this value is `0`.
`sensor_address`	The IP address of the managed device that generated the event. Format is *`ipv4_address,ipv6_address`*.
`sensor_name`	The name of the managed device that generated the intrusion event.
`sensor_uuid`	A unique identifier for the managed device, or `0` if `sensor_name` is `null`.

intrusion_event_packet Joins

You cannot perform joins on the **intrusion_event_packet** table.

intrusion_event_packet Sample Query

The following query returns the packet information for all packets matching the selected event ID.

```
SELECT event_id, packet_time_sec, sensor_address, packet_data
FROM intrusion_event_packet
WHERE event_id="1";
```

rule_message

The **rule_message** table is a master list of the rule messages for intrusion rules. Each rule message is accompanied by its identifying information.

For more information, see the following sections:

- rule_message Fields, page 4-8
- rule_message Joins, page 4-8
- rule_message Sample Query, page 4-9

rule_message Fields

The following table describes the database fields you can access in the **rule_message** table.

Table 4-5 rule_message Fields

Field	Description
generator_id	The GID of the component that triggers the rule.
message	The message associated with the rule that is triggered.
rev_uuid	A unique identifier for the rule revision.
revision	The revision number for the rule.
signature_id	The rule identification number as it is rendered in the appliance user interface.
uuid	A unique identifier for the rule.

rule_message Joins

You cannot perform joins on the **rule_message** table.

rule_message Sample Query

The following query returns the intrusion rule message for the intrusion rule that has a GID of `1` and a SID of `1200`.

```
SELECT generator_id, signature_id, revision, message

FROM rule_message

WHERE generator_id="1"

AND signature_id="1200";
```

rule_documentation

The **`rule_documentation`** table contains information about rules used to generate alerts.

For more information, see the following sections:

- rule_documentation Fields, page 4-9
- rule_documentation Joins, page 4-10
- rule_documentation Sample Query, page 4-10

rule_documentation Fields

The following table describes the database fields you can access in the **`rule_documentation`** table.

Table 4-6 rule_documentation Fields

Field	Description
`additional_references`	Additional information and references.
`affected_systems`	Systems affected by the vulnerability.
`attack_scenarios`	Examples of possible attacks.
`contributors`	Contact information for the author of the rule and other relevant documentation.
`corrective_action`	Information regarding patches, upgrades, or other means to remove or mitigate the vulnerability.
`detailed_information`	Information regarding the underlying vulnerability, what the rule actually looks for, and what systems are affected.
`ease_of_attack`	Whether the attack is considered simple, medium, hard, or difficult, and whether or not it can be performed using a script.
`false_negatives`	Examples that may result in a false negative. The default value is `None Known`.
`false_positives`	Examples that may result in a false positive. The default value is `None Known`.
`impact`	How a compromise that uses this vulnerability may impact various systems.
`rule_revision`	Rule revision number.
`rule_signature_id`	Rule identification number that corresponds with the event.
`summary`	Explanation of the threat or vulnerability.
`updated`	The UNIX timestamp of the date and time the rule was last updated.

rule_documentation Joins

You cannot perform joins on the **rule_documentation** table.

rule_documentation Sample Query

The following query returns the attack scenarios, corrective action, impact, and summary for the intrusion rule that has an ID of 1.

```
SELECT attack_scenarios, corrective_action, impact, summary

FROM rule_documentation

WHERE rule_signature_id="1";
```

Schema: Statistics Tracking Tables

This chapter contains information on the schema and supported joins for application and URL statistics tracking tables. These tables collect statistical information on:

- access control and intrusion events by application and by user
- bandwidth usage and connection decisions by application and by user
- bandwidth usage and connection decisions by URL reputation (risk) and by URL business relevance

For links to details on each table, see the following table.

Table 5-1 Application and URL Statistics Tables

See	For the table that stores statistics on...	Version
app_ids_stats_current_timeframe, page 5-4	Access control and intrusion protection activity, by application and a range of application attributes.	5.0+
app_stats_current_timeframe, page 5-6	Traffic volume and system access control activity (connections allowed or denied), by application and a range of application attributes.	5.0+
geolocation_stats_current_timeframe, page 5-7	Access control activity by location.	5.2+
ids_impact_stats_current_timeframe, page 5-9	Statistics for intrusion events (connections blocked and would have dropped) by impact levels.	5.1.1+
session_stats_current_timeframe, page 5-10	Contain statistics for all connections. Statistics can be extracted based on bytes, connection, sensor, and time.	5.2+
ssl_stats_current_timeframe, page 5-11	Contain statistics for SSL connections. Statistics can be extracted based on bytes, connection, sensor, and time.	5.4+
storage_stats_by_disposition_current_timeframe, page 5-14	Contain statistics for files based on disposition. Statistics can be extracted based on bytes, disposition, sensor, and time.	5.3+
storage_stats_by_file_type_current_timeframe, page 5-15	Contain statistics for files based on file type. Statistics can be extracted based on bytes, file type, sensor, and time.	5.3+
transmission_stats_by_file_type_current_timeframe, page 5-16	Contain statistics for connections based on file type. Statistics can be extracted based on bytes, connection, file type, sensor, and time.	5.3+
url_category_stats_current_timeframe, page 5-17	Traffic volume and system access control activity (connections allowed or denied), by the category of the requested website.	5.0+

Table 5-1 Application and URL Statistics Tables (continued)

See	For the table that stores statistics on...	Version
url_reputation_stats_current_timeframe, page 5-19	Traffic volume and system access control activity (connections allowed or denied), by the reputation of the requested website.	5.0+
user_ids_stats_current_timeframe, page 5-20	Access control and intrusion protection activity, by user.	5.0+
user_stats_current_timeframe, page 5-21	Traffic volume and system access control activity (connections allowed or denied), by user.	5.0+

Understanding Statistics Tracking Tables

A table's name ends with **current_day**, **current_month**, or **current_year** to indicate the timeframe of its data. For example, the **app_ids_stats_current_***timeframe* describes **app_stats_current_day**, **app_stats_current_month**, and **app_stats_current_year**. The **app_stats_current_year** table stores statistics for 360 days; the **current_month** table stores statistics for 30 days.

Each time the Defense Center receives raw counts from managed devices in your network, it updates all three table types, but does so at successively coarser resolution. The **current_day** table has the finest resolution (15 seconds or 5 minutes, depending on the particular table); the **current_year** table has the coarsest resolution (24 hours). See Storage Characteristics for Statistics Tracking Tables, page 5-2 for specific information.

Storage Characteristics for Statistics Tracking Tables

See the following table for important details.

Table 5-2 Storage Characteristics of Statistics Tables

Table Type	Interval (Resolution)	Storage Lifespan
current_day	15 seconds for **app_ids_stats_current_***timeframe* and **user_ids_stats_current_***timeframe*	current interval plus all intervals in the preceding 24 hours
	5 minutes for **app_stats_current_***timeframe*, **user_stats_current_***timeframe*, **url_category_stats_current_***timeframe*, and **url_reputation_stats_current_***timeframe*	current interval plus all intervals in the preceding 24 hours
current_month	one hour	current hour plus the hours stretching back 30 days
current_year	24 hours	current day plus the preceding 360 days

A storage interval is defined by its start time. For example, the **current_month** table contains counts for the hour 10:00:00 - 10:59:59 as one record with a timestamp of 10:00:00. Note that a day begins at 00:00:00 and ends at 23:59:59. Interval start times are stored as UNIX timestamps (GMT).

Specifying Time Intervals When Querying Statistics Tables

The effective time interval for a query is defined by both the table and the `time_start_sec` field in the query.

For example, if your SQL statement specifies `time_start_sec` = `6:00:00`, the interval varies for each table type:

- for **`current_day`** tables: either `6:00:00` to `6:00:14` (for 15 second tables) or `6:00:00` to `6:04:59` (for 5 minute tables).
- for **`current_month`** tables: `6:00:00` to `6:59:59`.
- for **`current_year`** tables: `0:00:00` to `23:59:59` on the following day.

The simplest way to retrieve data is to state the interval start time. For example, to retrieve from the **`app_ids_stats_current_day`** table, specify one of the following:

```
00:00:00
00:00:15
00:00:30
23:59:45
```

If your query contains a timestamp that is other than an interval start time, the system modifies the request as follows:

- rounds up the start time to the nearest interval time
- rounds down the end time to the nearest interval time

For example, the following query rounds up the start time:

```
SELECT application_id
FROM app_ids_stats_current_month
WHERE start_time_sec = UNIX_TIMESTAMP("2011-12-01 12:30:00");
```

and is the same as:

```
SELECT application_id
FROM app_ids_stats_current_month
WHERE start_time_sec = UNIX_TIMESTAMP("2011-12-01 01:00:00");
```

When querying a range of intervals, the starting time interval is rounded up, and the ending time interval is rounded down. For example:

```
SELECT application_id
FROM app_ids_stats_current_month
WHERE start_time_sec BETWEEN UNIX_TIMESTAMP("2011-12-10 12:59:00") and
UNIX_TIMESTAMP("2011-12-10 16:28:00");
```

is changed to:

```
SELECT application_id
FROM app_ids_stats_current_month
WHERE start_time_sec BETWEEN UNIX_TIMESTAMP("2011-12-10 13:00:00") and
UNIX_TIMESTAMP("2011-12-12 16:00:00");
```

If your query interval extends beyond a table's time frame, you can usually obtain the additional data from another table, although the data in the other table will have a coarser resolution. For example, to retrieve bandwidth usage for the past two days, you can get results for yesterday from the **`current_day`** table (at 5 minute resolution), but you can get statistics for the previous day only from **`current_month`** (in hour chunks) or **`current_year`** (in day chunks).

app_ids_stats_current_timeframe

The **app_ids_stats_current_***timeframe* tables contain statistics about application activity and intrusion events on your monitored network. Statistics can be extracted per detected application, per application type (application protocol, client application, or web application), and also per risk and business relevance of the application. The tables also track blocked connections due to intrusion policy violations and the estimated potential impact of an intrusion.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **app_ids_stats_current_***timeframe* tables, see the following sections:

app_ids_stats_current_timeframe Fields

The following table describes the fields you can access in the **app_ids_stats_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-3 app_ids_stats_current_timeframe Fields

Field	Description
application_id	The internal identification number for the application.
application_name	The application name that appears in the user interface.
blocked	Number of connections blocked due to violation of an intrusion policy.
business_relevance	An index (from 1 to 5) of the application's relevance to business productivity where 1 is very low and 5 is very high.
business_relevance_description	A description of business relevance (very low, low, medium, high, very high).
impact_level_1	The number of impact level 1 (vulnerable) intrusion events recorded for the application.
impact_level_2	The number of impact level 2 (potentially vulnerable) intrusion events.
impact_level_3	The number of impact level 3 (host currently not vulnerable) intrusion events.
impact_level_4	The number of impact level 4 (unknown target) intrusion events.
impact_level_5	The number of impact level 5 (unknown vulnerability) intrusion events.
is_client_application	A true-false flag that indicates if the detected application is a client application.
is_server_application	A true-false flag that indicates if the detected application is an application protocol.
is_web_application	A true-false flag that indicates if the detected application is a web application.
risk	An index (from 1 to 5) of the application's estimated risk where 1 is very low risk and 5 is critical risk.

Table 5-3 app_ids_stats_current_timeframe Fields (continued)

Field	Description
risk_description	A description of the estimated risk (very low, low, medium, high, critical).
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_id	ID of the device that provided the event.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.
would_have_dropped	Number of packets that would have been dropped if the intrusion policy had been configured to drop packets in an inline deployment.

app_ids_stats_current_timeframe Joins

The following table describes the joins you can perform on the **app_ids_stats_current_***timeframe* tables.

Table 5-4 app_ids_stats_current_timeframe Joins

You can join this table on...	And...
application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

app_ids_stats_current_timeframe Sample Query

The following query returns up to 25 application records from the **app_ids_stats_current_month** table. Each record contains the number of blocked connections and intrusion events for the application over the time interval.

```
SELECT from_unixtime(start_time_sec), sum(blocked)

FROM app_ids_stats_current_day

WHERE start_time_sec = unix_timestamp("2013-12-15");
```

app_stats_current_timeframe

The **app_stats_current**_*timeframe* tables contain statistics on bandwidth usage and access control actions (connection allowed or denied), by application and by device that monitored the traffic. You can filter these statistics by the business relevance, estimated risk, and type of the application.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **app_stats_current**_*timeframe* tables, see the following sections:

app_stats_current_timeframe Fields

The following table describes the fields you can access in the **app_stats_current**_*timeframe* tables.

Table 5-5 app_stats_current_timeframe Fields

Field	Description
application_id	The internal identification number for the application.
application_name	The application name that appears in the user interface.
business_relevance	An index (from 1 to 5) of the application's relevance to business productivity where 1 is very low and 5 is very high.
business_relevance_description	A description of business relevance (very low, low, medium, high, very high).
bytes_in	The bytes of inbound traffic for the application during the specified interval.
bytes_out	The bytes of outbound traffic for the application during the specified interval.
connections_allowed	The number of connections allowed.
connections_denied	The number of connections denied due to violation of an access control policy.
is_client_application	A true-false flag that indicates if the detected application is a client application.
is_server_application	A true-false flag that indicates if the detected application is an application protocol.
is_web_application	A true-false flag that indicates if the detected application is a web application.
risk	An index (from 1 to 5) of the application's estimated risk where 1 is very low risk and 5 is critical risk.
risk_description	A description of the estimated risk (very low, low, medium, high, critical).
sensor_address	The IP address of the managed device that monitored the traffic. Format is *ipv4_address,ipv6_address*.
sensor_id	The internal identification number of the managed device that detected the traffic.

Table 5-5 app_stats_current_timeframe Fields (continued)

Field	Description
sensor_name	The name of the managed device that detected the traffic.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the start of the measurement interval. For information on specifying the start time, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

app_stats_current_timeframe Joins

The following table describes the joins you can perform on the **app_stats_current_***timeframe* tables.

Table 5-6 app_stats_current_timeframe Joins

You can join this table on...	And...
application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

app_stats_current_timeframe Sample Query

The following query returns the inbound and outbound traffic load associated with applications that have low business relevance and high risk in the period of a day, for all managed devices connected to the Defense Center.

```
SELECT start_time_sec, sum(bytes_in), sum(bytes_out)

FROM app_stats_current_day

WHERE business_relevance <= 2

AND risk >= 4 AND start_time_sec = unix_timestamp("2013-12-15");
```

geolocation_stats_current_timeframe

The **geolocation_stats_***timeframe* tables contain statistics regarding intrusion events based on location levels. Statistics can be extracted based on impact level, device, and how the packets are handled.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **geolocation_stats_current_***timeframe* tables, see the following sections:

- geolocation_stats_current_timeframe Fields, page 5-8

geolocation_stats_current_timeframe Fields

The following table describes the fields you can access in the **geolocation_stats_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-7 geolocation_stats_current_timeframe Fields

Field	Description
bytes_from	The total number of bytes transmitted by the session responder.
bytes_to	Total number of bytes transmitted by the session initiator.
destination_continent	The name of the continent of the destination host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
destination_country	Code for the country of the destination host.
flows_allowed	The number of flows allowed.
flows_denied	The number of flows denied due to violation of an access control policy.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address*,*ipv6_address*.
sensor_id	ID of the device that provided the event.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
source_continent	The name of the continent of the source host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica

Table 5-7 geolocation_stats_current_timeframe Fields (continued)

Field	Description
source_country	Code for the country of the source host.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

geolocation_stats_current_timeframe Joins

You cannot perform joins on the **geolocation_stats_current_***timeframe* tables.

geolocation_stats_current_timeframe Sample Query

The following query returns source country and sensor name for the first 25 connection events from Asia during the current day.

```
SELECT sensor_name, source_continent
FROM geolocation_stats_current_year
WHERE destination_continent='as'
LIMIT 20;
```

ids_impact_stats_current_timeframe

The **ids_impact_stats_***timeframe* tables contain statistics regarding intrusion events based on impact levels. Statistics can be extracted based on impact level, device, and how the packets are handled.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **ids_impact_stats_current_***timeframe* tables, see the following sections:

ids_impact_stats_current_timeframe Fields

The following table describes the fields you can access in the **ids_impact_stats_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-8 ids_impact_stats_current_timeframe Fields

Field	Description
blocked	Number of connections blocked due to violation of an intrusion policy.
impact_level_1	The number of impact level 1 (vulnerable) intrusion events recorded for the application.
impact_level_2	The number of impact level 2 (potentially vulnerable) intrusion events.
impact_level_3	The number of impact level 3 (host currently not vulnerable) intrusion events.
impact_level_4	The number of impact level 4 (unknown target) intrusion events.
impact_level_5	The number of impact level 5 (unknown vulnerability) intrusion events.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_id	ID of the device that provided the event.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.
would_have_dropped	Number of packets that would have been dropped if the intrusion policy had been set to drop packets in an inline deployment.

ids_impact_stats_current_timeframe Joins

You cannot perform joins on the **ids_impact_stats_current_***timeframe* tables.

ids_impact_stats_current_timeframe Sample Query

The following query returns the first 25 blocked and would_have_dropped events during the current day.

```
SELECT blocked, would_have_dropped
FROM ids_impact_stats_current_year
LIMIT 25;
```

session_stats_current_timeframe

The **session_stats_***timeframe* tables contain statistics for all connections. Statistics can be extracted based on bytes, connection, sensor, and time.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **session_stats_current_***timeframe* tables, see the following sections:

- session_stats_current_timeframe Fields, page 5-11

session_stats_current_timeframe Fields

The following table describes the fields you can access in the **session_stats_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-9 session_stats_current_timeframe Fields

Field	Description
bytes_in	The bytes of inbound traffic during the specified interval.
bytes_out	The bytes of outbound traffic during the specified interval.
connections_allowed	The number of connections allowed for the specified URL category.
connections_denied	The number of connections denied for the specified URL category due to violation of an access control policy.
id	This field is not used and will always return 0.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_id	ID of the device that provided the event.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

session_stats_current_timeframe Joins

You cannot perform joins on the **session_stats_current_***timeframe* tables.

session_stats_current_timeframe Sample Query

The following query returns the number of denied and allowed connections for each sensor, in descending order by sensor_name during the current day.

```
SELECT sensor_name, connections_denied, connections_allowed
FROM session_stats_current_day
ORDER BY sensor_id DESC;
```

ssl_stats_current_timeframe

The **ssl_stats_current_***timeframe* tables contain statistics for SSL connections. Statistics can be extracted based on bytes, connection, sensor, and time.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **ssl_stats_current_***timeframe* tables, see the following sections:

- ssl_stats_current_timeframe Fields, page 5-12
- ssl_stats_current_timeframe Joins, page 5-13
- ssl_stats_current_timeframe Sample Query, page 5-14

ssl_stats_current_timeframe Fields

The following table describes the fields you can access in the **ssl_stats_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-10 ssl_stats_current_timeframe Fields

Field	Description
block	Number of SSL sessions dropped with no reset.
block_with_reset	Number of SSL sessions dropped with reset.
cached_session	Number of SSL sessions found in the session cache.
cannot_determine_verdict	Number of handshake errors that occurred while evaluating SSL rules.
cert_expired	Number of SSL sessions in which the certificate was expired.
cert_invalid_issuer	Number of SSL sessions in which the certificate issuer was either not valid or not found in the Trusted CA list.
cert_invalid_signature	Number of SSL sessions in which the certificate had an invalid signature.
cert_not_checked	Number of SSL sessions in which the certificate was not checked.
cert_not_yet_valid	Number of SSL sessions in which the certificate was not yet valid.
cert_revoked	Number of SSL sessions in which the certificate had been revoked.
cert_self_signed	Number of SSL sessions in which the certificate was self-signed.
cert_unknown	Number of SSL sessions in which the certificate status was unknown.
cert_valid	Number of SSL sessions in which the certificate was valid.
cert_validation_cache_hit	Number of times a certificate was found in the validation cache.
cert_validation_cache_miss	Number of times a certificate was not found in the validation cache.
decrypt_resign_self_signed	Number of times an SSL session using a self-signed certificate was decrypted using the decrypt-resign method.
decrypt_resign_self_signed_replace_key_only	Number of times an SSL session using a self-signed certificate was decrypted using the decrypt-resign with replace key only method.
decrypt_resign_signed_cert	Number of times an SSL session using a signed certificate was decrypted using the decrypt-resign method.
decrypt_with_known_key	Number of times an SSL session was decrypted using the known-key method.
decryption_error	Number of SSL sessions which suffered an error during decryption.
do_not_decrypt	Number of times an SSL session was found but not decrypted.
handshake_error	Number of handshake errors that occurred prior to evaluating SSL rules.

