L'installation 5.1.0 intelligente de satellite de gestionnaire de logiciel (SSM) échoue au noyau basé par KVM

Contenu

Introduction

Ce document décrit la solution au problème qui se pose quand l'installation 5.1.0 intelligente de satellite de gestionnaire de logiciel (SSM) échoue au noyau basé de clavier/vidéo/souris (KVM) qui inclut la plate-forme de service en nuage de Cisco.

Problème

L'installation se termine par l'intermédiaire de la console et l'interface utilisateur (UI) est accessible.

Au moment de la procédure d'installation d'enregistrement CSSM, on le note que l'enregistrement échoue tandis que l'enregistrement de réseau, aussi bien que l'enregistrement manuel, est exécuté. La version de chat est validée, le noyau et Java Virtual Machine (JVM) dans le système basé par KVM. prenez la note que JVM exécute 1.8.0_102-b14 et noyau 3.10.0-514.el7. Rivalisez avec l'installation basée par ESXI, où le noyau exécute 3.10.0-862.14.4.el7 et JVM 1.8.0_191-b12.

```
[root@satellite bin]# ./version.sh
Using CATALINA_BASE: /opt/tc
Using CATALINA_HOME: /opt/tc
Using CATALINA_TMPDIR: /opt/tc/temp
Using JRE_HOME: /
Using CLASSPATH: /opt/tc/bin/bootstrap.jar:/opt/tc/bin/tomcat-juli.jar
Using CATALINA_PID: /opt/tomcat/temp/tomcat.pid
Server version: Apache Tomcat/9.0.1
Server built: Sep 27 2017 17:31:52 UTC
Server number: 9.0.1.0
OS Name: Linux
OS Version: 3.10.0-514.el7.x86_64
Architecture: amd64
JVM Version: 1.8.0_102-b14
JVM Vendor: Oracle Corporation
```

Composants

Plate-forme : KVM a basé le noyau

Logiciel : Image ISO du classique 5.1

Solution

Étape 1. Naviguez vers **cd/opt/tomcat/logs/**.

Étape 2. **Les logs** ouverts **catalina.out** et trouvent l'exception qui a lieu au moment de la procédure d'enregistrement avec CSSM.

Le fournisseur IAIK-JCE IAIK est une extension de chiffrement de Javas qui a un ensemble d'API et peut implémenter la fonctionnalité cryptographique. Il est utilisé afin de prendre en charge des fonctionnalités supplémentaires de Sécurité au JDK. Le module LCS ne génère pas la paire de clés pour le fichier de demande CSR dû à l'indisponibilité du fichier jar IAIK.

```
2019-05-15 20:35:01,604 [http-nio-8080-exec-9] INFO controller.LindosController - Invoked GET
/lcsSetupStatus
2019-05-15 20:35:01,606 [http-nio-8080-exec-9] INFO controller.LindosController - LCS Setup
Status = 0
2019-05-15 23:53:12,226 [http-nio-8080-exec-10] INFO controller.LindosController - Invoked GET
/lcsSetupStatus
2019-05-15 23:53:12,230 [http-nio-8080-exec-10] INFO controller.LindosController - LCS Setup
Status = 0
2019-05-15 23:53:12,241 [http-nio-8080-exec-1] INFO controller.LindosController - Invoked
/lcsSetup
2019-05-15 23:53:12,243 [http-nio-8080-exec-1] DEBUG controller.LindosController - Setup Status
= 0 (0=empty, 1=key/CSR generated, 2=Signer certs installed)
2019-05-15 23:53:12,243 [http-nio-8080-exec-1] DEBUG controller.LindosController - First time
setup invoked (ID element not present in JSON). CN=5fc62a80-59a0-0137-54ab-023a01ab3207
2019-05-15 23:53:12,243 [http-nio-8080-exec-1] DEBUG domain.LcsSignerSetup - In LcsSignerSetup
2019-05-15 23:53:12,244 [http-nio-8080-exec-1] DEBUG domain.LcsSignerSetup - Generating Key
Pair...
2019-05-15 23:53:12,244 [http-nio-8080-exec-1] ERROR error.RestResponseEntityExceptionHandler -
java.security.NoSuchProviderException: no such provider: IAIK
com.cisco.ias.lindos.data.domain.LcsSetupException: java.security.NoSuchProviderException: no
such provider: IAIK
at com.cisco.ias.lindos.data.domain.LcsSignerSetup.<init>(LcsSignerSetup.java:50)
at com.cisco.ias.lindos.web.controller.LindosController.setupLcs(LindosController.java:126)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at
org.springframework.web.method.support.InvocableHandlerMethod.invoke(InvocableHandlerMethod.java
:215)
at
org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerM
ethod.java:132)
at
org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHan
dle(ServletInvocableHandlerMethod.java:104)
at
org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandleM
ethod(RequestMappingHandlerAdapter.java:749)
at
org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInterna
l(RequestMappingHandlerAdapter.java:690)
at
org.springframework.web.servlet.mvc.method.AbstractHandlerMethodAdapter.handle(AbstractHandlerMe
```

```
thodAdapter.java:83)
at org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:945)
at org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:876)
at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:961)
at org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:863)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:660)
at org.springframework.web.servlet.FrameworkServlet.service(FrameworkServlet.java:837)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:741)
at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:231
)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:193
)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:199)
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:140)
at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:81)
at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:651)
at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:87)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:342)
at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:500)
at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:66)
at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:754)
at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1376)
at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
at java.lang.Thread.run(Thread.java:745)
2019-05-15 23:53:12,254 [http-nio-8080-exec-2] INFO controller.LindosController - Invoked GET
/lcsSetupStatus
2019-05-15 23:53:12,256 [http-nio-8080-exec-2] INFO controller.LindosController - LCS Setup
Status = 0
```