Table 5-10 ssl_stats_current_timeframe Fields (continued)

Field	Description
orig_cert_cache_hit	Number of times an original certificate was found in the cache.
orig_cert_cache_miss	Number of times an original certificate was not found in the cache.
resigned_cert_cache_hit	Number of times a resigned certificate was found in the cache.
resigned_cert_cache_miss	Number of times a resigned certificate was not found in the cache.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_id	ID of the device that provided the event.
sensor_name	The name of the managed device that generated the event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
session_cache_hit	Number of times an SSL session ID or ticket was found in the cache.
session_cache_miss	Number of times an SSL session ID or ticket was not found in the cache.
session_incorrectly_identified_as_ssl	Number of sessions that were incorrectly identified as using SSL.
ssl_compression	Number of sessions that used SSL compression.
ssl_sessions_decrypted	Number of SSL sessions that were successfully decrypted.
ssl_sessions_not_decrypted	Number of SSL sessions that were not successfully decrypted.
ssl_sessions_reused_by_id	Number of times an SSL session reused an ID.
ssl_sessions_reused_by_ticket	Number of times an SSL session reused a ticket.
ssl_sessions_with_errors	Number of SSL sessions which have errors.
ssl_v20	Number of SSL sessions using SSL version 2.0
ssl_v30	Number of SSL sessions using SSL version 3.0
ssl_version_unknown	Number of SSL sessions using an unknown SSL version.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.
tls_v10	Number of SSL sessions using TLS version 1.0
tls_v11	Number of SSL sessions using TLS version 1.1
tls_v12	Number of SSL sessions using TLS version 1.2
total_ssl_sessions	Total number of SSL sessions detected.
uncached_session	Number of times that a cache miss on an ID or ticket prevented decryption.
undecryptable_in_passive_mode	Number of SSL sessions that could not be decrypted because the device is in passive mode.
unknown_cipher_suite	Number of SSL sessions using an unknown cipher suite.
unsupported_cipher_suite	Number of SSL sessions using a cipher suite which is known but not supported.

ssl_stats_current_timeframe Joins

You cannot perform joins on the **ssl_stats_current_***timeframe* tables.

ssl_stats_current_timeframe Sample Query

The following query returns the number of SSL sessions, sessions that were decrypted, sessions that were not decrypted, and sessions which cannot be decrypted in passive mode for each sensor, in descending order by `sensor_name` during the current day.

```
SELECT sensor_name, total_ssl_sessions, ssl_sessions_decrypted,
ssl_sessions_not_decrypted, undecryptable_in_passive_mode

FROM ssl_stats_current_day

ORDER BY sensor_id DESC;
```

storage_stats_by_disposition_current_timeframe

The **storage_stats_by_disposition**_*timeframe* tables contain statistics for stores files. Statistics can be extracted based on bytes, connection, sensor, and time.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **storage_stats_by_disposition**_*timeframe* tables, see the following sections:

- storage_stats_by_disposition_current_timeframe Fields, page 5-14
- storage_stats_by_disposition_current_timeframe Joins, page 5-15
- storage_stats_by_disposition_current_timeframe Sample Query, page 5-15

storage_stats_by_disposition_current_timeframe Fields

The following table describes the fields you can access in the **storage_stats_by_disposition_current**_*timeframe* tables. All tables of this type contain the same fields.

Table 5-11 storage_stats_by_disposition_current_timeframe Fields

Field	Description
bytes_written	The size of the file, in bytes.
disposition	The malware status of the file. Possible values include: • CLEAN — The file is clean and does not contain malware. • UNKNOWN — It is unknown whether the file contains malware. • MALWARE — The file contains malware. • UNAVAILABLE — The software was unable to send a request to the Cisco cloud for a disposition, or the Cisco cloud services did not respond to the request. • CUSTOM SIGNATURE — The file matches a user-defined hash, and is treated in a fashion designated by the user.
number_dropped	Number of files of this disposition dropped.
number_stored	Number of files of this disposition stored.

Table 5-11 storage_stats_by_disposition_current_timeframe Fields (continued)

Field	Description
sensor	ID of the device that detected the file.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

storage_stats_by_disposition_current_timeframe Joins

You cannot perform joins on the **session_stats_current**_*timeframe* tables.

storage_stats_by_disposition_current_timeframe Sample Query

The following query returns the number of dropped and stored files for each sensor, in descending order by sensor_name during the current day.

```
SELECT sensor_name, number_dropped, number_stored
FROM storage_stats_by_disposition_current_day
ORDER BY sensor_name DESC;
```

storage_stats_by_file_type_current_timeframe

The **storage_stats_by_file_type_current**_*timeframe* tables contain statistics for stored files by file type. Statistics can be extracted based on bytes, connection, sensor, and time.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **storage_stats_by_file_type_current**_*timeframe* tables, see the following sections:

storage_stats_by_file_type_current_timeframe Fields

The following table describes the fields you can access in the **storage_stats_by_file_type_current**_*timeframe* tables. All tables of this type contain the same fields.

Table 5-12 storage_stats_by_file_type_current_timeframe Fields

Field	Description
bytes_written	The size of the file, in bytes.
file_type	The file type of the detected or quarantined file.
file_type_id	ID number that maps to the file type.
number_dropped	Number of files of this type dropped.
number_stored	Number of files of this type stored.
sensor	ID of the device that detected the file.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

storage_stats_by_file_type_current_timeframe Joins

You cannot perform joins on the **session_stats_current_***timeframe* tables.

storage_stats_by_file_type_current_timeframe Sample Query

The following query returns the number of dropped and stored files for each sensor, in descending order by file_type during the current day.

```
SELECT sensor_name, number_dropped, number_stored, file_type

FROM storage_stats_by_file_type_current_day

ORDER BY file_type DESC;
```

transmission_stats_by_file_type_current_timeframe

The **transmission_stats_by_file_type_current_***timeframe* tables contain statistics for stored files by file type. Statistics can be extracted based on bytes, connection, sensor, and time.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **transmission_stats_by_file_type_current_***timeframe* tables, see the following sections:

transmission_stats_by_file_type_current_timeframe Fields

The following table describes the fields you can access in the **storage_stats_by_file_type_current_***timeframe* tables. All tables of this type contain the same fields.

Table 5-13 transmission_stats_by_file_type_current_timeframe Fields

Field	Description
bytes_sent	The number of transmitted bytes.
file_type	The file type of the detected or quarantined file.
file_type_id	ID number that maps to the file type.
number_dropped	Number of files of this type dropped.
number_sent	Number of files of this type sent.
sensor	ID of the device that detected the file.
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address*,*ipv6_address*.
sensor_name	The name of the managed device that generated the intrusion event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the date and time the measurement interval starts. For detailed information, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

transmission_stats_by_file_type_current_timeframe Joins

You cannot perform joins on the **transmission_stats_current_***timeframe* tables.

transmission_stats_by_file_type_current_timeframe Sample Query

The following query returns the number of dropped and sent connections for each sensor, in descending order by file_type during the current day.

```
SELECT sensor_name, number_dropped, number_sent, file_type
FROM transmission_stats_by_file_type_current_day
ORDER BY file_type DESC;
```

url_category_stats_current_timeframe

The **url_category_stats_current_***timeframe* tables contain statistics on the bandwidth usage and connections associated with requests to URLs in specified URL categories. You can also constrain queries on the managed device that monitored the traffic.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **url_category_stats_current_***timeframe* tables, see the following sections:

url_category_stats_current_timeframe Fields

The following table describes the fields you can access in the **url_category_stats_current_***timeframe* tables.

Table 5-14 url_category_stats_current_timeframe Fields

Field	Description
bytes_in	The bytes of inbound traffic during the specified interval.
bytes_out	The bytes of outbound traffic during the specified interval.
category	The category of the URL.
connections_allowed	The number of connections allowed for the specified URL category.
connections_denied	The number of connections denied for the specified URL category due to violation of an access control policy.
sensor_address	The IP address of the managed device that monitored the traffic. Format is *ipv4_address*,*ipv6_address*.
sensor_id	The internal identification number of the managed device that detected the traffic.
sensor_name	The managed device that monitored the traffic.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the start of the measurement interval. For information on specifying the start time, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

url_category_stats_current_timeframe Joins

You cannot perform joins on the **url_category_stats_current_***timeframe* tables.

url_category_stats_current_timeframe Sample Query

The following query returns up to 25 URL category records. Each record contains the bytes of associated inbound and outbound traffic, as well as allowed and denied connections, over the specified time interval.

```
SELECT category, sensor_name, sensor_address, start_time_sec, bytes_in, bytes_out,
connections_allowed, connections_denied

FROM url_category_stats_current_year

WHERE category="Games"

LIMIT 0, 25;
```

url_reputation_stats_current_timeframe

The **url_reputation_stats_current_***timeframe* tables contain statistics on the bandwidth usage and connections associated with requests to URLs with specified reputations. Query results can also be constrained on the managed device that monitored the traffic.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information on the **url_reputation_stats_current_***timeframe* tables, see the following sections:

url_reputation_stats_current_timeframe Fields

The following table describes the fields you can access in the **url_category_stats_current_***timeframe* tables.

Table 5-15 url_reputation_stats_current_timeframe Fields

Field	Description
bytes_in	The bytes of inbound traffic during the specified interval.
bytes_out	The bytes of outbound traffic during the specified interval.
connections_allowed	The number of connections allowed.
connections_denied	The number of connections denied due to violation of an access control policy.
reputation	The risk associated with the requested URL. One of the following: • High risk • Suspicious site • Benign site with security risks • Benign site • Well known • Risk unknown
sensor_address	The IP address of the managed device that monitored the traffic. Format is *ipv4_address*,*ipv6_address*.
sensor_id	Internal identification number of the managed device that monitored the traffic.
sensor_name	The name of the managed device that monitored the traffic.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the start of the measurement interval. For information on specifying the start time, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.

url_reputation_stats_current_timeframe Joins

You cannot perform joins on the **url_reputation_stats_current_***timeframe* tables.

url_reputation_stats_current_timeframe Sample Query

The following query returns up to 25 URL reputation records from the **url_reputation_stats_current_month** table. Each record contains the bytes of inbound and outbound traffic, as well as allowed and denied connections over the measurement time interval.

```
SELECT sensor_name, sensor_address, reputation, start_time_sec, bytes_in, bytes_out,
connections_allowed, connections_denied

FROM url_reputation_stats_current_year

WHERE reputation="High risk"

LIMIT 0, 25;
```

user_ids_stats_current_timeframe

The **user_ids_stats_current_***timeframe* tables are round-robin tables that contain statistics on access filtering and impact statistics by user.

For an understanding of the **current_day**, **current_month**, and **current_year** tables in this type, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For general information on using the round robin statistics tables, see Understanding Statistics Tracking Tables, page 5-2.

For more information on the **user_ids_stats_current_***timeframe* tables, see the following sections:

user_ids_stats_current_timeframe Fields

The following table describes the fields you can access in the **user_ids_stats_current_***timeframe* tables.

Table 5-16 user_ids_stats_current_timeframe Fields

Field	Description
blocked	The number of connections blocked due to violation of an intrusion policy.
impact_level_1	The number of impact level 1 (vulnerable) intrusion events recorded for the user.
impact_level_2	The number of impact level 2 (potentially vulnerable) intrusion events recorded for the user.
impact_level_3	The number of impact level 3 (host currently not vulnerable) intrusion events recorded for the user.

Table 5-16 user_ids_stats_current_timeframe Fields (continued)

Field	Description
impact_level_4	The number of impact level 4 (unknown target) intrusion events recorded for the user.
impact_level_5	The number of impact level 5 (unknown vulnerability) intrusion events recorded for the user.
sensor_address	The IP address of the managed device that monitored the traffic. Format is *ipv4_address,ipv6_address*.
sensor_id	The internal identification number of the managed device that detected the traffic.
sensor_name	The name of the managed device that detected the traffic.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the start of the measurement interval. For information on specifying the start time, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.
user_id	An internal identification number for the user who last logged into the host.
username	The user name of the user who last logged into the host.
would_have_dropped	Number of packets that would have been dropped if the intrusion policy had been configured to drop packets in an inline deployment.

user_ids_stats_current_timeframe Joins

You cannot perform joins on the **user_ids_stats_current**_*timeframe* tables.

user_ids_stats_current_timeframe Sample Query

The following query returns up to 25 user records from the **user_ids_stats_current_month** table. Each record contains the number of blocked connections and intrusion events for the selected username.

```
SELECT username, start_time_sec, blocked, impact_level_1, impact_level_2,
impact_level_3, impact_level_4, impact_level_5 FROM user_ids_stats_current_year

WHERE username="username"

LIMIT 0, 25;
```

user_stats_current_timeframe

The **user_stats_current**_*timeframe* tables contain statistics on bandwidth usage and access control actions (connection allowed or denied) by user. You can also constrain queries on the managed device that monitored the traffic.

For an understanding of the **current_day**, **current_month**, and **current_year** statistics tables, see Storage Characteristics for Statistics Tracking Tables, page 5-2.

For more information, see the following sections:

- user_stats_current_timeframe Fields, page 5-22

user_stats_current_timeframe Fields

The following table describes the fields you can access in the **user_stats_current_***timeframe* tables.

Table 5-17 user_stats_current_timeframe Fields

Field	Description
bytes_in	The number of bytes of inbound traffic for the user in the measured interval.
bytes_out	The number of bytes of outbound traffic for the user in the measured interval.
connections_allowed	The number of connections allowed for this user in the measured time frame.
connections_denied	The number of connections denied for this user due to violation of an access control policy.
sensor_address	The IP address of the managed device that monitored the traffic. Format is *ipv4_address,ipv6_address*.
sensor_id	The internal identification number of the managed device that detected the traffic.
sensor_name	The name of the managed device that detected the traffic.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
start_time_sec	The UNIX timestamp of the start of the measurement interval. For information on specifying the start time, see Specifying Time Intervals When Querying Statistics Tables, page 5-3.
user_id	The internal identification number for the user who last logged into the host that generated the traffic.
username	User name for the user who last logged into the host that generated the traffic.

user_stats_current_timeframe Joins

You cannot perform joins on the **user_stats_current_***timeframe* tables.

user_stats_current_timeframe Sample Query

The following query returns up to 25 user records. Each record contains the bytes of inbound and outbound traffic, as well as allowed and denied connections over the measurement time interval.

```
SELECT sensor_name, sensor_address, username, start_time_sec, bytes_in, bytes_out,
connections_allowed, connections_denied
FROM user_stats_current_year
WHERE username="username" LIMIT 0, 25;
```

Schema: Discovery Event and Network Map Tables

This chapter contains information on the schema and supported joins for tables related to discovery events and the Cisco network map.

Your FireSIGHT System generates discovery events continuously as it monitors the traffic produced by your hosts and network devices.

The network map is a repository of information about the network assets reported in discovery events. For each detected host and network device, the network map contains information such as operating system, servers, client applications, host attributes, vulnerabilities, and so on.

Vulnerabilities are descriptions of specific compromises or exploits to which hosts may be susceptible. Cisco maintains its own vulnerability database (VDB), which cross-references the Bugtraq database and MITRE's CVE database. You can also import third-party vulnerability data using the host input feature.

Note that the information about a given host in the network map can vary according to the type of host and the information available in the monitored traffic.

For more information, see the sections listed in the following table. The Version column indicates the FireSIGHT System versions that support each table. While support for deprecated tables continues in the current product release, Cisco **strongly** recommends avoiding the use of deprecated tables and fields, to ensure continued support in the future.

Table 6-1 Schema for Discovery Event and Network Map Tables

See...	For the table that stores information on...	Version
application_host_map, page 6-5	Applications detected on the hosts in your monitored network.	5.0+
application_ip_map, page A-1	The category, tags, productivity, and risk associated with an application detected in your monitored network.	5.2+
application_ip_map, page A-1	The category, tags, productivity, and risk associated with an application detected in your monitored network. deprecated in Version 5.2. Superseded by **application_ip_map, page A-1**.	5.0-5.1.x
application_tag_map, page 6-9	The tags associated with an application detected in your monitored network.	5.0+
network_discovery_event, page 6-11	Discovery and host input events.	5.0+
rna_host, page 6-12	Basic information on the hosts in your monitored network.	5.2+

Table 6-1 Schema for Discovery Event and Network Map Tables (continued)

See...	For the table that stores information on...	Version
rna_host_attribute, page 6-14	The host attributes associated with each host in your monitored network.	5.2+
rna_host_client_app, page 6-15	The client applications detected on the hosts in your monitored network.	5.2+
rna_host_client_app, page 6-15	The payloads associated with HTTP (web browser) client applications detected on the hosts in your monitored network.	5.2+
rna_host_ioc_state, page 6-21	Stores compromise state for hosts.	5.3+
rna_host_ip_map, page 6-25	Correlates host IDs to MAC addresses for hosts in your monitored network.	5.2+
rna_host_os, page 6-28	The operating systems detected on the hosts in your monitored network.	5.2+
rna_host_os_vulns, page 6-29	The vulnerabilities associated with the hosts in your monitored network.	5.2+
rna_host_protocol, page 6-31	The protocols detected on the hosts in your monitored network.	4.10.x+
rna_host_protocol, page 6-31	The hosts in your monitored network with regard to the managed device that detected them.	5.2+
rna_host_service, page 6-34	The services detected on the hosts in your monitored network.	5.2+
rna_host_service_banner, page 6-36	Headers from network traffic that advertise service vendors and versions (“banners”) for the services detected on hosts in your monitored network.	5.2+
rna_host_service_info, page 6-37	Details of the services detected on the hosts in your monitored network.	5.2+
rna_host_service_payload, page 6-42	The payloads associated with services detected on the hosts in your monitored network.	5.2+
rna_host_service_subtype, page 6-45	The sub-services for the services detected on the hosts in your monitored network.	5.2+
rna_host_service_vulns, page 6-46	The vulnerabilities associated with the services detected on the hosts in your monitored network.	5.2+
rna_host_third_party_vuln, page 6-47	The third-party vulnerabilities associated with the hosts in your monitored network.	5.2+
rna_host_third_party_vuln_bugtraq_id, page 6-49	The third-party vulnerabilities associated with the hosts in your monitored network that are also associated with a vulnerability in the Bugtraq database (`http://www.securityfocus.com/bid/`).	5.2+
rna_host_third_party_vuln_cve_id, page 6-50	The third-party vulnerabilities associated with the hosts in your monitored network that are also associated with a vulnerability in MITRE’s CVE database. (`http://www.cve.mitre.org/`).	5.2+

Table 6-1 Schema for Discovery Event and Network Map Tables (continued)

See...	For the table that stores information on...	Version
rna_host_third_party_vuln_rna_id, page 6-52	The third-party vulnerabilities associated with the hosts in your monitored network that are also associated with a vulnerability in the VDB.	5.2+
rna_ip_host, page A-1	Basic information on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host, page 6-12**.	4.10.x-5.1.x
rna_ip_host_client_app, page A-1	The client applications detected on the hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_client_app, page 6-15**.	4.10.x-5.1.x
rna_ip_host_client_app_payload, page A-1	The payloads associated with HTTP (web browser) client applications detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_client_app, page 6-15**.	4.10.x-5.1.x
rna_ip_host_os, page A-1	The operating systems detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_os, page 6-28**.	4.10.x-5.1.x
rna_ip_host_os_vulns, page A-1	The vulnerabilities associated with the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_os_vulns, page 6-29**.	4.10.x--5.1.x
rna_ip_host_sensor, page A-1	The IP hosts in your monitored network with regard to the managed device that detected them. deprecated in Version 5.2. Superseded by **rna_host_protocol, page 6-31**.	5.0-5.1.x
rna_ip_host_service, page A-1	The services detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service, page 6-34**.	4.10.x-5.1.x
rna_ip_host_service_banner, page A-1	Headers from network traffic that advertise service vendors and versions (“banners”) for the services detected on hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service_banner, page 6-36**.	4.10.x-5.1.x
rna_ip_host_service_info, page A-1	Details of the services detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service_info, page 6-37**.	4.10.x-5.1.x

Table 6-1 Schema for Discovery Event and Network Map Tables (continued)

See...	For the table that stores information on...	Version
rna_ip_host_service_payload, page A-1	The payloads associated with services detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service_payload, page 6-42**.	4.10.x-5.1.x
rna_ip_host_service_subtype, page A-1	The sub-services for the services detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service_subtype, page 6-45**.	4.10.x-5.1.x
rna_ip_host_service_vulns, page A-1	The vulnerabilities associated with the services detected on the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_service_vulns, page 6-46**.	4.10.x-5.1.x
rna_ip_host_third_party_vuln, page A-1	The third-party vulnerabilities associated with the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_third_party_vuln, page 6-47**.	4.10.x-5.1.x
rna_ip_host_third_party_vuln_bugtraq_id, page A-1	The third-party vulnerabilities associated with the IP hosts in your monitored network that are also associated with a vulnerability in the Bugtraq database (`http://www.securityfocus.com/bid/`). deprecated in Version 5.2. Superseded by **rna_host_third_party_vuln_bugtraq_id, page 6-49**.	4.10.x-5.1.x
rna_ip_host_third_party_vuln_cve_id, page A-1	The third-party vulnerabilities associated with the IP hosts in your monitored network that are also associated with a vulnerability in MITRE's CVE database. (`http://www.cve.mitre.org/`). deprecated in Version 5.2. Superseded by **rna_host_third_party_vuln_cve_id, page 6-50**.	4.10.x-5.1.x
rna_ip_host_third_party_vuln_rna_id, page A-1	The third-party vulnerabilities associated with the IP hosts in your monitored network that are also associated with a vulnerability in the VDB. deprecated in Version 5.2. Superseded by **rna_host_third_party_vuln_rna_id, page 6-52**.	4.10.x-5.1.x
rna_ip_host_user_history, page A-1	User activity for a particular IP host in your monitored network. deprecated in Version 5.2. Superseded by **user_ipaddr_history, page 6-59**.	4.10.x-5.1.x
rna_mac_host, page A-1	The MAC hosts (hosts without an IP address) in your monitored network.	4.10.x-5.1.x
rna_mac_host_sensor, page A-2	The IP hosts in your monitored network with regard to the managed devices that detected them.	5.0-5.1.x

Table 6-1 Schema for Discovery Event and Network Map Tables (continued)

See...	For the table that stores information on...	Version
rna_mac_ip_map, page A-2	The MAC addresses of the IP hosts in your monitored network. deprecated in Version 5.2. Superseded by **rna_host_ip_map, page 6-25** and **rna_host_mac_map, page 6-27**.	4.10.x-5.1.x
rna_vuln, page 6-54	The vulnerabilities in the Cisco VDB.	4.10.x+
tag_info, page 6-57	The tags that characterize detected applications.	5.0+
url_categories, page 6-58	The categories that characterize URLs accessed from hosts in your monitored network.	5.0+
url_reputations, page 6-58	The reputations that characterize URLs accessed from hosts in your monitored network.	5.0+
user_ipaddr_history, page 6-59	User activity for a particular host in your monitored network.	5.2+

application_host_map

The **application_host_map** table contains information on the categories and tags associated with each application detected on your network.

For more information, see the following sections:

- application_host_map Fields, page 6-5
- application_host_map Joins, page 6-6
- application_host_map Sample Query, page 6-7

application_host_map Fields

The following table describes the fields you can access in the **application_host_map** table.

Table 6-2 application_host_map Fields

Field	Description
application_id	The internal identification number for the application.
application_name	The application name that appears in the user interface.
application_tag_id	This field has been deprecated and will now return null.
business_relevance	The index (from 1 to 5) of the application's relevance to business productivity, where 1 is very low and 5 is very high.
business_relevance_description	The description of the business relevance (very low, low, medium, high, very high).
host_id	ID number of the host.
risk	An index (from 1 to 5) of the application's risk, where 1 is very low risk and 5 is critical risk.
risk_description	The description of the risk (very low, low, medium, high, critical).

application_host_map Joins

The following table describes the joins you can perform on the **application_host_map** table.

Table 6-3 application_host_map Joins

You can join this table on...	And...
application_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

application_host_map Sample Query

The following query returns information about the applications detected on the host with a `host_id` of `8`.

```
SELECT host_id, application_id, application_name, business_relevance, risk
FROM application_host_map
```

```
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

application_info

The **application_info** table contains information about the applications that can be detected on the hosts in your monitored network.

You can retrieve the list of tags associated with an application from the **application_tag_map** table by joining on **application_id**. Similarly, you can retrieve an application's list of associated categories from the **application_host_map** by joining on **application_id**.

For more information, see the following sections:

- application_info Fields, page 6-8
- application_info Joins, page 6-8
- application_info Sample Query, page 6-9

application_info Fields

The following table describes the fields you can access in the **application_info** table.

Table 6-4 application_info Fields

Field	Description
application_description	A description of the application.
application_id	The internal identification number for the application.
application_name	The application name that appears in the user interface.
business_relevance	An index (from 1 to 5) of the application's relevance to business productivity, where 1 is very low and 5 is very high.
business_relevance_description	A description of business relevance (very low, low, medium, high, very high).
is_client_application	A true-false flag that indicates if the detected application is a client.
is_server_application	A true-false flag that indicates if the detected application is a server application.
is_web_application	A true-false flag that indicates if the detected application is a web application.
risk	An index (from 1 to 5) of the application's estimated risk where 1 is very low risk and 5 is critical risk.
risk_description	A description of the risk (very low, low, medium, high, and critical).

application_info Joins

The following table describes the joins you can perform on the **application_info** table.

Table 6-5 application_info Joins

You can join this table on...	And...
application_id	application_host_map.application_id app_ids_stats_current_timeframe.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id si_connection_log.application_protocol_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id connection_summary.application_protocol_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

application_info Sample Query

The following query returns the record for the application with a `host_id` of `8`.

```
SELECT application_id, application_name, application_description, business_relevance,
risk

FROM application_info

WHERE application_id="8";
```

application_tag_map

The **application_tag_map** table contains information on the tags associated with each application detected on your network.

For more information, see the following sections:

- application_tag_map Fields, page 6-10
- application_tag_map Joins, page 6-10
- application_tag_map Sample Query, page 6-10

application_tag_map Fields

The following table describes the fields you can access in the **application_tag_map** table.

Table 6-6 application_tag_map Fields

Field	Description
application_id	The internal identification number for the application.
application_name	The application that appears in the user interface.
tag_id	The internal identification number for the tag.
tag_name	The text of the tag that appears in the user interface.
tag_type	One of the following: category or type.

application_tag_map Joins

The following table describes the joins you can perform on the **application_tag_map** table.

Table 6-7 application_tag_map Joins

You can join this table on...	And...
application_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_host_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id si_connection_log.application_protocol_name si_connection_log.application_protocol_id si_connection_log.client_application_id si_connection_log.web_application_id
tag_id	tag_info.tag_id

application_tag_map Sample Query

The following query returns all tag records associated with the specified application.

```
SELECT application_id, application_name, tag_id, tag_name
FROM application_tag_map
WHERE application_name="Active Directory";
```

network_discovery_event

The **network_discovery_event** table contains information on discovery and host input events. The FireSIGHT System generates discovery events when it detects a change on your monitored network, whether by discovering new network features or by detecting changes in previously identified network assets. The FireSIGHT System generates host input events when a user manually modifies the network map by adding, modifying, or deleting network assets.

The **network_discovery_event** table supersedes the deprecated **rna_events** table starting with Version 5.0 of the FireSIGHT System.

For more information, see the following sections:

- network_discovery_event Fields, page 6-11
- network_discovery_event Joins, page 6-12
- network_discovery_event Sample Query, page 6-12

network_discovery_event Fields

The following table describes the fields you can access in the **network_discovery_event** table.

Table 6-8 network_discovery_event Fields

Field	Description
confidence	The FireSIGHT System-assigned confidence rating (from 0 to 100) for the identification of the service.
description	The description of the event.
event_id	The internal identification number for the event.
event_time_sec	The UNIX timestamp of the date and time the event was generated.
event_time_usec	The microsecond increment of the event timestamp.
event_type	The event type. For example, New Host or Identity Conflict.
ip_address	This field has been deprecated and will now return null.
ipaddr	A binary representation of the IPv4 or IPv6 address for the host involved in the event.
mac_address	The MAC address of the host involved in the event.
mac_vendor	The NIC hardware vendor of the host involved in the event.
port	The port used by the network traffic that triggered the event.
sensor_address	The IP address of the managed device that generated the discovery event. Format is *ipv4_address,ipv6_address*.
sensor_name	The managed device that generated the discovery event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.

Table 6-8 network_discovery_event Fields (continued)

Field	Description
user_dept	The department of the user who last logged into the host.
user_email	The email address of the user who last logged into the host.
user_first_name	The first name of the user who last logged into the host.
user_id	The internal identification number for the user who last logged into the host.
user_last_name	The last name of the user who last logged into the host.
user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the host.
user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the host.
user_name	The user name of the user who last logged into the host.
user_phone	The phone number of the user who last logged into the host.

network_discovery_event Joins

The following table describes the joins you can perform using the **network_discovery_event** table.

Table 6-9 network_discovery_event Joins

You can join this table on...	And...
ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

network_discovery_event Sample Query

The following query returns discovery event records that include the user, detecting device name, timestamp, host IP address, and so on within the specified times.

```
SELECT sensor_name, event_time_sec, event_time_usec, event_type, ipaddr, user_id,
hex(mac_address), mac_vendor, port, confidence FROM network_discovery_event
WHERE event_time_sec
BETWEEN UNIX_TIMESTAMP("2013-01-01 00:00:00") AND UNIX_TIMESTAMP("2013-01-01 23:59:59")
ORDER BY event_time_sec DESC, event_time_usec DESC;
```

rna_host

The **rna_host** table contains basic information on the hosts in your monitored network.

This table supersedes **rna_ip_host** as of Version 5.2.

For more information, see the following sections:

- rna_host Fields, page 6-13

rna_host Fields

The following table describes the fields you can access in the **rna_host** table.

Table 6-10 **rna_host** ***Fields***

Field	Description
criticality	The host criticality level: None, Low, Medium, or High.
hops	The number of network hops from the host to the managed device that detected the host.
host_id	ID number of the host.
host_name	Name of the host.
host_type	The host type: Host, Router, Bridge, NAT Device, or Load Balancer.
jailbroken	A true-false flag indicating whether a mobile device operating system is jailbroken.
last_seen_sec	The UNIX timestamp of the date and time the system last detected host activity.
mobile	A true-false flag indicating whether the detected host is a mobile device.
netbios_name	The host NetBIOS name string.
notes	The contents of the Notes host attribute for the host.
vlan_id	The VLAN identification number, if applicable.
vlan_priority	The priority value included in the VLAN tag.
vlan_type	The type of encapsulated packet that contains the VLAN tag: • 0 — Ethernet • 1 — Token Ring

rna_host Joins

The following table describes the joins you can perform on the **rna_host** table.

Table 6-11 rna_host Joins

You can join this table on...	And...
host_id	application_host_map.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ip_map.host_id rna_host_ioc_state.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host Sample Query

The following query returns 25 **rna_host** records that include the host ID, VLAN ID, when the host was last seen, and the type of host, ordered by the type of host.

```
SELECT host_id, vlan_id, last_seen_sec, host_type
FROM rna_host
ORDER BY host_type
LIMIT 0, 25;
```

rna_host_attribute

The **rna_host_attribute** table contains information on the host attributes associated with each host in your monitored network. It supersedes the deprecated **rna_ip_host_attribute** table.

For more information, see the following sections:

- rna_host_attribute Fields, page 6-14
- rna_host_attribute Joins, page 6-15
- rna_host_attribute Sample Query, page 6-15

rna_host_attribute Fields

The following table describes the fields you can access in the **rna_host_attribute** table.

Table 6-12 rna_host_attribute Fields

Field	Description
attribute_name	The host attribute. For example, Host Criticality or Default White List.
attribute_value	The value of the host attribute.
host_id	ID number of the host.

rna_host_attribute Joins

The following table describes the joins you can perform on the **rna_host_attribute** table.

Table 6-13 rna_host_attribute Joins

You can join this table on...	And...
host_id	application_host_map.host_id rna_host.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_attribute Sample Query

The following query returns all host attributes and values associated with the selected host ID.

```
SELECT attribute_name, attribute_value

FROM rna_host_attribute

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_client_app

The **rna_host_client_app** table contains information on the client applications detected on the hosts in your monitored network. It supersedes the deprecated **rna_ip_host_client_app** table.

For more information, see the following sections:

- rna_host_client_app Fields, page 6-16
- rna_host_client_app Joins, page 6-16
- rna_host_client_app Sample Query, page 6-18

rna_host_client_app Fields

The following table describes the fields you can access in the **rna_host_client_app** table.

Table 6-14 rna_host_client_app Fields

Field	Description
application	Field deprecated in Version 5.0. Returns null for all queries.
application_protocol_id	An internal identifier for the detected application protocol.
application_protocol_name	One of: • the name of the application, if a positive identification can be made • pending if the system requires more data • blank if there is no application information in the connection
application_type	Field deprecated in Version 5.0. Returns null for all queries.
client_application_id	The internal identification number for the application, if the application is identifiable.
client_application_name	One of: • the name of the application, if a positive identification can be made. • a generic client name if the system detects a client application but cannot identify a specific one. • blank if there is no client application information in the connection.
hits	The number of times the client application was detected.
host_id	ID number of the host.
last_used_sec	The UNIX timestamp of the date and time the system last detected application activity.
version	The version of the application detected on the host.

rna_host_client_app Joins

The following table describes the joins you can perform on the **rna_host_client_app** table.

Table 6-15 rna_host_client_app Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

Table 6-15 rna_host_client_app Joins (continued)

You can join this table on...	And...
host_id and application_protocol_id and client_application_id and version	the set of: rna_host_client_app_payload.host_id rna_host_client_app_payload.application_protocol_id rna_host_client_app_payload.client_application_id rna_host_client_app_payload.version
application_protocol_id or client_application_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_host_map.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_service_info.application_protocol_id rna_host_service_payload.web_application_id

rna_host_client_app Sample Query

The following query returns information about the client applications detected on the host with `host_id` of `8`.

```
SELECT host_id, client_application_id, client_application_name, version, hits,
application_protocol_id, application_protocol_name, last_used_sec

FROM rna_host_client_app

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_client_app_payload

The **`rna_host_client_app_payload`** table contains information on the payloads in HTTP traffic associated with web applications on hosts detected in your monitored network.

For more information, see the following sections:

- rna_host_client_app_payload Fields, page 6-19
- rna_host_client_app_payload Joins, page 6-20
- rna_host_client_app_payload Sample Query, page 6-21

rna_host_client_app_payload Fields

The following table describes the fields you can access in the **rna_host_client_app_payload** table.

Table 6-16 rna_host_client_app_payload Fields

Field	Description
application	Field deprecated in Version 5.0. Returns null for all queries.
application_protocol_id	An internal identifier for the detected application protocol, if available. For traffic that has characteristics of both client applications and web applications, the client_application_id and web_application_id fields have the same value.
application_protocol_name	One of: • the name of the application, if a positive identification can be made • pending if the system requires more data • blank if there is no application information in the connection
application_type	Field deprecated in Version 5.0. Returns null for all queries.
client_application_id	The internal identification number for the client application.
client_application_name	One of: • the name of the application, if a positive identification can be made. • a generic client name if the system detects a client application but cannot identify a specific one. • blank if there is no client application information in the connection.
host_id	ID number of the host.
payload_name	Field deprecated in Version 5.0. Returns null for all queries.
payload_type	Field deprecated in Version 5.0. Returns null for all queries.
version	The version of the web application detected on the host.
web_application_id	The internal identification number for the web application, if available. For traffic that has characteristics of both client applications and web applications, the client_application_id and web_application_id fields have the same value.
web_application_name	One of: • the name of the application, if a positive identification can be made. • web browsing if the system detects an application protocol of HTTP but cannot identify a specific web application. • blank if the connection has no HTTP traffic.

rna_host_client_app_payload Joins

The following table describes the joins you can perform on the **rna_host_client_app_payload** table.

Table 6-17 rna_host_client_app_payload Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

Table 6-17 rna_host_client_app_payload Joins (continued)

You can join this table on...	And...
the set of: host_id, application_protocol_id, client_application_id, version	the set of: rna_host_client_app.host_id rna_host_client_app.application_protocol_id rna_host_client_app.client_application_id rna_host_client_app.version
client_application_id or web_application_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_host_map.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

rna_host_client_app_payload Sample Query

The following query returns information about the web applications detected on the host with `host_id` of `8`.

```
SELECT host_id, web_application_id, web_application_name, version,
client_application_id, client_application_name

FROM rna_host_client_app_payload

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_ioc_state

The **`rna_host_ioc_state`** table stores the IOC state for hosts in your monitored network.

For more information, see the following sections:

- rna_host_ioc_state Fields, page 6-22
- rna_host_ioc_state Joins, page 6-24
- rna_host_ioc_state Sample Query, page 6-24

rna_host_ioc_state Fields

The following table describes the fields you can access in the **rna_host_ioc_state** table.

Table 6-18 rna_host_ioc_state Fields

Field	Description
first_seen	Unix timestamp when the compromise was first detected.
first_seen_sensor_address	The IP address of the managed device that first detected the compromise. Format is *ipv4_address,ipv6_address*.
first_seen_sensor_name	The managed device that first detected the compromise.
host_id	ID number of the host.
ioc_category	The category for the compromise. Possible values include: • CnC Connected • Exploit Kit • High Impact Attack • Low Impact Attack • Malware Detected • Malware Executed • Dropper Infection • Java Compromise • Word Compromise • Adobe Reader Compromise • Excel Compromise • PowerPoint Compromise • QuickTime Compromise
ioc_description	Description of the compromise.

Table 6-18 rna_host_ioc_state Fields (continued)

Field	Description
ioc_event_type	The event type for the compromise. Possible values include: • Adobe Reader launched shell • Dropper Infection Detected by FireAMP • Excel Compromise Detected by FireAMP • Excel launched shell • Impact 1 Intrusion Event — attempted-admin • Impact 1 Intrusion Event — attempted-user • Impact 1 Intrusion Event — successful-admin • Impact 1 Intrusion Event — successful-user • Impact 1 Intrusion Event — web-application-attack • Impact 2 Intrusion Event — attempted-admin • Impact 2 Intrusion Event — attempted-user • Impact 2 Intrusion Event — successful-admin • Impact 2 Intrusion Event — successful-user • Impact 2 Intrusion Event — web-application-attack • Intrusion Event — exploit-kit • Intrusion Event — malware-backdoor • Intrusion Event — malware-CnC • Java Compromise Detected by FireAMP • Java launched shell • PDF Compromise Detected by FireAMP • PowerPoint Compromise Detected by FireAMP • PowerPoint launched shell • QuickTime Compromise Detected by FireAMP • QuickTime launched shell • Security Intelligence Event — CnC • Suspected Botnet Detected by FireAMP • Threat Detected by FireAMP — Subtype is 'executed' • Threat Detected by FireAMP — Subtype is not 'executed' • Threat Detected in File Transfer — Action is not 'block' • Word Compromise Detected by FireAMP • Word launched shell
ioc_id	Unique ID number for the compromise.
is_disabled	Whether this compromise has been disabled.
last_seen	Unix timestamp when this compromise was last detected.

Table 6-18 rna_host_ioc_state Fields (continued)

Field	Description
last_seen_sensor_address	The IP address of the managed device that last detected the compromise. Format is *ipv4_address,ipv6_address*.
last_seen_sensor_name	The managed device that last detected the compromise.

rna_host_ioc_state Joins

The following table describes the joins you can perform on the **rna_host_ioc_state** table.

Table 6-19 rna_host_ioc_state Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_ioc_state Sample Query

The following query returns up to 25 hosts with their ioc within a specified timespan.

```
SELECT host_id, ioc_id

FROM rna_host_ioc_state

WHERE first_seen

BETWEEN UNIX_TIMESTAMP("2011-10-01 00:00:00")

AND UNIX_TIMESTAMP("2011-10-07 23:59:59")

ORDER BY ioc_id DESC

LIMIT 0, 25;
```

rna_host_ip_map

The **rna_host_ip_map** table correlates host IDs to IP addresses for hosts in your monitored network.