Étape 3. Placez le fournisseur requis de Sécurité dans le classpath ; **cp /opt/tomcat/webapps/Lindos/WEB-INF/lib/iaik_jce-5.1.jar /usr/lib/jvm/java/jre/lib/ext/**.

Étape 4. Assurez-vous que le pot est accessible en lecture par d'autres modules ; **chmod o+r /usr/lib/jvm/java/jre/lib/ext/iaik_jce-5.1.jar**.

Étape 5. Enregistrez le **chemin de fichier java.security à une** variable de temp ; **java_security=/usr/lib/jvm/java/jre/lib/security/java.security.**

Étape 6. Priorité existante de fournisseurs d'incrément par ; **Perl - pi - e 's/^security.provider.) (\ d+/« security.provider. » . ($1+1)/e $java_security.**

Étape 7. Insérez IAIK comme premier fournisseur dans la liste (notez la barre oblique inverse qui échappe au saut de ligne) ; **sed - I '/security.provider.2/i **

security.provider.1=iaik.security.provider.IAIK $java_security.

Étape 8. Redémarrez le chat pour des modifications afin de le prendre effet avec la commande ; **chat de reprise de systemctl**.

Étape 9. Enregistrez le satellite avec CSSM et quand l'enregistrement dans le satellite est terminé, l'UI ne redémarrera pas.

Étape 10. Pliez les deux Certificats x509 utilisés pour des connexions de Transport Layer Security (TLS) sur les ports 443 et 8443 afin de rencontrer le format de l'email amélioré par intimité (PEM) ; **pli - W 64 /drbd/certs/rails_ssl.crt > && système mv /drbd/certs/rails_ssl_folded.crt /drbd/certs/rails_ssl.crt de /drbd/certs/rails_ssl_folded.crt**

pli - W 64 /drbd/certs/pi_ssl.crt > && système mv /drbd/certs/pi_ssl_folded.crt /drbd/certs/pi_ssl.crt de /drbd/certs/pi_ssl_folded.crt.

> Remarque: N'exécutez pas ces commandes se plient aussi bien que ligne différente d'installation comme ils corrompent le CERT PEM 64-encoded.

Nginx de l'étape 11.Start ; **nginx de début de systemctl**.

> Remarque: Si l'UI ne monte pas après une synchronisation, alors il est dû à ces CERT étant mis à jour/remplacés. Par conséquent, étapes 8-10 devront être répétées.

Après que vous suiviez ces étapes, accédez à l'UI et vous pouvez voir que synchronisation de courrier avec CSSM et enregistrement final est succès.

Vous pouvez voir l'inventaire et le permis sectionner le permis tracé du VA. Vous pouvez enregistrer des exemples intelligents de produit au satellite.