For more information, see the following sections:

- rna_host_ip_map Fields, page 6-25
- rna_host_ip_map Joins, page 6-25
- rna_host_ip_map Sample Query, page 6-26

rna_host_ip_map Fields

The following table describes the fields you can access in the **rna_host_ip_map** table.

Table 6-20 rna_host_ip_map Fields

Field	Description
host_id	ID number of the host.
ipaddr	A binary representation of the IP address of the host.

rna_host_ip_map Joins

The following table describes the joins you can perform on the **rna_host_ip_map** table.

Table 6-21 rna_host_ip_map Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id
ipaddr	compliance_event.dst_ipaddr compliance_event.src_ipaddr connection_log.initiator_ipaddr connection_log.responder_ipaddr connection_summary.initiator_ipaddr connection_summary.responder_ipaddr fireamp_event.dst_ipaddr fireamp_event.src_ipaddr intrusion_event.dst_ipaddr intrusion_event.src_ipaddr network_discovery_event.ipaddr si_connection_log.initiator_ipaddr si_connection_log.responder_ipaddr user_discovery_event.ipaddr user_ipaddr_history.ipaddr white_list_event.ipaddr

rna_host_ip_map Sample Query

The following query returns MAC information for the selected host.

```
SELECT host_id

FROM rna_host_ip_map

WHERE HEX(ipaddr) = "00000000000000000000FFFF0A0A0A04";
```

rna_host_mac_map

The **rna_host_mac_map** table correlates host IDs to MAC addresses for hosts in your monitored network.

For more information, see the following sections:

- rna_host_mac_map Fields, page 6-27
- rna_host_mac_map Joins, page 6-27
- rna_host_mac_map Sample Query, page 6-28

rna_host_mac_map Fields

The following table describes the fields you can access in the **rna_host_mac_map** table.

Table 6-22 rna_host_mac_map Fields

Field	Description
host_id	ID number of the host.
mac_address	The host's MAC address.
mac_vendor	Vendor of the network interface of the detected host.

rna_host_mac_map Joins

The following table describes the joins you can perform on the **rna_host_mac_map** table.

Table 6-23 rna_host_mac_map Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_mac_map Sample Query

The following query returns MAC information for the host with `host_id` of `8`.

```
SELECT HEX(mac_address)
FROM rna_host_mac_map
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_os

The **`rna_host_os`** table contains information on the operating systems detected on the hosts in your monitored network.

For more information, see the following sections:

- rna_host_os Fields, page 6-28
- rna_host_os Joins, page 6-29
- rna_host_os Sample Query, page 6-29

rna_host_os Fields

The following table describes the fields you can access in the **`rna_host_os`** table.

Table 6-24 rna_host_os Fields

Field	Description
`confidence`	The FireSIGHT System-assigned confidence rating (from `0` to `100`) for the identification of the operating system.
`created_sec`	The UNIX timestamp of the date and time the system first detected host activity.
`host_id`	ID number of the host.
`last_seen_sec`	The UNIX timestamp of the date and time the system last detected host activity.
`os_uuid`	A unique identifier for the operating system detected on the host. The UUID maps to the operating system name, vendor, and version in the Cisco database.
`product`	The operating system detected on the host.
`source_type`	The source of the host's operating system identity: • `User` — Name of the user who entered the data via the web user interface • `Application` — Imported from another application via the host input feature • `Scanner` — Either Nmap or another scanner added through system policy • `rna` — Detected by the FireSIGHT System, either by a discovery event, port match, or pattern match • `NetFlow` — The data was exported by a NetFlow-enabled device
`vendor`	The vendor of the operating system detected on the host.
`version`	The version of the operating system detected on the host.

rna_host_os Joins

The following table describes the joins you can perform on the **rna_host_os** table.

Table 6-25 rna_host_os Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_os Sample Query

The following query returns operating system information for the host with `host_id` of `8`.

```
SELECT vendor, product, version, source_type, confidence

FROM rna_host_os

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_os_vulns

The **rna_host_os_vulns** table contains information on the vulnerabilities associated with the hosts in your monitored network.

For more information, see the following sections:

- rna_host_os_vulns Fields, page 6-30
- rna_host_os_vulns Joins, page 6-30
- rna_host_os_vulns Sample Query, page 6-30

rna_host_os_vulns Fields

The following table describes the fields you can access in the **rna_host_os_vulns** table.

Table 6-26 rna_host_os_vulns Fields

Field	Description
host_id	ID number of the host.
invalid	A value indicating whether the vulnerability is valid for the host: • 0 — Vulnerability is valid • 1 — Vulnerability is invalid
rna_vuln_id	An internal identification number for the vulnerability.

rna_host_os_vulns Joins

The following table describes the joins you can perform on the **rna_host_os_vulns** table.

Table 6-27 rna_host_os_vulns Joins

You can join this table on...	And...
rna_vuln_id	rna_vuln.bugtraq_id rna_vuln.rna_vuln_id rna_host_third_party_vuln_rna_id.rna_vuln_id rna_host_third_party_vuln_cve_id.cve_id rna_host_third_party_vuln_bugtraq_id.bugtraq_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_os_vulns Sample Query

The following query returns the operating system vulnerabilities for the host with host_id of 8.

```
SELECT rna_vuln_id, invalid
FROM rna_host_os_vulns
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_protocol

The **rna_host_protocol** table contains information on the protocols detected on the hosts in your monitored network.

For more information, see the following sections:

- rna_host_protocol Fields, page 6-31
- rna_host_protocol Joins, page 6-31
- rna_host_protocol Sample Query, page 6-32

rna_host_protocol Fields

The following table describes the fields you can access in the **rna_host_protocol** table.

Table 6-28 rna_host_protocol Fields

Field	Description
host_id	ID number of the host.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
layer	The network layer where the protocol is running: Network or Transport.
mac_address	Field deprecated in Version 5.2. Returns null for all queries.
mac_vendor	Field deprecated in Version 5.2. Returns null for all queries.
protocol_name	The traffic protocol used by the host.
protocol_num	The IANA-specified protocol number for the protocol.

rna_host_protocol Joins

The following table describes the joins you can perform on the **rna_host_protocol** table.

Table 6-29 rna_host_protocol Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_protocol Sample Query

The following query returns all protocol records for the host with `host_id` of `8`.

```
SELECT protocol_num, protocol_name

FROM rna_host_protocol

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_sensor

The **rna_host_sensor** table lists the host IP addresses in your monitored network and indicates the managed device that detected each one.

The **rna_host_sensor** table supersedes the deprecated **rna_ip_host_sensor** table starting with Version 5.2 of the FireSIGHT System.

For more information, see the following sections:

- rna_host_sensor Fields, page 6-32
- rna_host_sensor Joins, page 6-33
- rna_host_sensor Sample Query, page 6-33

rna_host_sensor Fields

The following table describes the fields you can access in the **rna_host_sensor** table.

Table 6-30 rna_host_sensor Fields

Field	Description
host_id	ID number of the host.
sensor_address	The IP address of the managed device that generated the discovery event. Format is *ipv4_address,ipv6_address*.
sensor_name	The name of the managed device.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.

rna_host_sensor Joins

The following table describes the joins you can perform on the **rna_host_sensor** table.

Table 6-31 rna_host_sensor Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_sensor Sample Query

The following query returns up to 25 hosts, and the sensor that detected them, from the **rna_host_sensor** table.

```
SELECT host_id, sensor_address, sensor_name

FROM rna_host_sensor

LIMIT 0, 25;
```

rna_host_service

The **rna_host_service** table contains general information about the servers detected on the hosts in your managed network through network port and traffic protocol combinations.

For more information, see the following sections:

- rna_host_service Fields, page 6-34
- rna_host_service Joins, page 6-34
- rna_host_service Sample Query, page 6-35

rna_host_service Fields

The following table describes the fields you can access in the **rna_host_service** table.

Table 6-32 rna_host_service Fields

Field	Description
confidence	The FireSIGHT System-assigned confidence rating (from 0 to 100) for the identification of the server.
hits	The number of times the server was detected.
host_id	ID number of the host.
last_used_sec	UNIX timestamp of the date and time the system last detected server activity.
port	The port used by the server.
protocol	The traffic protocol: TCP or UDP.

rna_host_service Joins

The following table describes the joins you can perform on the **rna_host_service** table.

Table 6-33 rna_host_service Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id
The set of: host_id port protocol	The set of: rna_host_service_banner.host_id rna_host_service_banner.port rna_host_service_banner.protocol The set of: rna_host_service_info.host_id rna_host_service_info.port rna_host_service_info.protocol The set of: rna_host_service_payload.host_id rna_host_service_payload.port rna_host_service_payload.protocol

rna_host_service Sample Query

The following query returns the first 25 detected server records for the host with host_id of 8:

```
SELECT hits, protocol, port, confidence
FROM rna_host_service
WHERE HEX(host_id) = "00000000000000000000000000000008"
LIMIT 0, 25;
```

rna_host_service_banner

The **rna_ip_host_service_banner** table contains header information from network traffic that advertises vendors and versions ("banners") for the servers on hosts in your monitored network. Keep in mind that the FireSIGHT System does not store server banners unless you enable the **Capture Banners** option in the your network discovery policy.

For more information, see the following sections:

- rna_ip_host_service_banner Fields, page 6-36
- rna_host_service_banner Joins, page 6-36
- rna_host_service_banner Sample Query, page 6-37

rna_ip_host_service_banner Fields

The following table describes the fields you can access in the **rna_host_service_banner** table.

Table 6-34 rna_host_service_banner Fields

Field	Description
banner	The server banner, that is, the first 256 bytes of the first packet detected for the server.
host_id	ID number of the host.
port	The port used by the server.
protocol	The traffic protocol: TCP or UDP.

rna_host_service_banner Joins

The following table describes the joins you can perform on the **rna_host_service_banner** table.

Table 6-35 rna_host_service_banner Joins

You can join this table on...	And...
The set of: host_id port protocol	The set of: rna_host_service.host_id rna_host_service.port rna_host_service.protocol The set of: rna_host_service_info.host_id rna_host_service_info.port rna_host_service_info.protocol The set of: rna_host_service_payload.host_id rna_host_service_payload.port rna_host_service_payload.protocol
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_service_banner Sample Query

The following query returns the server banner for the host with `host_id` of `8`.

```
SELECT port, protocol, banner

FROM rna_host_service_banner

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_service_info

The **`rna_host_service_info`** table contains detailed information about the servers detected on the hosts in your monitored network.

For more information, see the following sections:

- rna_host_service_info Fields, page 6-38
- rna_host_service_info Joins, page 6-39
- rna_host_service_info Sample Query, page 6-41

rna_host_service_info Fields

The following table describes the fields you can access in the **rna_host_service_info** table.

Table 6-36 rna_host_service_info Fields

Field	Description
application_id	Field deprecated in Version 5.0. Returns blank for all queries.
application_protocol_id	An internal identifier for the detected application protocol, if available.
application_protocol_name	One of: • the name of the application protocol, if a positive identification can be made • pending if the system requires more data • blank if there is no application information in the connection
business_relevance	An index (from 1 to 5) of the application's relevance to business productivity where 1 is very low and 5 is very high.
business_relevance_ description	A description of business relevance (very low, low, medium, high, very high).
created_sec	The UNIX timestamp of the date and time the system first detected the application protocol.
host_id	ID number of the host.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
last_used_sec	The UNIX timestamp of the date and time the system last detected server activity.
port	The port used by the server.
protocol	The traffic protocol: TCP or UDP.
risk	An index (from 1 to 5) of the application's risk where 1 is very low risk and 5 is very high risk.
risk_description	A description of the risk (very low, low, medium, high, very high).
service_info_id	An internal identification number for the server.
service_name	Field deprecated in Version 5.0. Returns null for all queries.

Table 6-36 rna_host_service_info Fields (continued)

Field	Description
source_type	The source of the identity of the server: • User — Name of the user who entered the data via the web user interface • Application — Imported from another application via the host input feature • Scanner — Added through NMAP or imported via the host input feature with a source type of Scanner • rna — Detected by the FireSIGHT System, either by a discovery event, port match, or pattern match • NetFlow — The data was exported by a NetFlow-enabled device
vendor	The vendor of the server on the host.
version	The version of the server detected on the host.

rna_host_service_info Joins

The following table describes the joins you can perform on the **rna_host_service_info** table.

Table 6-37 rna_host_service_info Joins

You can join this table on...	And...
application_protocol_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_host_map.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

Table 6-37 rna_host_service_info Joins (continued)

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_payload.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id
The set of: host_id and port and protocol	The set of: rna_host_service.host_id rna_host_service.port rna_host_service.protocol The set of: rna_host_service_banner.host_id rna_host_service_banner.port rna_host_service_banner.protocol The set of: rna_host_service_payload.host_id rna_host_service_payload.port rna_host_service_payload.protocol

rna_host_service_info Sample Query

The following query returns information about the application protocols detected on the host with host_id of 8.

```
SELECT host_id, application_protocol_name, version, vendor, created_sec, last_used_sec,
business_relevance, risk

FROM rna_host_service_info

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_service_payload

The **rna_host_service_payload** table contains information on the web applications associated by the hosts in your monitored network.

For more information, see the following sections:

rna_host_service_payload Fields

The following table describes the fields you can access in the **rna_host_service_payload** table.

Table 6-38 rna_host_service_payload Fields

Field	Description
application_id	Field deprecated in Version 5.0. Returns null for all queries.
application_name	Field deprecated in Version 5.0. Returns null for all queries.
host_id	ID number of the host.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
payload_name	Field deprecated in Version 5.0. Returns null for all queries.
payload_type	Field deprecated in Version 5.0. Returns null for all queries.
port	The port used by the server.
protocol	The traffic protocol: TCP or UDP.
web_application_id	The internal identification number for the web application.
web_application_name	One of: • the name of the web application, if a positive identification can be made • web browsing if the system detects an application protocol of HTTP but cannot identify a specific web application • blank if the connection has no HTTP traffic

rna_host_service_payload Joins

The following table describes the joins you can perform on the **rna_host_service_payload** table.

Table 6-39 rna_host_service_payload Joins

You can join this table on...	And...
web_application_id	app_ids_stats_current_timeframe.application_id application_info.application_id application_host_map.application_id application_tag_map.application_id app_stats_current_timeframe.application_id connection_log.application_protocol_id connection_log.client_application_id connection_log.web_application_id connection_summary.application_protocol_id si_connection_log.application_protocol_name si_connection_log.client_application_id si_connection_log.web_application_id file_event.application_id intrusion_event.application_protocol_id intrusion_event.client_application_id intrusion_event.web_application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id

Table 6-39 rna_host_service_payload Joins (continued)

You can join this table on...	And...
The set of: host_id port protocol	The set of: rna_host_service.host_id rna_host_service.port rna_host_service.protocol The set of: rna_host_service_banner.host_id rna_host_service_banner.port rna_host_service_banner.protocol The set of: rna_host_service_info.host_id rna_host_service_info.port rna_host_service_info.protocol
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_vulns.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_service_payload Sample Query

The following query returns information about the web applications detected on the host with `host_id` of `8`.

```
SELECT host_id, web_application_id, web_application_name, port, protocol

FROM rna_host_service_payload

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_service_subtype

The **rna_host_service_subtype** table contains information on the sub-servers for a server detected on the hosts in your monitored network.

For more information, see the following sections:

- rna_host_service_subtype Fields, page 6-45
- rna_host_service_subtype Joins, page 6-45
- rna_host_service_subtype Sample Query, page 6-46

rna_host_service_subtype Fields

The following table describes the fields you can access in the **rna_host_service_subtype** table.

Table 6-40 rna_host_service_subtype Fields

Field	Description
host_id	ID number of the host.
port	The port used by the server.
protocol	The traffic protocol: TCP or UDP.
service_name	One of: • the server on the host that is associated with the triggering event • none or blank if data for an identification is unavailable • pending if additional data is required • unknown if the system cannot identify the server based on known server fingerprints
source_type	The source of the identity of the server: • User - name of the user who entered the data via the web user interface • Application - imported from another application via the host input feature • Scanner - added through NMAP or imported via the host input feature with a source type of Scanner • rna - detected by the FireSIGHT System, either by a discovery event, port match, or pattern match • NetFlow - the data was exported by a NetFlow-enabled device
sub_service_name	The sub-server detected on the host.
sub_service_vendor	The vendor of the sub-server detected on the host.
sub_service_version	The version of the sub-server detected on the host.
vendor	The vendor of the server detected on the host.
version	The version of the server detected on the host.

rna_host_service_subtype Joins

You cannot perform joins on the **rna_host_service_subtype** table.

rna_host_service_subtype Sample Query

The following query returns all detected sub-server records for the host with `host_id` of `8`.

```
SELECT host_id, service_name, version, sub_service_name, sub_service_version,
sub_service_vendor

FROM rna_host_service_subtype

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_service_vulns

The **`rna_host_service_vulns`** table contains information on the vulnerabilities mapped to the servers detected on the hosts in your monitored network.

For more information, see the following sections:

- rna_host_service_vulns Fields, page 6-46
- rna_host_service_vulns Joins, page 6-47
- rna_host_service_vulns Sample Query, page 6-47

rna_host_service_vulns Fields

The following table describes the fields you can access in the **`rna_host_service_vulns`** table.

Table 6-41 rna_host_service_vulns Fields

Field	Description
`application_id`	An internal identification number for the application protocol running on the host.
`application_name`	The application protocol name that appears in the user interface.
`host_id`	ID number of the host.
`invalid`	A value indicating whether the vulnerability is valid for the host running the application protocol: • `0` — Vulnerability is valid • `1` — Vulnerability is invalid
`ip_address`	Field deprecated in Version 5.2. Returns `null` for all queries.
`port`	The port used by the server.
`protocol`	The traffic protocol: `TCP` or `UDP`.
`rna_vuln_id`	An internal identification number for the vulnerability.
`service_name`	Field deprecated in Version 5.0. Returns `null` for all queries.
`vendor`	The vendor of the server detected on the host.
`version`	The version of the server detected on the host.

rna_host_service_vulns Joins

The following table describes the joins you can perform on the **rna_host_service_vulns** table.

Table 6-42 rna_host_service_vulns Joins

You can join this table on...	And...
rna_vuln_id	rna_vuln.bugtraq_id rna_vuln.rna_vuln_id rna_host_third_party_vuln_rna_id.rna_vuln_id rna_host_third_party_vuln_cve_id.cve_id rna_host_third_party_vuln_bugtraq_id.bugtraq_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_payload.host_id rna_host_third_party_vuln_bugtraq_id.host_id rna_host_third_party_vuln_cve_id.host_id rna_host_third_party_vuln_rna_id.host_id rna_host_third_party_vuln.host_id

rna_host_service_vulns Sample Query

The following query returns information about all server vulnerabilities for the host with `host_id` of `8`.

```
SELECT host_id, rna_vuln_id, vendor, service_name, version, invalid FROM
rna_host_service_vulns
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_third_party_vuln

The **rna_host_third_party_vuln** table contains information on the third-party vulnerabilities associated with the hosts in your monitored network. Note that the information in this table is determined by the third-party vulnerability data imported via the host input feature.

For more information, see the following sections:

- rna_host_third_party_vuln Fields, page 6-48
- rna_host_third_party_vuln Joins, page 6-48

- rna_host_third_party_vuln Sample Query, page 6-49

rna_host_third_party_vuln Fields

The following table describes the fields you can access in the **rna_host_third_party_vuln** table.

Table 6-43 rna_host_third_party_vuln Fields

Field	Description
description	A description of the vulnerability.
host_id	ID number of the host.
invalid	A value indicating whether the vulnerability is valid for the host: • 0 — Vulnerability is valid • 1 — Vulnerability is invalid
name	The title of the vulnerability.
port	A port number, if the vulnerability is associated with a server or related application detected on a specific port.
protocol	The traffic protocol (TCP or UDP), if the vulnerability is associated with an application using that protocol.
source	The source of the vulnerability.
third_party_vuln_id	An identification number associated with the vulnerability.

rna_host_third_party_vuln Joins

The following table describes the joins you can perform on the **rna_host_third_party_vuln** table.

Table 6-44 rna_host_third_party_vuln Joins

You can join this table on...	And...
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id

rna_host_third_party_vuln Sample Query

The following query returns information about the third party vulnerabilities for host with `host_id` of `8`.

```
SELECT host_id, third_party_vuln_id, name, description, source, invalid
FROM rna_host_third_party_vuln
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_third_party_vuln_bugtraq_id

The **`rna_host_third_party_vuln_bugtraq_id`** table contains information on the third-party vulnerabilities that are mapped to vulnerabilities in the Bugtraq database and also associated with hosts in your monitored network. Note that the third-party vulnerability data in this table is imported via the host input feature.

For more information, see the following sections:

- rna_host_third_party_vuln_bugtraq_id Fields, page 6-49
- rna_host_third_party_vuln_bugtraq_id Joins, page 6-50
- rna_host_third_party_vuln_bugtraq_id Sample Query, page 6-50

rna_host_third_party_vuln_bugtraq_id Fields

The following table describes the fields you can access in the **`rna_host_third_party_vuln_bugtraq_id`** table.

Table 6-45 rna_host_third_party_vuln_bugtraq_id Fields

Field	Description
`bugtraq_id`	The Bugtraq database identification number associated with the vulnerability.
`description`	A description of the vulnerability.
`host_id`	ID number of the host.
`invalid`	A value indicating whether the vulnerability is valid for the host: • `0` — Vulnerability is valid • `1` — Vulnerability is invalid
`ip_address`	Field deprecated in Version 5.2. Returns `null` for all queries.
`name`	The name, or title, of the vulnerability.
`port`	A port number, if the vulnerability is associated with a server or related application detected on a specific port.
`protocol`	The traffic protocol (`TCP` or `UDP`), if the vulnerability is associated with an application using that protocol.
`source`	The source of the vulnerability.
`third_party_vuln_id`	The third-party identification number associated with the vulnerability.

rna_host_third_party_vuln_bugtraq_id Joins

The following table describes the joins you can perform on the **rna_host_third_party_vuln_bugtraq_id** table.

Table 6-46 rna_host_third_party_vuln_bugtraq_id Joins

You can join this table on...	And...
bugtraq_id	rna_vuln.bugtraq_id rna_vuln.rna_vuln_id rna_host_os_vulns.rna_vuln_id rna_host_service_vulns.rna_vuln_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id

rna_host_third_party_vuln_bugtraq_id Sample Query

The following query returns the BugTraq vulnerabilities for the host with `host_id` of `8`.

```
SELECT host_id, third_party_vuln_id, bugtraq_id, name, description, source, invalid
FROM rna_host_third_party_vuln_bugtraq_id
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_third_party_vuln_cve_id

The **rna_host_third_party_vuln_cve_id** table contains information on the third-party vulnerabilities that are mapped to vulnerabilities in MITRE's CVE database and also associated with the hosts in your monitored network. Note that this table contains third-party vulnerability data imported via the host input feature.

For more information, see the following sections:

- rna_host_third_party_vuln_cve_id Fields, page 6-51
- rna_host_third_party_vuln_cve_id Joins, page 6-51

- rna_host_third_party_vuln_cve_id Sample Query, page 6-52

rna_host_third_party_vuln_cve_id Fields

The following table describes the fields you can access in the **rna_host_third_party_vuln_cve_id** table.

Table 6-47 rna_host_third_party_vuln_cve_id Fields

Field	Description
cve_id	The identification number associated with the vulnerability in MITRE's CVE database.
description	A description of the vulnerability.
host_id	ID number of the host.
invalid	A value indicating whether the vulnerability is valid for the host: • 0 — Vulnerability is valid • 1 — Vulnerability is invalid
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
name	The name, or title, of the vulnerability.
port	A port number, if the vulnerability is associated with a server or related application detected on a specific port.
protocol	The traffic protocol (TCP or UDP), if the vulnerability is associated with an application using that protocol.
source	The source of the vulnerability.
third_party_vuln_id	The identification number associated with the vulnerability.

rna_host_third_party_vuln_cve_id Joins

The following table describes the joins you can perform on the **rna_host_third_party_vuln_cve_id** table.

Table 6-48 rna_host_third_party_vuln_cve_id Joins

You can join this table on...	And...
cve_id	rna_vuln.bugtraq_id rna_vuln.rna_vuln_id rna_host_os_vulns.rna_vuln_id rna_host_service_vulns.rna_vuln_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id

rna_host_third_party_vuln_cve_id Sample Query

The following query returns the CVE vulnerabilities for the host with `host_id` of `8`.

```
SELECT host_id, third_party_vuln_id, cve_id, name, description, source, invalid

FROM rna_host_third_party_vuln_cve_id

WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_host_third_party_vuln_rna_id

The **`rna_host_third_party_vuln_rna_id`** table contains information on third-party vulnerabilities that are mapped to vulnerabilities in the Cisco vulnerability database (VDB) and also associated with hosts in your monitored network. Note that the third-party vulnerability data in this table is imported via the host input feature.

For more information, see the following sections:

- rna_host_third_party_vuln_rna_id Fields, page 6-53
- rna_host_third_party_vuln_rna_id Joins, page 6-53
- rna_host_third_party_vuln_rna_id Sample Query, page 6-54

rna_host_third_party_vuln_rna_id Fields

The following table describes the fields you can access in the **rna_host_third_party_vuln_rna_id** table.

Table 6-49 rna_host_third_party_vuln_rna_id Fields

Field	Description
description	A description of the vulnerability.
host_id	ID number of the host.
invalid	A value indicating whether the vulnerability is valid for the host: • 0 — Vulnerability is valid • 1 — Vulnerability is invalid
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
name	The name, or title, of the vulnerability.
port	A port number, if the vulnerability is associated with a server or related application detected on a specific port.
protocol	The traffic protocol (TCP or UDP), if the vulnerability is associated with an application using that protocol.
rna_vuln_id	The vulnerability identification number that Cisco uses to track the vulnerability.
source	The source of the vulnerability.
third_party_vuln_id	The identification number associated with the vulnerability.

rna_host_third_party_vuln_rna_id Joins

The following table describes the joins you can perform on the **rna_host_third_party_vuln_rna_id** table.

Table 6-50 rna_host_third_party_vuln_rna_id Joins

You can join this table on...	And...
rna_vuln_id	rna_vuln.bugtraq_id rna_vuln.rna_vuln_id rna_host_os.rna_vuln_id rna_host_service_vulns.rna_vuln_id
host_id	rna_host.host_id rna_host_attribute.host_id rna_host_protocol.host_id rna_host_os_vulns.host_id application_host_map.host_id rna_host_client_app.host_id rna_host_client_app_payload.host_id rna_host_ioc_state.host_id rna_host_ip_map.host_id rna_host_mac_map.host_id rna_host_os.host_id rna_host_sensor.host_id rna_host_service.host_id rna_host_service_banner.host_id rna_host_service_info.host_id rna_host_service_payload.host_id rna_host_service_vulns.host_id

rna_host_third_party_vuln_rna_id Sample Query

The following query returns all third party vulnerabilities with VDB IDs for the host with `host_id` of `8`.

```
SELECT host_id, third_party_vuln_id, rna_vuln_id, name, description, source, invalid
FROM rna_host_third_party_vuln_rna_id
WHERE HEX(host_id) = "00000000000000000000000000000008";
```

rna_vuln

The **`rna_vuln`** table contains information on the vulnerabilities in the Cisco VDB.

For more information, see the following sections:

- rna_vuln Fields, page 6-54
- rna_vuln Joins, page 6-56
- rna_vuln Sample Query, page 6-56

rna_vuln Fields

The following table describes the fields you can access in the **`rna_vuln`** table.

Table 6-51 rna_vuln Fields

Field	Description
authentication	Whether authentication is required to exploit the vulnerability: • Required • Not Required • Unknown
availability	When the vulnerability can be exploited: • Always • User Initiated • Time Dependent • Unknown
available_exploits	Whether there are available exploits for the vulnerability: • TRUE • FALSE
bugtraq_id	The identification number associated with the vulnerability in the Bugtraq database.
class	The class of vulnerability: • Configuration Error • Boundary Condition Error • Design Error
credibility	How credible the vulnerability is: • Conflicting Reports • Conflicting Details • Single Source • Reliable Source • Multiple Sources • Vendor Confirmed
credit	The person or organization credited with reporting the vulnerability.
ease	The ease of exploiting the vulnerability: • No Exploit Required • Exploit Available • No Exploit Available
effect	Details on what could happen when the vulnerability is exploited.
entry_date	The date the vulnerability was entered in the database.
exploit	Information on where you can find exploits for the vulnerability.
impact	The vulnerability impact, corresponding to the impact level determined through correlation of intrusion data, discovery events, and vulnerability assessments. The value can be from 1 to 10, with 10 being the most severe. The impact value of a vulnerability is determined by the writer of the Bugtraq entry.

Table 6-51 rna_vuln Fields (continued)

Field	Description
local	Indicates whether the vulnerability must be exploited locally: • TRUE • FALSE
long_description	A general description of the vulnerability.
mitigation	A description of how you can mitigate the vulnerability.
modified_date	The date of the most recent modification to the vulnerability, if applicable.
publish_date	The date the vulnerability was published.
remote	Indicates whether the vulnerability can be exploited across a network: • TRUE • FALSE
rna_vuln_id	The Cisco vulnerability ID number that the system uses to track vulnerabilities.
scenario	A description of a scenario where an attacker is exploiting the vulnerability.
short_description	A summary description of the vulnerability.
snort_id	The identification number associated with the vulnerability in the Snort ID (SID) database. That is, if an intrusion rule can detect network traffic that exploits a particular vulnerability, that vulnerability is associated with the intrusion rule's SID.
solution	The solution to the vulnerability.
technical_description	The technical description of the vulnerability.
title	The title of the vulnerability.

rna_vuln Joins

The following table describes the joins you can perform on the **rna_vuln** table.

Table 6-52 rna_vuln Joins

You can join this table on...	And...
rna_vuln_id or bugtraq_id	rna_host_os_vulns.rna_vuln_id rna_host_service_vulns.rna_vuln_id rna_host_third_party_vuln_rna_id.rna_vuln_id rna_host_third_party_vuln_cve_id.cve_id rna_host_third_party_vuln_bugtraq_id.bugtraq_id

rna_vuln Sample Query

The following query returns information about up to 25 vulnerabilities. The records are sorted in order of most events generated based on the vulnerability.

```
SELECT rna_vuln_id, bugtraq_id, snort_id, title, publish_date, impact, remote, exploit,
long_description, technical_description, solution, count(*) as count

FROM rna_vuln
```

```
GROUP BY rna_vuln_id
ORDER BY rna_vuln_id DESC LIMIT 0, 25;
```

tag_info

The **tag_info** table contains information on the tags that are associated with the applications detected on your network. Note that an application can have multiple associated tags.

For more information, see the following sections:

- tag_info Fields, page 6-57
- tag_info Joins, page 6-57
- tag_info Sample Query, page 6-57

tag_info Fields

The following table describes the fields you can access in the **tag_info** table.

Table 6-53 tag_info Fields

Field	Description
tag_description	Tag description.
tag_id	Internal identifier for the tag.
tag_name	Text of the tag that appears in the user interface.
tag_type	One of the following: • category • tag

tag_info Joins

The following table describes the joins you can perform on the **tag_info** table.

Table 6-54 tag_info Joins

You can join this table on...	And...
tag_id	application_tag_map.tag_id

tag_info Sample Query

The following query returns the application tag record for a selected tag ID.

```
SELECT tag_id, tag_name, tag_type, tag_description
FROM tag_info
WHERE tag_id="100";
```

url_categories

The **url_categories** table lists the categories that characterize URLs requested by hosts in your monitored network.

For more information, see the following sections:

- url_categories Fields, page 6-58
- url_categories Joins, page 6-58
- url_categories Sample Query, page 6-58

url_categories Fields

The following table describes the fields in the **url_categories** table.

Table 6-55 url_categories Fields

Field	Description
category_description	The description of the URL category.
category_id	The internal identification number of the URL category.

url_categories Joins

You cannot perform joins on the **url_categories** table.

url_categories Sample Query

The following query returns a category record for the selected category ID.

```
SELECT category_id, category_description
FROM url_categories
WHERE category_id="1";
```

url_reputations

The **url_reputations** table lists the reputations that characterize URLs requested by hosts in your monitored request.

For more information, see the following sections:

- url_reputations Fields, page 6-59
- url_reputations Joins, page 6-59
- url_reputations Sample Query, page 6-59

url_reputations Fields

The following table describes the fields in the **url_reputations** table.

Table 6-56 url_reputations Fields

Field	Description
reputation_description	The description of the reputation.
reputation_id	An internal identification number for the URL reputation.

url_reputations Joins

You cannot perform joins on the **url_reputations** table.

url_reputations Sample Query

The following query returns URL reputation information for a reputation ID.

```
SELECT reputation_id, reputation_description
FROM url_reputations
WHERE reputation_id="1";
```

user_ipaddr_history

The **user_ipaddr_history** table contains information on user activity for a particular host in your monitored network.

For more information, see the following sections:

- user_ipaddr_history Fields, page 6-59
- user_ipaddr_history Joins, page 6-60
- user_ipaddr_history Sample Query, page 6-61

user_ipaddr_history Fields

The following table describes the fields you can access in the **user_ipaddr_history** table.

Table 6-57 user_ipaddr_history Fields

Field	Description
end_time_sec	The UNIX timestamp of the date and time the FireSIGHT System detected a different user logging into the host, marking the assumed end of the previous user's session. Note that the FireSIGHT System does not detect logoffs.
id	An internal identification number for the user history record.
ipaddr	A binary representation of the IP address of the host.

Table 6-57 user_ipaddr_history Fields (continued)

Field	Description
start_time_sec	The UNIX timestamp of the date and time the FireSIGHT System detected the user logging into host.
user_dept	The department of the user.
user_email	The email address of the user.
user_first_name	The first name of the user.
user_id	An internal identification number for the user.
user_last_name	The last name of the user.
user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user.
user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user.
user_name	The user name of the user.
user_phone	The phone number of the user.
user_rna_service	Name of the application protocol being used when the user was detected, if available.

user_ipaddr_history Joins

The following table describes the joins you can perform on the **user_ipaddr_history** table.

Table 6-58 user_ipaddr_history Joins

You can join this table on...	And...
ipaddr	compliance_event.dst_ipaddr compliance_event.src_ipaddr connection_log.initiator_ipaddr connection_log.responder_ipaddr connection_summary.initiator_ipaddr connection_summary.responder_ipaddr fireamp_event.dst_ipaddr fireamp_event.src_ipaddr intrusion_event.dst_ipaddr intrusion_event.src_ipaddr network_discovery_event.ipaddr rna_host_ip_map.ipaddr si_connection_log.initiator_ipaddr si_connection_log.responder_ipaddr user_discovery_event.ipaddr white_list_event.ipaddr
user_id	discovered_users.user_id user_discovery_event.user_id

user_ipaddr_history Sample Query

The following query returns all user activity records for the selected IP address after the specified start timestamp.

```
SELECT ipaddr, start_time_sec, end_time_sec, user_name, user_rna_service,
user_last_seen_sec, user_last_updated_sec

FROM user_ipaddr_history

WHERE HEX(ipaddr) = "00000000000000000000FFFF0A0A0A04" AND start_time_sec >=
UNIX_TIMESTAMP("2011-10-01 00:00:00");
```

Schema: Connection Log Tables

This chapter contains information on the schema and supported joins for connection data.

For more information, see the sections listed in the following table. The Version column indicates the Database Access versions supported by each listed table.

Table 7-1 Schema for Connection Log Tables

See...	For the table that stores information on...	Version
connection_log, page 7-1	Individual connections. Supersedes deprecated table **rna_flow**.	5.0+
connection_summary, page 7-12	Connection log summaries. Supersedes deprecated table **rna_flow_summary**.	5.0+
si_connection_log, page 7-16	Individual connections. Used for security intelligence.	5.3+

connection_log

The **connection_log** table contains information on connection events. The FireSIGHT System generates a connection event when a connection between a monitored host and any other host is established; the event contains detailed information about the monitored traffic.

The **connection_log** table supersedes the deprecated **rna_flow** table starting with Version 5.0 of the FireSIGHT System.

For more information, see the following sections:

- connection_log Fields, page 7-1
- connection_log Joins, page 7-12
- connection_log Sample Query, page 7-12

connection_log Fields

The following table describes the database fields you can access in the **connection_log** table.

Table 7-2 connection_log Fields

Field	Description
access_control_policy_name	The access control policy that contains the access control rule (or default action) that logged the connection.
access_control_policy_UUID	The UUID of the access control policy that contains the access control rule (or default action) that logged the connection.
access_control_reason	The reason that the access control rule logged the connection. One or more of the following: • IP Block • IP Monitor • User Bypass • File Monitor • File Block • Intrusion Monitor • Intrusion Block • File Resume Block • File Resume Allow • File Custom Detection • SSL Block • DNS Block • DNS Monitor • URL Block • URL Monitor • HTTP Injection • Intelligent App Bypass • blank if there is no connection logged
access_control_rule_action	The action associated with the access control rule (or default action): allow, block, and so on.
access_control_rule_id	An internal identification number for the rule.
access_control_rule_name	The access control rule (or default action) that logged the connection.
application_protocol_id	An internal identification number of the application protocol.
application_protocol_name	One of: • the name of the application, if a positive identification can be made • unknown if the system cannot identify the server based on known server fingerprints • pending if the system requires more data • blank if there is no application information in the connection
bytes_recv	The total number of bytes transmitted by the session responder.
bytes_sent	Total number of bytes transmitted by the session initiator.
cert_valid_end_date	The Unix timestamp on which the SSL certificate used in the connection ceases to be valid.

Table 7-2 connection_log Fields (continued)

Field	Description
cert_valid_start_date	The Unix timestamp when the SSL certificate used in the connection was issued.
client_application_id	An internal identification number for the client application that was used in the intrusion event.
client_application_name	The client application, if available, that was used in the intrusion event. One of: • the name of the application, if a positive identification can be made. • a generic client name if the system detects a client application but cannot identify a specific one. • blank if there is no client application information in the connection.
client_application_version	The version of the client application.
connection_type	The detection source for the connection information. Either: • rna, if detected by a Cisco device • netflow, if exported by a NetFlow-enabled device
counter	Counter for the intrusion event associated with the connection event.
file_count	The number of files identified by Snort in a session. A record is generated for each file identified in the session.
first_packet_sec	The UNIX timestamp of the date and time the first packet of the session was seen.
flow_id	An internal identification number for the connection.
icmp_code	ICMP code if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.
icmp_type	ICMP type if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.
initiator_continent_name	The name of the continent of the host that initiated the session: ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
initiator_country_id	Code for the country of the host that initiated the session.
initiator_country_name	Name of the country of the host that initiated the session.
initiator_ip	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
initiator_ip_address	Field deprecated in Version 5.0. Returns null for all queries.

Table 7-2 connection_log Fields (continued)

Field	Description
initiator_ipaddr	A binary representation of the IP address of the host that initiated the session.
initiator_ipv4	Field deprecated in Version 5.2. Returns null for all queries.
initiator_port	The port used by the session initiator.
initiator_user_dept	The department of the user who last logged into the initiator host.
initiator_user_email	The email address of the user who last logged into the initiator host.
initiator_user_first_name	The first name of the user who last logged into the initiator host.
initiator_user_id	An internal identification number for the user who last logged into the initiator host.
initiator_user_last_name	The last name of the user who last logged into the initiator host.
initiator_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the initiator host.
initiator_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the initiator host.
initiator_user_name	The user name of the user who last logged into the initiator host.
initiator_user_phone	The phone number of the user who last logged into the initiator host.
instance_id	Numerical ID of the Snort instance on the managed device that generated the event.
interface_egress_name	The ingress interface associated with the connection.
interface_ingress_name	The egress interface associated with the connection.
ioc_count	Number of indications of compromise found in the connection.
ips_event_count	The number of intrusion events generated in the connection prior to intrusion event thresholding.
last_packet_sec	The UNIX timestamp of the date and time the last packet of the session was seen.
monitor_rule_id_1	The ID of the first monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_1.
monitor_rule_id_2	The ID of the second monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_2.
monitor_rule_id_3	The ID of the third monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_3.
monitor_rule_id_4	The ID of the fourth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_4.
monitor_rule_id_5	The ID of the fifth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_5.
monitor_rule_id_6	The ID of the sixth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_6.
monitor_rule_id_7	The ID of the seventh monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_7.

Table 7-2 connection_log Fields (continued)

Field	Description
monitor_rule_id_8	The ID of the eighth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_8.
monitor_rule_name_1	The name of the first monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_1.
monitor_rule_name_2	The name of the second monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_2.
monitor_rule_name_3	The name of the third monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_3.
monitor_rule_name_4	The name of the fourth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_4.
monitor_rule_name_5	The name of the fifth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_5.
monitor_rule_name_6	The name of the sixth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_6.
monitor_rule_name_7	The name of the seventh monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_7.
monitor_rule_name_8	The name of the eighth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_8.
netbios_domain	The NetBIOS domain used in the connection.
netflow_dst_as	Netflow autonomous system number of the destination, either origin or peer.
netflow_dst_mask	Netflow destination address prefix mask.
netflow_dst_tos	Type of service from the IP header when packets are flowing from the destination to the source.
netflow_snmp_in	ID of the interface used by packets flowing from the source to the destination.
netflow_snmp_out	ID of the interface used by packets flowing from the destination to the source.
netflow_src_as	Netflow autonomous system number of the source, either origin or peer.
netflow_src_mask	Netflow source address prefix mask.
netflow_src_tos	Type of service from the IP header when packets are flowing from the source to the destination.
network_analysis_policy_name	The network analysis policy associated with the intrusion policy that generated the intrusion event.
network_analysis_policy_UUID	The UUID of the network analysis policy associated with the intrusion policy that generated the intrusion event.
packets_recv	The total number of packets received by the host that initiated the session.
packets_sent	The total number of packets transmitted by the host that initiated the session.
protocol_name	The name of the protocol used in the connection.

Table 7-2 connection_log Fields (continued)

Field	Description
protocol_num	The IANA number of the protocol as listed in http://www.iana.org/assignments/protocol-numbers.
responder_continent_name	The name of the continent of the host that responded to the session initiator: ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
responder_country_id	Code for the country of the host that responded to the session initiator.
responder_country_name	Name of the country of the host that responded to the session initiator.
responder_ip	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
responder_ip_address	Field deprecated in Version 5.2. Returns null for all queries.
responder_ipaddr	A binary representation of the IPv4 or IPv6 address for the host that responded to the session initiator.
responder_ipv4	Field deprecated in Version 5.2. Returns null for all queries.
responder_port	The port used by the session responder.
responder_user_dept	The department of the user who last logged into the host that responded to the session initiator.
responder_user_email	The email address of the user who last logged into the host that responded to the session initiator.
responder_user_first_name	The first name of the user who last logged into the host that responded to the session initiator.
responder_user_id	An internal identification number for the user who last logged into the host that responded to the session initiator.
responder_user_last_name	The last name of the user who last logged into the host that responded to the session initiator.
responder_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the host that responded to the session initiator.
responder_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the host that responded to the session initiator.
responder_user_name	The user name of the user who last logged into the host that responded to the session initiator.
responder_user_phone	The phone number of the user who last logged into the host that responded to the session initiator.

Table 7-2 connection_log Fields (continued)

Field	Description
security_context	Description of the security context (virtual firewall) that the traffic passed through. Note that the system only populates this field for ASA FirePOWER devices in multi-context mode.
security_intelligence_category	The Security Intelligence category associated with the connection.
security_intelligence_ip	Whether the Security Intelligence-monitored IP address associated with the connection is a source IP (src) or destination IP (dst).
security_zone_egress_name	The egress security zone in the connection event.
security_zone_ingress_name	The ingress security zone in the connection event.
sensor_address	The IP address of the managed device that generated the event. Format is ipv4 address,ipv6 address.
sensor_name	The name of the managed device that monitored the session.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
source_device	Field deprecated in Version 5.0. Returns null for all queries.
src_device_ip	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
src_device_ipaddr	Either: • A binary representation of the IP address of the NetFlow-enabled device that exported the connection data • 0, for connections detected by Cisco managed devices.
src_device_ipv4	• Field deprecated in Version 5.2. Returns null for all queries.
ssl_actual_action	The action performed on the connection based on the SSL Rule. This may differ from the expected action, as the action as specified in the rule may be impossible. Possible values include: • Unknown • Do Not Decrypt • Block • Block With Reset • Decrypt (Known Key) • Decrypt (Replace Key) • Decrypt (Resign)
ssl_cipher_suite	Encryption suite used by the SSL connection. The value is stored in decimal format. See www.iana.org/assignments/tls-parameters/tls-parameters.xhtml for the cipher suite designated by the value.

Table 7-2 connection_log Fields (continued)

Field	Description
ssl_expected_action	The action which should be performed on the connection based on the SSL Rule. Possible values include: • *Unknown* • *Do Not Decrypt* • *Block* • *Block With Reset* • *Decrypt (Known Key)* • *Decrypt (Replace Key)* • *Decrypt (Resign)*
ssl_flow_flags	The debugging level flags for an encrypted connection. Possible values include: • 0x00000001 — NSE_FLOW__VALID — must be set for other fields to be valid • 0x00000002 — NSE_FLOW__INITIALIZED — internal structures ready for processing • 0x00000004 — NSE_FLOW__INTERCEPT — SSL session has been intercepted

Table 7-2 connection_log Fields (continued)

Field	Description
ssl_flow_messages	The messages exchanged between client and server during the SSL handshake. See http://tools.ietf.org/html/rfc5246 for more information. • 0x00000001 — NSE_MT__HELLO_REQUEST • 0x00000002 — NSE_MT__CLIENT_ALERT • 0x00000004 — NSE_MT__SERVER_ALERT • 0x00000008 — NSE_MT__CLIENT_HELLO • 0x00000010 — NSE_MT__SERVER_HELLO • 0x00000020 — NSE_MT__SERVER_CERTIFICATE • 0x00000040 — NSE_MT__SERVER_KEY_EXCHANGE • 0x00000080 — NSE_MT__CERTIFICATE_REQUEST • 0x00000100 — NSE_MT__SERVER_HELLO_DONE • 0x00000200 — NSE_MT__CLIENT_CERTIFICATE • 0x00000400 — NSE_MT__CLIENT_KEY_EXCHANGE • 0x00000800 — NSE_MT__CERTIFICATE_VERIFY • 0x00001000 — NSE_MT__CLIENT_CHANGE_CIPHER_SPEC • 0x00002000 — NSE_MT__CLIENT_FINISHED • 0x00004000 — NSE_MT__SERVER_CHANGE_CIPHER_SPEC • 0x00008000 — NSE_MT__SERVER_FINISHED • 0x00010000 — NSE_MT__NEW_SESSION_TICKET • 0x00020000 — NSE_MT__HANDSHAKE_OTHER • 0x00040000 — NSE_MT__APP_DATA_FROM_CLIENT • 0x00080000 — NSE_MT__APP_DATA_FROM_SERVER

Table 7-2 connection_log Fields (continued)

Field	Description
ssl_flow_status	Status of the SSL Flow. These values describe the reason behind the action taken or the error message seen. Possible values include: • 'Unknown' • 'No Match' • 'Success' • 'Uncached Session' • 'Unknown Cipher Suite' • 'Unsupported Cipher Suite' • 'Unsupported SSL Version' • 'SSL Compression Used' • 'Session Undecryptable in Passive Mode' • 'Handshake Error' • 'Decryption Error' • 'Pending Server Name Category Lookup' • 'Pending Common Name Category Lookup' • 'Internal Error' • 'Network Parameters Unavailable' • 'Invalid Server Certificate Handle' • 'Server Certificate Fingerprint Unavailable' • 'Cannot Cache Subject DN' • 'Cannot Cache Issuer DN' • 'Unknown SSL Version' • 'External Certificate List Unavailable' • 'External Certificate Fingerprint Unavailable' • 'Internal Certificate List Invalid' • 'Internal Certificate List Unavailable' • 'Internal Certificate Unavailable' • 'Internal Certificate Fingerprint Unavailable' • 'Server Certificate Validation Unavailable' • 'Server Certificate Validation Failure' • 'Invalid Action'
ssl_issuer_common_name	Issuer Common name from the SSL certificate. This is typically the host and domain name of the certificate issuer, but may contain other information.
ssl_issuer_country	The country of the SSL certificate issuer.

Table 7-2 connection_log Fields (continued)

Field	Description
ssl_issuer_organization	The organization of the SSL certificate issuer.
ssl_issuer_organization_unit	The organizational unit of the SSL certificate issuer.
ssl_policy_action	The default action configured for the policy when no rules match.
ssl_policy_name	ID number of the SSL policy that handled the connection.
ssl_policy_reason	The reason the SSL policy logged the SSL session.
ssl_rule_action	The action selected in the user interface for the SSL rule (`allow`, `block`, and so forth).
ssl_rule_name	ID number of the SSL rule or default action that handled the connection.
ssl_serial_number	The serial number of the SSL certificate, assigned by the issuing CA.
ssl_server_name	Name provided in the server name indication in the SSL Client Hello.
ssl_subject_common_name	Subject Common name from the SSL certificate. This is typically the host and domain name of the certificate subject, but may contain other information.
ssl_subject_country	The country of the SSL certificate subject.
ssl_subject_organization	The organization of the SSL certificate subject.
ssl_subject_organization_unit	The organizational unit of the SSL certificate subject.
ssl_url_category	Category of the flow as identified from the server name and certificate common name.
ssl_version	The SSL or TLS protocol version used to encrypt the connection.
tcp_flags	The TCP flags detected in the session.
url	The URL requested by the monitored host during the session, if available.
url_category	The category of the URL requested by the monitored host.
url_reputation	The reputation of the URL requested by the monitored host. One of the following: • `1` — High risk • `2` — Suspicious sites • `3` — Benign sites with security risks • `4` — Benign sites • `5` — Well known
web_application_id	An internal identification number for the web application.
web_application_name	One of: • the name of the application, if a positive identification can be made. • `web browsing` if the system detects an application protocol of HTTP but cannot identify a specific web application. • blank if the connection has no HTTP traffic.

connection_log Joins

The following table describes the joins you can perform using the **connection_log** table.

Table 7-3 connection_log Joins

You can join this table on...	And...
application_protocol_id or client_application_id or web_application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id
initiator_ipaddr or responder_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

connection_log Sample Query

The following query returns up to 25 connection event records from the **connection_log** table, sorted in descending order based on packet timestamps.

```
SELECT first_packet_sec, last_packet_sec, initiator_ipaddr, responder_ipaddr,
security_zone_ingress_name, security_zone_egress_name, initiator_port, protocol_name,
responder_port, application_protocol_id, client_application_id, web_application_id, url,
url_category, url_reputation

FROM connection_log

WHERE first_packet_sec <= UNIX_TIMESTAMP("2011-10-01 00:00:00" ) ORDER BY
first_packet_sec

DESC, last_packet_sec DESC LIMIT 0, 25;
```

connection_summary

The **connection_summary** table contains information on connection summaries or aggregated connections. The FireSIGHT System aggregates connections over five-minute intervals. To be aggregated, connections must:

- have the same source and destination IP addresses
- use the same protocol
- use the same application
- either be detected by the same managed device (for sessions detected by managed devices with FireSIGHT) or be exported by the same NetFlow-enabled device and processed by the same managed device

The aggregated data in a connection summary includes the total number of packets and bytes sent by the initiator and responder hosts, as well as the number of connections in the summary.

The **connection_summary** table supersedes the deprecated **rna_flow_summary** table starting with Version 5.0 of the FireSIGHT System.

For more information, see the following sections:

- connection_summary Fields, page 7-13
- connection_summary Joins, page 7-15
- connection_summary Sample Query, page 7-15

connection_summary Fields

The following table describes the database fields you can access in the **connection_summary** table.

Table 7-4 connection_summary Fields

Field	Description
application_protocol_id	An internal identification number for the application protocol.
application_protocol_name	One of: • the name of the application, if a positive identification can be made • unknown if the system cannot identify the server based on known server fingerprints • pending if the system requires more data • blank if there is no application information in the connection
bytes_recv	The total number of bytes transmitted by the session responder.
bytes_sent	The total number of bytes transmitted by the session initiator.
connection_type	The detection source for the connection information. Either: • rna, if detected by a Cisco device • netflow, if exported by a NetFlow-enabled device
flow_type	Field deprecated in Version 5.0. Returns null for all queries.
id	An internal identification number for the connection summary.
initiator_ip_address	Field deprecated in Version 5.2. Returns null for all queries.
initiator_ipaddr	A binary representation of the IP address of the host that initiated the session.
initiator_user_dept	The department of the user who last logged into the initiator host.
initiator_user_email	The email address of the user who last logged into the initiator host.
initiator_user_first_name	The first name of the user who last logged into the initiator host.
initiator_user_id	An internal identification number for the user who last logged into the initiator host.
initiator_user_last_name	The last name of the user who last logged into the initiator host.
initiator_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the initiator host.

Table 7-4 connection_summary Fields (continued)

Field	Description
initiator_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the initiator host.
initiator_user_name	The user name of the user who last logged into the initiator host.
initiator_user_phone	The phone number of the user who last logged into the initiator host.
interface_egress_name	The ingress interface associated with the connection.
interface_ingress_name	The egress interface associated with the connection.
num_connections	The number of connections in the summary. For long-running connections, that is, connections that span multiple connection summary intervals, only the first connection summary is incremented.
packets_recv	The total number of packets transmitted by the session responder.
packets_sent	The total number of packets transmitted by the session initiator.
protocol_name	The name of the protocol used in the aggregated sessions.
protocol_num	The IANA number of the protocol as listed in http://www.iana.org/assignments/protocol-numbers.
responder_ip_address	Field deprecated in Version 5.2. Returns null for all queries.
responder_ipaddr	A binary representation of the IP address of the host that responded to the initiator of the aggregated sessions.
responder_port	The port used by the responder in the aggregated sessions.
responder_user_dept	The department of the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_email	The email address of the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_first_name	The first name of the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_id	An internal identification number for the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_last_name	The last name of the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the host that responded to the session initiator.
responder_user_name	The user name of the user who last logged into the host that responded to the initiator of the aggregated sessions.
responder_user_phone	The phone number of the user who last logged into the host that responded to the initiator of the aggregated sessions.
security_zone_egress_name	The egress security zone in the connection event.
security_zone_ingress_name	The ingress security zone in the connection event.

Table 7-4 connection_summary Fields (continued)

Field	Description
sensor_address	The IP address of the managed device that generated the event. Format is *ipv4_address,ipv6_address*.
sensor_name	The name of the managed device that monitored the aggregated sessions.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
source_device	The identification of the source device, which is either: • the IP address of the NetFlow-enabled device that exported the data for the connection • FireSIGHT if the connection was detected by a Cisco managed device
start_time_sec	The UNIX timestamp of the date and time the five-minute interval used to aggregate the sessions in the summary started.

connection_summary Joins

The following table describes the joins you can perform using the **connection_summary** table.

Table 7-5 connection_summary Joins

You can join this table on...	And...
application_protocol_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id
initiator_ipaddr or responder_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

connection_summary Sample Query

The following query returns up to five connection event summary records detected by the selected device.

```
SELECT initiator_ipaddr, responder_ipaddr, protocol_name, application_protocol_id,
source_device, sensor_name, sensor_address, packets_recv, packets_sent, bytes_recv,
bytes_sent, connection_type, num_connections

FROM connection_summary

WHERE sensor_name='linden' limit 5;
```

si_connection_log

The **si_connection_log** table contains information on security intelligence events. The FireSIGHT System generates a Security Intelligence event when a connection is blacklisted or monitored by Security Intelligence; the event contains detailed information about the monitored traffic.

For more information, see the following sections:

- si_connection_log Fields, page 7-16
- si_connection_log Joins, page 7-26
- si_connection_log Sample Query, page 7-26

si_connection_log Fields

The following table describes the database fields you can access in the **si_connection_log** table.

Table 7-6 si_connection_log Fields

Field	Description
access_control_policy_name	The access control policy that contains the access control rule (or default action) that logged the connection.
access_control_policy_UUID	The UUID of the access control policy that contains the access control rule (or default action) that logged the connection.
access_control_reason	The reason that the access control rule logged the connection. One or more of the following: • IP Block • IP Monitor • User Bypass • File Monitor • File Block • Intrusion Monitor • Intrusion Block • File Resume Block • File Resume Allow • File Custom Detection • SSL Block • DNS Block • DNS Monitor • URL Block • URL Monitor • HTTP Injection • Intelligent App Bypass • blank if there is no connection logged
access_control_rule_action	The action associated with the access control rule (or default action): allow, block, and so on.
access_control_rule_id	An internal identification number for the rule.

Table 7-6 si_connection_log Fields (continued)

Field	Description
access_control_rule_name	The access control rule (or default action) that logged the connection.
application_protocol_id	An internal identification number of the application protocol.
application_protocol_name	One of: • the name of the application, if a positive identification can be made • unknown if the system cannot identify the server based on known server fingerprints • pending if the system requires more data • blank if there is no application information in the connection
bytes_recv	The total number of bytes transmitted by the session responder.
bytes_sent	Total number of bytes transmitted by the session initiator.
cert_valid_end_date	The Unix timestamp on which the SSL certificate used in the connection ceases to be valid.
cert_valid_start_date	The Unix timestamp when the SSL certificate used in the connection was issued.
client_application_id	An internal identification number for the client application that was used in the intrusion event.
client_application_name	The client application, if available, that was used in the intrusion event. One of: • the name of the application, if a positive identification can be made • a generic client name if the system detects a client application but cannot identify a specific one • blank if there is no client application information in the connection
client_application_version	The version of the client application.
connection_type	The detection source for the connection information. Either: • rna, if detected by a Cisco device • netflow, if exported by a NetFlow-enabled device
counter	Counter for the intrusion event associated with the connection event.
file_count	The number of files identified by snort in a session. A record is generated for each file identified in the session.
first_packet_sec	The UNIX timestamp of the date and time the first packet of the session was seen.
icmp_code	ICMP code if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.
icmp_type	ICMP type if the event is ICMP traffic, or null if the event was not generated from ICMP traffic.

Table 7-6 si_connection_log Fields (continued)

Field	Description
initiator_continent_name	The name of the continent of the host that initiated the session. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
initiator_country_id	Code for the country of the host that initiated the session.
initiator_country_name	Name of the country of the host that initiated the session.
initiator_ipaddr	A binary representation of the IP address of the host that initiated the session.
initiator_port	The port used by the session initiator.
initiator_user_dept	The department of the user who last logged into the initiator host.
initiator_user_email	The email address of the user who last logged into the initiator host.
initiator_user_first_name	The first name of the user who last logged into the initiator host.
initiator_user_id	An internal identification number for the user who last logged into the initiator host.
initiator_user_last_name	The last name of the user who last logged into the initiator host.
initiator_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the initiator host.
initiator_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the initiator host.
initiator_user_name	The user name of the user who last logged into the initiator host.
initiator_user_phone	The phone number of the user who last logged into the initiator host.
instance_id	Numerical ID of the Snort instance on the managed device that generated the event.
interface_egress_name	The ingress interface associated with the connection.
interface_ingress_name	The egress interface associated with the connection.
ioc_count	Number of indications of compromise found in the connection.
ips_event_count	The number of intrusion events generated in the connection prior to intrusion event thresholding.
last_packet_sec	The UNIX timestamp of the date and time the last packet of the session was seen.
monitor_rule_id_1	The ID of the first monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_1.

Table 7-6 si_connection_log Fields (continued)

Field	Description
monitor_rule_id_2	The ID of the second monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_2.
monitor_rule_id_3	The ID of the third monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_3.
monitor_rule_id_4	The ID of the fourth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_4.
monitor_rule_id_5	The ID of the fifth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_5.
monitor_rule_id_6	The ID of the sixth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_6.
monitor_rule_id_7	The ID of the seventh monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_7.
monitor_rule_id_8	The ID of the eighth monitor rule associated with the connection. This ID is associated with the name stored in monitor_rule_name_8.
monitor_rule_name_1	The name of the first monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_1.
monitor_rule_name_2	The name of the second monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_2.
monitor_rule_name_3	The name of the third monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_3.
monitor_rule_name_4	The name of the fourth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_4.
monitor_rule_name_5	The name of the fifth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_5.
monitor_rule_name_6	The name of the sixth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_6.
monitor_rule_name_7	The name of the seventh monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_7.
monitor_rule_name_8	The name of the eighth monitor rule associated with the connection. This name is associated with the ID stored in monitor_rule_id_8.
netbios_domain	The NetBIOS domain used in the connection.
netflow_dst_as	Netflow autonomous system number of the destination, either origin or peer.
netflow_dst_mask	Netflow destination address prefix mask.
netflow_dst_tos	Type of service from the IP header when packets are flowing from the destination to the source.
netflow_snmp_in	ID of the interface used by packets flowing from the source to the destination.
netflow_snmp_out	ID of the interface used by packets flowing from the destination to the source.
netflow_src_as	Netflow autonomous system number of the source, either origin or peer.

Table 7-6 si_connection_log Fields (continued)

Field	Description
netflow_src_mask	Netflow source address prefix mask.
netflow_src_tos	Type of service from the IP header when packets are flowing from the source to the destination.
network_analysis_policy_name	The network analysis policy associated with the intrusion policy that generated the intrusion event.
network_analysis_policy_UUID	The UUID of the network analysis policy associated with the intrusion policy that generated the intrusion event.
packets_recv	The total number of packets received by the host that initiated the session.
packets_sent	The total number of packets transmitted by the host that initiated the session.
protocol_name	The name of the protocol used in the connection.
protocol_num	The IANA number of the protocol as listed in http://www.iana.org/assignments/protocol-numbers.
responder_continent_name	The name of the continent of the host that responded to the session initiator. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
responder_country_id	Code for the country of the host that responded to the session initiator.
responder_country_name	Name of the country of the host that responded to the session initiator.
responder_ipaddr	A binary representation of the IPv4 or IPv6 address for the host that responded to the session initiator.
responder_port	The port used by the session responder.
responder_user_dept	The department of the user who last logged into the host that responded to the session initiator.
responder_user_email	The email address of the user who last logged into the host that responded to the session initiator.
responder_user_first_name	The first name of the user who last logged into the host that responded to the session initiator.
responder_user_id	An internal identification number for the user who last logged into the host that responded to the session initiator.
responder_user_last_name	The last name of the user who last logged into the host that responded to the session initiator.

Table 7-6 si_connection_log Fields (continued)

Field	Description
responder_user_last_seen_sec	The UNIX timestamp of the date and time the FireSIGHT System last detected user activity for the user who last logged into the host that responded to the session initiator.
responder_user_last_updated_sec	The UNIX timestamp of the date and time the FireSIGHT System last updated the user record for the user who last logged into the host that responded to the session initiator.
responder_user_name	The user name of the user who last logged into the host that responded to the session initiator.
responder_user_phone	The phone number of the user who last logged into the host that responded to the session initiator.
security_context	Description of the security context (virtual firewall) that the traffic passed through. Note that the system only populates this field for ASA FirePOWER devices in multi-context mode.
security_intelligence_category	The Security Intelligence category associated with the connection.
security_intelligence_ip	Whether the Security Intelligence-monitored IP address associated with the connection is a source IP (`src`) or destination IP (`dst`).
security_zone_egress_name	The egress security zone in the connection event.
security_zone_ingress_name	The ingress security zone in the connection event.
sensor_address	The IP address of the managed device that generated the event. Format is `ipv4 address,ipv6 address`.
sensor_name	The name of the managed device that monitored the session.
sensor_uuid	A unique identifier for the managed device, or `0` if `sensor_name` is `null`.
src_device_ipaddr	Either: • A binary representation of the IP address of the NetFlow-enabled device that exported the connection data • `0`, for connections detected by Cisco managed devices.
ssl_actual_action	The action performed on the connection based on the SSL Rule. This may differ from the expected action, as the action as specified in the rule may be impossible. Possible values include: • `'Unknown'` • `'Do Not Decrypt'` • `'Block'` • `'Block With Reset'` • `'Decrypt (Known Key)'` • `'Decrypt (Replace Key)'` • `'Decrypt (Resign)'`

Table 7-6 si_connection_log Fields (continued)

Field	Description
ssl_cipher_suite	Encryption suite used by the SSL connection. The value is stored in decimal format. See www.iana.org/assignments/tls-parameters/tls-parameters.xhtml for the cipher suite designated by the value.
ssl_expected_action	The action which should be performed on the connection based on the SSL Rule. Possible values include: • 'Unknown' • 'Do Not Decrypt' • 'Block' • 'Block With Reset' • 'Decrypt (Known Key)' • 'Decrypt (Replace Key)' • 'Decrypt (Resign)'
ssl_flow_flags	The debugging level flags for an encrypted connection. Possible values include: • 0x00000001 — NSE_FLOW__VALID — must be set for other fields to be valid • 0x00000002 — NSE_FLOW__INITIALIZED — internal structures ready for processing • 0x00000004 — NSE_FLOW__INTERCEPT — SSL session has been intercepted

Table 7-6 si_connection_log Fields (continued)

Field	Description
ssl_flow_messages	The messages exchanged between client and server during the SSL handshake. See http://tools.ietf.org/html/rfc5246 for more information. • 0x00000001 — NSE_MT__HELLO_REQUEST • 0x00000002 — NSE_MT__CLIENT_ALERT • 0x00000004 — NSE_MT__SERVER_ALERT • 0x00000008 — NSE_MT__CLIENT_HELLO • 0x00000010 — NSE_MT__SERVER_HELLO • 0x00000020 — NSE_MT__SERVER_CERTIFICATE • 0x00000040 — NSE_MT__SERVER_KEY_EXCHANGE • 0x00000080 — NSE_MT__CERTIFICATE_REQUEST • 0x00000100 — NSE_MT__SERVER_HELLO_DONE • 0x00000200 — NSE_MT__CLIENT_CERTIFICATE • 0x00000400 — NSE_MT__CLIENT_KEY_EXCHANGE • 0x00000800 — NSE_MT__CERTIFICATE_VERIFY • 0x00001000 — NSE_MT__CLIENT_CHANGE_CIPHER_SPEC • 0x00002000 — NSE_MT__CLIENT_FINISHED • 0x00004000 — NSE_MT__SERVER_CHANGE_CIPHER_SPEC • 0x00008000 — NSE_MT__SERVER_FINISHED • 0x00010000 — NSE_MT__NEW_SESSION_TICKET • 0x00020000 — NSE_MT__HANDSHAKE_OTHER • 0x00040000 — NSE_MT__APP_DATA_FROM_CLIENT • 0x00080000 — NSE_MT__APP_DATA_FROM_SERVER

Table 7-6 si_connection_log Fields (continued)

Field	Description
ssl_flow_status	Status of the SSL Flow. These values describe the reason behind the action taken or the error message seen. Possible values include: • 'Unknown' • 'No Match' • 'Success' • 'Uncached Session' • 'Unknown Cipher Suite' • 'Unsupported Cipher Suite' • 'Unsupported SSL Version' • 'SSL Compression Used' • 'Session Undecryptable in Passive Mode' • 'Handshake Error' • 'Decryption Error' • 'Pending Server Name Category Lookup' • 'Pending Common Name Category Lookup' • 'Internal Error' • 'Network Parameters Unavailable' • 'Invalid Server Certificate Handle' • 'Server Certificate Fingerprint Unavailable' • 'Cannot Cache Subject DN' • 'Cannot Cache Issuer DN' • 'Unknown SSL Version' • 'External Certificate List Unavailable' • 'External Certificate Fingerprint Unavailable' • 'Internal Certificate List Invalid' • 'Internal Certificate List Unavailable' • 'Internal Certificate Unavailable' • 'Internal Certificate Fingerprint Unavailable' • 'Server Certificate Validation Unavailable' • 'Server Certificate Validation Failure' • 'Invalid Action'
ssl_issuer_common_name	Issuer Common name from the SSL certificate. This is typically the host and domain name of the certificate issuer, but may contain other information.
ssl_issuer_country	The country of the SSL certificate issuer.

Table 7-6 si_connection_log Fields (continued)

Field	Description
ssl_issuer_organization	The organization of the SSL certificate issuer.
ssl_issuer_organization_unit	The organizational unit of the SSL certificate issuer.
ssl_policy_action	The default action configured for the policy when no rules match.
ssl_policy_name	ID number of the SSL policy that handled the connection.
ssl_policy_reason	The reason the SSL policy logged the SSL session.
ssl_rule_action	The action selected in the user interface for the SSL rule (`allow`, `block`, and so forth).
ssl_rule_name	ID number of the SSL rule or default action that handled the connection.
ssl_serial_number	The serial number of the SSL certificate, assigned by the issuing CA.
ssl_server_name	Name provided in the server name indication in the SSL Client Hello.
ssl_subject_common_name	Subject Common name from the SSL certificate. This is typically the host and domain name of the certificate subject, but may contain other information.
ssl_subject_country	The country of the SSL certificate subject.
ssl_subject_organization	The organization of the SSL certificate subject.
ssl_subject_organization_unit	The organizational unit of the SSL certificate subject.
ssl_url_category	Category of the flow as identified from the server name and certificate common name.
ssl_version	The SSL or TLS protocol version used to encrypt the connection.
tcp_flags	The TCP flags detected in the session.
url	The URL requested by the monitored host during the session, if available.
url_category	The category of the URL requested by the monitored host.
url_reputation	The reputation of the URL requested by the monitored host. One of the following: • `1` — High risk • `2` — Suspicious sites • `3` — Benign sites with security risks • `4` — Benign sites • `5` — Well known
web_application_id	An internal identification number for the web application.
web_application_name	One of: • the name of the application, if a positive identification can be made. • `web browsing` if the system detects an application protocol of HTTP but cannot identify a specific web application. • blank if the connection has no HTTP traffic.

si_connection_log Joins

The following table describes the joins you can perform using the **si_connection_log** table.

Table 7-7 si_connection_log Joins

You can join this table on...	And...
application_protocol_name or application_id or client_application_id or web_application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id
initiator_ipaddr or responder_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

si_connection_log Sample Query

The following query returns up to 25 connection event records from the **si_connection_log** table, sorted in descending order based on packet timestamps.

```
SELECT first_packet_sec, last_packet_sec, initiator_ipaddr, responder_ipaddr,
security_zone_ingress_name, security_zone_egress_name, initiator_port, protocol_name,
responder_port, application_protocol_id, client_application_id, web_application_id, url,
url_category, url_reputation

FROM si_connection_log

WHERE first_packet_sec <= UNIX_TIMESTAMP("2011-10-01 00:00:00") ORDER BY
first_packet_sec

DESC, last_packet_sec DESC LIMIT 0, 25;
```

Schema: User Activity Tables

This chapter contains information on the schema and supported joins for user activity and identity events. The FireSIGHT System can detect user activity on your network by tracking various types of user logins, including LDAP, POP3, IMAP, SMTP, AIM, and SIP.

For more information, see the sections listed in the following table.

Table 8-1 Schema for User Identity Tables

See...	For the table that stores information on...	Version
discovered_users, page 8-1	Information about the users detected by the system.	5.0+
user_discovery_event, page 8-2	User discovery events, which communicate the details of user activity on your network.	5.0+

discovered_users

The **discovered_users** table contains detailed information about each user detected by the system.

The **discovered_users** table supersedes the deprecated **rua_users** table starting with Version 5.0 of the FireSIGHT System.

For more information, see the following sections:

- discovered_users Fields, page 8-1
- discovered_users Joins, page 8-2
- discovered_users Sample Query, page 8-2

discovered_users Fields

The following table describes the fields you can access in the **discovered_users** table.

Table 8-2 discovered_users Fields

Field	Description
dept	The department of the user.
email	The email address for the user.
first_name	The first name for the user.

Table 8-2 discovered_users Fields (continued)

Field	Description
ip_address	This field has been deprecated and returns null for all queries.
ipaddr	A binary representation of the IPv4 or IPv6 address for the host where the user login was detected.
last_name	The last name for the user.
last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the user.
last_updated_sec	The UNIX timestamp of the date and time the user's information was last updated.
name	The name for the user.
phone	The phone number for the user.
rna_service	Field deprecated in Version 5.0. Returns null for all queries.
user_id	The internal identification number of the user who last logged onto the host.

discovered_users Joins

The following table describes the joins you can perform on the **rua_user** table.

Table 8-3 discovered_users Joins

You can left join on this field...	With other tables that have join type of...
user_id	user_discovery_event.user_id user_ipaddr_history.user_id

discovered_users Sample Query

The following query returns up to 25 discovered user records that were generated since a specified date and time.

```
SELECT user_id, ip_address, email, name, last_seen_sec, last_updated_sec
FROM discovered_users
WHERE last_seen_sec >= UNIX_TIMESTAMP("2011-10-01 00:00:00")
LIMIT 0, 25;
```

user_discovery_event

The **user_discovery_event** table contains a record for each user discovery event.

Note that starting in Version 5.0, the FireSIGHT System records the detection of user activity at the managed device level, no longer by detection engine. The detection_engine_name and detection_engine_uuid fields in this table have been replaced by the sensor_name and sensor_uuid fields respectively. Queries on these fields will return information about the managed device that generated the user discovery event.

For more information, see the following sections:

user_discovery_event Fields

The following table describes the fields you can access in the **user_discovery_event** table.

Table 8-4 user_discovery_event Fields

Field	Description
application_protocol_id	An internal identifier for the detected application protocol.
application_protocol_name	One of: • the name of the application used in the connection: LDAP, POP3, and so on • pending if the system cannot identify the application for one of several reasons • blank if there is no application information in the connection
description	The user name when the discovery event type is either Delete User Identity, or User Identity Dropped. Otherwise, blank.
event_id	An internal identification number for the discovery event.
event_time_sec	The UNIX timestamp of the date and time of the discovery event.
event_type	The type of discovery event. For example, New User Identity or User Login.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
ipaddr	A binary representation of the IP address of the host where the user activity was detected.
reported_by	The IPv4 address, IPv6 address, or NetBIOS name of the Active Directory server reporting a user login.
sensor_address	The IP address of the managed device that detected the user discovery event. Format is *ipv4_address,ipv6_address*.
sensor_name	The text name of the managed device that detected the user discovery event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
user_dept	The department of the user who last logged onto the host.
user_email	The email address of the user who last logged onto the host.
user_first_name	The first name of the user.
user_id	The internal identification number of the user who last logged onto the host.
user_last_name	The last name of the user.
user_last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the user.
user_last_updated_sec	The UNIX timestamp of the date and time the user's information was last updated.
user_name	The user name for the user who last logged onto the host.
user_phone	The phone number for the user who last logged onto the host.

user_discovery_event Joins

The following table describes the joins you can perform on the **user_discovery_event** table.

Table 8-5 user_discovery_event Joins

You can join this table on...	And...
ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr
user_id	discovered_users.user_id user_ipaddr_history.user_id

user_discovery_event Sample Query

The following query returns up to 25 user event records generated by a selected managed device since a particular date and time.

```
SELECT event_time_sec, ipaddr, sensor_name, event_type, user_name, user_last_seen_sec,
user_last_updated_sec

FROM user_discovery_event

WHERE sensor_name = sensor_name

AND user_last_seen_sec >= UNIX_TIMESTAMP("2011-10-01 00:00:00") ORDER BY event_type ASC

LIMIT 0, 25;
```

Schema: Correlation Tables

This chapter contains information on the schema and supported joins for correlation-related events, including remediation status and white list events. For more information, see the sections listed in the following table.

Table 9-1 Schema for Correlation Tables

See...	For the table that stores information on...	Version
compliance_event, page 9-1	Correlation events, which are generated when a correlation rule within an active correlation policy triggers.	4.10.x+
remediation_status, page 9-6	Remediation status events, which are generated when an active correlation policy triggers a remediation as a response.	4.10.x+
white_list_event, page 9-7	White list events, which are generated when the system detects a host out of compliance with a white list in an active white list compliance policy.	4.10.x+
white_list_violation, page 9-9	White list violations, which track the ways that the hosts on your network violate the compliance white lists in active compliance policies.	4.10.x+

compliance_event

The **`compliance_event`** table contains information about the correlation events that your Defense Center generates.

For more information, see the following sections:

- `compliance_event Fields, page 9-2`
- `compliance_event Joins, page 9-5`
- `compliance_event Sample Query, page 9-5`

compliance_event Fields

Keep in mind that many of the fields in the table can be blank, depending on what type of event triggered the correlation rule. For example, if the Defense Center generates a correlation event because the system detects a specific application protocol or web application running on a specific port, that correlation event does not include intrusion-related information. Fields in this table can also be blank depending on your FireSIGHT System configuration. For example, if you do not have a Control license, correlation events do not include user identity information.

Note that starting in Version 5.0, the FireSIGHT System records the detection of network and user activity at the managed device level, rather than by detection engine. The `detection_engine_name` and `detection_engine_uuid` fields in the **`compliance_event`** table now return only blanks, and queries that join on those fields return zero records. You must query on the `sensor_uuid` field instead of `detection_engine_uuid` for information about the location of an event's detection.

The following table describes the fields you can access in the **`compliance_event`** table.

Table 9-2 compliance_event Fields

Field	Description
`blocked`	Value indicating what happened to the packet that triggered the intrusion event: • `0` — Packet not dropped • `1` — Packet dropped (inline, switched, or routed deployments) • `2` — Packet that triggered the event would have been dropped, if the intrusion policy had been applied to a device in an inline, switched, or routed deployment
`description`	Information about the correlation event and how it was triggered.
`detection_engine_name`	Field deprecated in Version 5.0. Returns `null` for all queries.
`detection_engine_uuid`	Field deprecated in Version 5.0. Returns `null` for all queries.
`dst_host_criticality`	The user-assigned host criticality of the destination host involved in the correlation event: `None`, `Low`, `Medium`, or `High`.
`dst_host_type`	The destination host type: `Host`, `Router`, `Bridge`, `NAT Device`, or `Load Balancer`.
`dst_ip_address`	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to `null`, but it is not reliable.
`dst_ip_address_v6`	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to `null`, but it is not reliable.
`dst_ipaddr`	A binary representation of the IPv4 or IPv6 address for the destination host involved in the triggering event.
`dst_os_product`	The operating system name on the destination host.
`dst_os_vendor`	The operating system's vendor on the destination host.
`dst_os_version`	The operating system's version number on the destination host.
`dst_port`	The port number for the host receiving the traffic if the event protocol type is TCP or UDP. The ICMP code if the protocol type is ICMP.

Table 9-2 compliance_event Fields (continued)

Field	Description
dst_rna_service	If identified, the application protocol on the source host that is associated with the triggering event. If not identified, one of the following: • none or blank - no application protocol traffic • unknown - the server cannot be identified based on known server fingerprints • pending - the system needs more information
dst_user_dept	The department of the destination user.
dst_user_email	The email address of the destination user.
dst_user_first_name	The first name of the destination user.
dst_user_id	The internal identification number for the destination user; that is, the user who last logged into the destination host before the event occurred.
dst_user_last_name	The last name of the destination user.
dst_user_last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the destination user.
dst_user_last_updated_sec	The UNIX timestamp of the date and time the destination user's information was last updated.
dst_user_name	The user name for the destination user.
dst_user_phone	The destination user's phone number.
dst_vlan_id	The destination host's VLAN identification number, if applicable.
event_id	The identification number of the triggering intrusion event generated by the device.
event_time_sec	The UNIX timestamp of the date and time of the triggering event.
event_time_usec	The microsecond increment of the triggering event timestamp.
event_type	The type of underlying event that triggered the correlation rule or caused the Defense Center to generate the correlation event. Values are: • ids, for intrusion event triggers • rna, for discovery event, host input event, connection event, or traffic profile change triggers • rua, for user discovery event triggers • whitelist, for compliance white list violation triggers
host_event_type	The event type, for example, New Host or Identity Conflict.
id	An internal identification number for the correlation event.
impact	The impact flag value of the event. Values are: • 1 — Red (vulnerable) • 2 — Orange (potentially vulnerable) • 3 — Yellow (currently not vulnerable) • 4 — Blue (unknown target) • 5 — Gray (unknown impact) Set only when the correlation rule was triggered by an intrusion event.

Table 9-2 compliance_event Fields (continued)

Field	Description
interface_egress_name	The ingress interface associated with the connection.
interface_ingress_name	The egress interface associated with the connection.
policy_name	The correlation policy that was violated.
policy_rule_name	The correlation rule that triggered the policy violation.
policy_rule_uuid	A unique identifier for the correlation rule.
policy_time_sec	The UNIX timestamp of the date and time the correlation event was generated.
policy_uuid	A unique identifier for the correlation policy.
priority	The priority for the correlation event, which is set in the user interface. The event priority is determined by the priority of either the triggered rule or the violated correlation policy.
protocol_name	The protocol associated with the event, if available.
protocol_num	The IANA-specified protocol number, if available.
rna_event_type	Field deprecated in Version 5.0. Returns null for all queries.
rua_event_type	Field deprecated in Version 5.0. Returns null for all queries.
rule_generator_id	The generator ID number (GID) of the component that generated the triggering intrusion event.
rule_message	Explanatory text about the intrusion event that triggered the correlation rule. For rule-based events, the message is generated from the rule. For decoder- and preprocessor-based events, the message is hard coded.
rule_signature_id	The signature ID (SID) for the event. Identifies the specific rule or rules, decoder message, or preprocessor message that caused the triggering intrusion event to be generated.
security_zone_egress_name	The egress security zone in the correlation event.
security_zone_ingress_name	The ingress security zone in the correlation event.
sensor_address	The IP address of the managed device that generated the underlying event that triggered the compliance event. Format is *ipv4_address,ipv6_address*.
sensor_name	The managed device that generated the underlying event that triggered the compliance event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
src_host_criticality	The user-assigned host criticality of the source host involved in the compliance event: None, Low, Medium, or High.
src_host_type	The source host type: Host, Router, Bridge, NAT Device, or Load Balancer.
src_ip_address	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
src_ip_address_v6	Field deprecated in Version 5.2. Due to backwards compatibility the value in this field is not set to null, but it is not reliable.
src_ipaddr	A binary representation of the IPv4 or IPv6 address for the source host involved in the triggering event.
src_os_product	The operating system's name on the source host.
src_os_vendor	The operating system's vendor on the source host.

Table 9-2 compliance_event Fields (continued)

Field	Description
src_os_version	The operating system's version number on the source host.
src_port	The port number on the source host. For ICMP traffic, the ICMP type appears instead.
src_rna_service	If identified, the application protocol on the source host that is associated with the triggering event. If not identified, one of the following: • none or blank - no application protocol traffic • unknown - the server and application protocol cannot be identified based on known server fingerprints • pending - the system needs more information
src_user_dept	The department of the source user.
src_user_email	The email address of the source user.
src_user_first_name	The first name of the source user.
src_user_id	The internal identification number for the source user; that is, the user who last logged into the source host before the event occurred.
src_user_last_name	The last name of the source user.
src_user_last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the source user.
src_user_last_updated_sec	The UNIX timestamp of the date and time the source user's information was last updated.
src_user_name	The login user name for the source user.
src_user_phone	The source user's phone number.
src_vlan_id	The source host's VLAN identification number, if applicable.
user_event_type	The type of triggering user event, for example, New User Identity or User Login.

compliance_event Joins

The following table describes the joins you can perform on the **compliance_event** table.

Table 9-3 compliance_event Joins

You can join this table on...	And...
dst_ipaddr or src_ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

compliance_event Sample Query

The following query returns up to 25 correlation event records from a week, with event information such as the event time, source and destination IP addresses, source and destination ports, policy information, and so on.

```
SELECT event_id, policy_time_sec, impact, blocked, src_ipaddr, dst_ipaddr, src_port,
dst_port, description, policy_name, policy_rule_name, priority, src_host_criticality,
dst_host_criticality, security_zone_egress_name, security_zone_ingress_name,
sensor_name, interface_egress_name, interface_ingress_name

FROM compliance_event WHERE event_type!="whitelist"

AND policy_time_sec

BETWEEN UNIX_TIMESTAMP("2011-10-01 00:00:00")

AND UNIX_TIMESTAMP("2011-10-07 23:59:59")

ORDER BY policy_time_sec

DESC LIMIT 0, 25;
```

remediation_status

The **remediation_status** table contains information about remediation events, which are generated when the Defense Center launches a remediation in response to a correlation policy violation.

For more information, see the following sections:

- remediation_status Fields, page 9-6
- remediation_status Joins, page 9-7
- remediation_status Sample Query, page 9-7

remediation_status Fields

The following table describes the database fields you can access in the **remediation_status** table.

Table 9-4 remediation_status Fields

Field	Description
id	The identification number of the policy that was violated and triggered the remediation.
policy_name	The correlation policy that was violated and triggered the remediation.
policy_rule_name	The specific correlation rule that triggered the remediation.
policy_rule_uuid	A unique identifier for the correlation rule.
policy_time_sec	The UNIX timestamp of the date and time that the correlation event that triggered the remediation was generated.
policy_uuid	A unique identifier for the correlation policy that triggered the correlation event.
remediation_name	The remediation that was launched.
remediation_time_sec	The UNIX timestamp of the date and time the Defense Center launched the remediation.
status_text	A message that describes what happened when the remediation was launched, such as "successful completion of remediation."

remediation_status Joins

You cannot perform joins on the **remediation_status** table.

remediation_status Sample Query

The following query returns up to 25 records generated before a given date. These records include remediation status information such as the remediation timestamp, the status message, and so on.

```
SELECT policy_time_sec, remediation_time_sec, remediation_name, policy_name,
policy_rule_name, status_text

FROM remediation_status WHERE remediation_time_sec <= UNIX_TIMESTAMP("2011-10-01
00:00:00")

ORDER BY policy_time_sec

DESC LIMIT 0, 25;
```

white_list_event

The **white_list_event** table contains white list events that are generated when the system detects a host not compliant with a white list in an active white list compliance policy.

Note that starting in Version 5.0, the FireSIGHT System records the detection of network and user activity at the managed device level, no longer by detection engine. The `detection_engine_name` and `detection_engine_uuid` fields in the **white_list_event** table now return only `null`, and queries that join on those fields return zero records. Querying on the `sensor_uuid` field instead of `detection_engine_uuid` provides the equivalent information.

For more information, see the following sections:

- white_list_event Fields, page 9-7
- white_list_event Joins, page 9-9
- white_list_event Sample Query, page 9-9

white_list_event Fields

The following table describes the database fields you can access in the **white_list_event** table.

Table 9-5 white_list_event Fields

Field	Description
description	A description of how the white list was violated.
detection_engine_name	Field deprecated in Version 5.0. Returns null for all queries.
detection_engine_uuid	Field deprecated in Version 5.0. Returns null for all queries.

Table 9-5 white_list_event Fields (continued)

Field	Description
host_criticality	The user-assigned criticality of the host that is out of compliance with the white list: None, Low, Medium, or High.
host_type	The host type: Host, Router, Bridge, NAT Device, or Load Balancer.
id	An internal unique identifier for the white list event.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.
ip_address_v6	Field deprecated in Version 5.2. Returns null for all queries.
ipaddr	A binary representation of the IP address of the non-compliant host.
os_product	The operating system's product name.
os_vendor	The operating system's vendor.
os_version	The operating system's version number.
policy_name	The violated compliance policy that includes the white list.
policy_time_sec	The UNIX timestamp of the date and time the event was generated.
policy_uuid	A unique identifier for the compliance policy that includes the white list event.
port	The port, if any, associated with the event that triggered a service white list violation (that is, when a violation occurs as a result of a non-compliant service). For other types of white list violations, the field is blank.
priority	The priority for the white list event, which is set in the user interface.
protocol_name	The protocol associated with the event, if available.
protocol_num	The IANA-specified protocol number, if available.
rna_service	The service that triggered the white list violation, if available.
sensor_address	IP address of the managed device that detected the traffic. Format is *ipv4_address,ipv6_address*.
sensor_name	The device that generated the white list event.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
user_dept	The department of the user.
user_email	The email address for the user.
user_first_name	The first name for the user.
user_id	Internal identification number of the user who last logged into the host before the event occurred.
user_last_name	The last name for the user.
user_last_seen_sec	The UNIX timestamp of the date and time the system last reported a login for the user.
user_last_updated_sec	The UNIX timestamp of the date and time the user's information was last updated.
user_name	The login user name for the user.
user_phone	The phone number for the user.
vlan_id	The VLAN identification number, if applicable.
white_list_name	The white list that was violated.
white_list_uuid	A unique identifier for the white list.

white_list_event Joins

The following table describes the joins you can perform on the **white_list_event** table.

Table 9-6 white_list_event Joins

You can join this table on...	And...
ipaddr	rna_host_ip_map.ipaddr user_ipaddr_history.ipaddr

white_list_event Sample Query

The following query returns up to 25 records generated before a specified time. The records include white list event information such as the compliance policy name, timestamp the event was generated, white list name, and so on.

```
SELECT policy_name, policy_time_sec, ipaddr, user_name, port, description,
white_list_name, priority, host_criticality, sensor_name

FROM white_list_event WHERE policy_time_sec <= UNIX_TIMESTAMP("2011-10-01 00:00:00")

ORDER BY policy_time_sec DESC LIMIT 0, 25;
```

white_list_violation

The **white_list_violation** table track compliance white list violations, which track the ways that the hosts on your network violate the compliance white lists in active compliance policies.

For more information, see the following sections:

- white_list_violation Fields, page 9-9
- white_list_violation Joins, page 9-10
- white_list_violation Sample Query, page 9-10

white_list_violation Fields

The following table describes the database fields you can access in the **white_list_violation** table.

Table 9-7 white_list_violation Fields

Field	Description
host_id	ID number of the host in violation of the white list.
info	Any available vendor, product, or version information associated with the white list violation. For protocols that violate a white list, the field also indicates whether the violation is due to a network or transport protocol.
ip_address	Field deprecated in Version 5.2. Returns null for all queries.

Table 9-7 white_list_violation Fields (continued)

Field	Description
port	The port, if any, associated with the event that triggered a service white list violation (that is, when a violation occurs as a result of a non-compliant service). For other types of white list violations, the field is blank.
protocol_name	The protocol associated with the event.
type	The type of white list violation, indicating whether the violation occurred due to a non-compliant: • operating system (os) • service (service) • client application (client app) • protocol (protocol)
violation_time_sec	The UNIX timestamp of the date and time the violation was logged.
white_list_name	The white list that was violated.
white_list_uuid	A unique identifier for the white list.

white_list_violation Joins

You cannot perform joins on the **white_list_violation** table.

white_list_violation Sample Query

The following query returns up to 25 records with white list violation information such as the host IP address violating the white list, the violated white list name, and the count of violations.

```
SELECT host_id, white_list_name, count(*)
FROM white_list_violation
GROUP BY white_list_name, host_id
ORDER BY white_list_name
DESC LIMIT 0, 25;
```

Schema: File Event Tables

This chapter contains information on the schema and supported joins for file events. For more information, see the section listed in the following table.

Table 10-1 Schema for File Event Tables

See...	For the table that stores information on...	Version
file_event, page 10-1	File events generated when file transfers are detected in the monitored network.	5.1.1+

While the following tables are available, Cisco does not currently support lookups on them:

- `file_categories`
- `file_rules`
- `file_types`
- `file_type_rule_map`
- `file_type_category_map`

file_event

The **`file_event`** table contains information about the file events that your Defense Center generates. A new file event is generated each time a file transfer is detected on the monitored network.

For more information, see the following sections:

- `file_event Fields, page 10-1`
- `file_event Joins, page 10-5`
- `file_event Sample Query, page 10-6`

file_event Fields

The **`file_event`** table contains information on files that are detected passing through the monitored network. Each file event can be correlated with a connection event. Details of the file and file transfer are recorded, including the name, size, source, destination, and direction of the file, a SHA256 hash of the file, the device that detected the file, and whether it is considered to be malware.

Table 10-2 file_event Fields

Field	Description
action	The action taken on the file based on the file type. Can have the following values: • 1 — Detect • 2 — Block • 3 — Malware Cloud Lookup • 4 — Malware Block • 5 — Malware Whitelist • 6 — Cloud Lookup Timeout
application_id	ID number that maps to the application using the file transfer.
application_name	One of the following: • the name of the application used in the connection • pending or unknown if the system cannot identify the application • blank if there is no application information in the connection
archived	Indicates whether the file has been archived.
cert_valid_end_date	The Unix timestamp on which the SSL certificate used in the connection ceases to be valid.
cert_valid_start_date	The Unix timestamp when the SSL certificate used in the connection was issued.
client_application_id	The internal identification number for the client application, if applicable.
client_application_name	The name of the client application, if applicable.
connection_sec	UNIX timestamp (seconds since 00:00:00 01/01/1970) of the connection event associated with the file event.
counter	Specific counter for the event, used to distinguish among multiple events that happened during the same second.
direction	Whether the file was uploaded or downloaded. Currently the value depends entirely on the protocol (for example, if the connection is HTTP it is a download).
disposition	The malware status of the file. Possible values include: • CLEAN — The file is clean and does not contain malware. • UNKNOWN — It is unknown whether the file contains malware. • MALWARE — The file contains malware. • UNAVAILABLE — The software was unable to send a request to the Cisco cloud for a disposition, or the Cisco cloud services did not respond to the request. • CUSTOM SIGNATURE — The file matches a user-defined hash, and is treated in a fashion designated by the user.

Table 10-2 file_event Fields (continued)

Field	Description
dst_continent_name	The name of the continent of the destination host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica
dst_country_id	Code for the country of the destination host.
dst_country_name	Name of the country of the destination host.
dst_ip_address_v6	Field deprecated in Version 5.2. Returns null for all queries.
dst_ipaddr	A binary representation of the IP address of the destination host involved in the triggering event.
dst_port	Port number for the destination of the connection.
event_description	The additional event information associated with the event type.
event_id	Event identification number.
file_name	Name of the detected file. This name can contain UTF-8 characters.
file_sha	SHA256 hash of the file.
file_size	Size of the detected file in bytes.
file_type	The file type of the detected or quarantined file.
file_type_category	Description of the file category.
file_type_category_id	Numeric identifier for the file category.
file_type_id	ID number that maps to the file type.
instance_id	Numerical ID of the Snort instance on the managed device that generated the event.
policy_uuid	Identification number that acts as a unique identifier for the access control policy that triggered the event.

Table 10-2 file_event Fields (continued)

Field	Description
sandboxed	Indicates whether the file was sent for dynamic analysis. Possible values are: • Sent for Analysis • Failed to Send • File Size is Too Small • File Size is Too Large • Sent for Analysis • Analysis Complete • Failure (Network Issue) • Failure (Rate Limit) • Failure (File Too Large) • Failure (File Read Error) • Failure (Internal Library Error) • File Not Sent, Disposition Unavailable • Failure (Cannot Run File) • Failure (Analysis Timeout) • File Not Supported
score	A numeric value from 0 to 100 based on the potentially malicious behaviors observed during dynamic analysis.
security_context	Description of the security context (virtual firewall) that the traffic passed through. Note that the system only populates this field for ASA FirePOWER devices in multi-context mode.
sensor_address	A binary representation of the IP address of the device that provided the event.
sensor_id	ID for the device that provided the event.
sensor_name	The text name of the managed device that generated the event record. This field is null when the event refers to the reporting device itself, rather than to a connected device.
sensor_uuid	A unique identifier for the managed device, or 0 if sensor_name is null.
signature_processed	Indicated whether the file's signature was processed.
src_continent_name	The name of the continent of the source host. ** — Unknown na — North America as — Asia af — Africa eu — Europe sa — South America au — Australia an — Antarctica

Table 10-2 file_event Fields (continued)

Field	Description
src_country_id	Code for the country of the source host.
src_country_name	Name of the country of the source host.
src_ip_address_v6	Field deprecated in Version 5.2. Returns null for all queries.
src_ipaddr	A binary representation of the IPv4 or IPv6 address of the source host involved in the triggering event.
src_port	Port number for the source of the connection.
ssl_issuer_common_name	Issuer Common Name from the SSL certificate. This is typically the host and domain name of the certificate issuer, but may contain other information.
ssl_issuer_country	The country of the SSL certificate issuer.
ssl_issuer_organization	The organization of the SSL certificate issuer.
ssl_issuer_organization_unit	The organizational unit of the SSL certificate issuer.
ssl_serial_number	The serial number of the SSL certificate, assigned by the issuing CA.
ssl_subject_common_name	Subject Common name from the SSL certificate. This is typically the host and domain name of the certificate subject, but may contain other information.
ssl_subject_country	The country of the SSL certificate subject.
ssl_subject_organization	The organization of the SSL certificate subject.
ssl_subject_organization_unit	The organizational unit of the SSL certificate subject.
storage	The storage status of the file. Possible values are: • File Stored • Unable to Store File • File Size is Too Large • File Size is Too Small • Unable to Store File • File Not Stored, Disposition Unavailable
threat_name	Name of the threat.
timestamp	UNIX timestamp when enough of the file has been transmitted to identify the file type.
url	URL of the file source.
user_id	The internal identification number for the destination user; that is, the user who last logged into the destination host before the event occurred.
username	Name associated with the user_id.
web_application_id	The internal identification number for the web application, if applicable.
web_application_name	Name of the web application, if applicable.

file_event Joins

The following table describes the joins you can perform on the **file_event** table.

Table 10-3 file_event Joins

You can join this table on...	And...
application_id	application_info.application_id application_host_map.application_id application_tag_map.application_id rna_host_service_info.application_protocol_id rna_host_client_app_payload.web_application_id rna_host_client_app_payload.client_application_id rna_host_client_app.client_application_id rna_host_client_app.application_protocol_id rna_host_service_payload.web_application_id

file_event Sample Query

The following query returns up to 10 file events with the application name, connection information, and file name, where the disposition is not CLEAN.

```
SELECT file_event.application_name, file_event.connection_sec, file_event.counter,
file_event.file_name

FROM file_event

WHERE file_event.disposition != 'CLEAN' limit 10;
```

Deprecated Tables

This appendix contains information on tables which were used in previous releases and are now deprecated. Although you can still query these tables, the values in the fields may not be correct, and in most cases are `null`. There are no supported joins for these tables.

Table A-1 Deprecated Tables

Table	Superseded by	Last Used Version
application_ip_map	application_host_map, page 6-5	5.1.1
rna_ip_host	rna_host, page 6-12	5.1.1
rna_ip_host_attribute	rna_host_attribute, page 6-14	5.1.1
rna_ip_host_client_app	rna_host_client_app, page 6-15	5.1.1
rna_ip_host_client_app_payload	rna_host_client_app_payload, page 6-18	5.1.1
rna_ip_host_os	rna_host_os, page 6-28	5.1.1
rna_ip_host_os_vulns	rna_host_os_vulns, page 6-29	5.1.1
rna_ip_host_sensor	rna_host_sensor, page 6-32	5.1.1
rna_ip_host_service	rna_host_service, page 6-34	5.1.1
rna_ip_host_service_banner	rna_host_service_banner, page 6-36	5.1.1
rna_ip_host_service_info	rna_host_service_info, page 6-37	5.1.1
rna_ip_host_service_payload	rna_host_service_payload, page 6-42	5.1.1
rna_ip_host_service_subtype	rna_host_service_subtype, page 6-45	5.1.1
rna_ip_host_service_vulns	rna_host_service_vulns, page 6-46	5.1.1
rna_ip_host_third_party_vuln	rna_host_third_party_vuln, page 6-47	5.1.1
rna_ip_host_third_party_vuln_bugtraq_id	rna_host_third_party_vuln_bugtraq_id, page 6-49	5.1.1
rna_ip_host_third_party_vuln_cve_id	rna_host_third_party_vuln_cve_id, page 6-50	5.1.1
rna_ip_host_third_party_vuln_rna_id	rna_host_third_party_vuln_rna_id, page 6-52	5.1.1
rna_ip_host_user_history	user_ipaddr_history, page 6-59	5.1.1
rna_mac_host	rna_host_mac_map, page 6-27	5.1.1

Table A-1 Deprecated Tables

Table	Superseded by	Last Used Version
rna_mac_host_sensor	rna_host_mac_map, page 6-27	5.1.1
rna_mac_ip_map	rna_host_ip_map, page 6-25 rna_host_mac_map, page 6-27	5.1.1

A

C

D

F

H

I

N

P

R

S

T

U

W