Troubleshoot Incomplete Diagnostics.sh Script Execution

Contents

Introduction
Prerequisites
Requirements
Components Used
Background Information
Problem
Solution

Introduction

This document describes the procedure to troubleshoot incomplete diagnostics.sh script execution in Cisco Policy Suite (CPS).

Contributed by Ullas Kumar E, Cisco TAC Engineer.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Linux
- CPS

Note: Cisco recommends that you must have root access privileges to CPS CLI.

Components Used

The information in this document is based on these software and hardware versions:

- CPS 21.1
- Centos 8.0
- Unified Computing System (UCS)-B

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Diagnostics.sh is the basic troubleshooting command that can be executed in pcrfclient or installer node of CPS to verify the current status of the system.

It provides a detailed list of parameters as part of the health check of CPS.

This script runs against the various access, monitoring, and configuration points of a running CPS system.

In High Availability(HA) or Geo-Redundant (GR) environments, the script always does a ping check for all Virtual Machines (VMs) prior to any other checks and adds any that fail the ping test to the IGNORED_HOSTS variable. This helps reduce the possibility of script function errors.

```
Examples:
 /var/qps/bin/diag/diagnostics.sh -q
 /var/qps/bin/diag/diagnostics.sh --basic_ports --clock_skew
```

These are the prominent checks this script does.

```
--basic_ports : Run basic port checks
 For AIO: 80, 11211, 27017, 27749, 7070, 8080, 8090, 8182, 9091, 9092
 For HA/GR: 80, 11211, 7070, 8080, 8081, 8090, 8182, 9091, 9092, and Mongo DB ports based on /etc/broadh
 --clock_skew : Check clock skew between lb01 and all vms (Multi-Node Environment only)
 --diskspace : Check diskspace
 --get_active_alarms : Get the active alarms in the CPS
 --get_frag_status : Get fragmentation status for Primary members of DBs viz. session_cache, sk_cache, d
 --get_replica_status : Get the status of the replica-sets present in environment. (Multi-Node Environme
 --get_shard_health : Get the status of the sharded database information present in environment. (Multi-
 --get_sharding_status : Get the status of the sharding information present in environment. (Multi-Node
 --get_session_shard_health : Get the session shard health status information present in environment. (M
 --get_peer_status : Get the diameter peer information present in environment. (Multi-Node Environment o
 --get_sharded_replica_status : Get the status of the shards present in environment. (Multi-Node Environ
 --ha_proxy : Connect to HAProxy to check operation and performance statistics, and ports (Multi-Node En
      http://lbvip01:5540/haproxy?stats
      http://lbvip01:5540//haproxy-diam?stats
 --help -h : Help - displays this help
 --hostnames : Check hostnames are valid (no underscores, resolvable, in /etc/broadhop/servers) (AIO onl
 --ignored_hosts : Ignore the comma separated list of hosts. For example --ignored_hosts='portal01,porta
      Default is 'portal01,portal02,portallb01,portallb02' (Multi-Node Environment only)
 --ping_check : Check ping status for all VM
 --policy_revision_status : Check the policy revision status on all QNS,LB,UDC VMs.
 --lwr_diagnostics : Retrieve diagnostics from CPS LWR kafka processes
 --qns_diagnostics : Retrieve diagnostics from CPS java processes
 --qns_login : Check qns user passwordless login
 --quiet -q : Quiet output - display only failed diagnostics
 --radius : Run radius specific checks
 --redis : Run redis specific checks
 --whisper : Run whisper specific checks
 --aido : Run Aido specific checks
 --svn : Check svn sync status between pcrfclient01 & pcrfclient02 (Multi-Node Environment only)
 --tacacs : Check Tacacs server reachability
 --swapspace : Check swap space
 --verbose -v : Verbose output - display *all* diagnostics (by default, some are grouped for readability
 --virtual_ips : Ensure Virtual IP Addresses are operational (Multi-Node Environment only)
 --vm_allocation : Ensure VM Memory and CPUs have been allocated according to recommendations
```

Problem

It is possible that in some situations, the execution of diagnostics.sh scripts hung at one point and it cannot move further or finish the script execution.

You can execute the script and observe the script is stuck at "checking forAuto Intelligent DB Operations (AIDO) Status" does not proceed for Subversion Number (SVN) check and further.

<#root>

```
[root@installer ~]#
```

`diagnostics.sh`

```
CPS Diagnostics HA Multi-Node Environment
---------------------------
Ping check for all VMs...
Hosts that are not 'pingable' are added to the IGNORED_HOSTS variable...[PASS]
Checking basic ports for all VMs...[PASS]
Checking qns passwordless logins for all VMs...[PASS]
Validating hostnames...[PASS]
Checking disk space for all VMs...[PASS]
Checking swap space for all VMs...[PASS]
Checking for clock skew for all VMs...[PASS]
Retrieving diagnostics from pcrfclient01:9045...[PASS]
Retrieving diagnostics from pcrfclient02:9045...[PASS]
Checking redis server instances status on lb01...[PASS]
Checking redis server instances status on lb02...[PASS]
Checking whisper status on all VMs...[PASS]
Checking AIDO status on all VMs...[PASS]
.
.
```

When you check the verbose output of diagnostics.sh, there is a step to check SVN status, the script doesn't move further from there. It is indicating that the diagnostics.sh script stuck at the facter check.

<#root>

```
[[32mPASS[0m] AIDO Pass
[[ -f /var/tmp/aido_extra_info ]]
cat /var/tmp/aido_extra_info
There is no provision to check AIDO service status of installer from this host
/bin/rm -fr /var/tmp/aido_extra_info
check_all_svn
++ is_enabled true
++ [[ '' == \t\r\u\e ]]
++ [[ true != \f\a\l\s\e ]]
++ echo true
[[ true == \t\r\u\e ]]
++ awk '{$1=""; $2=""; print}'
```

`++ /usr/bin/ssh root@pcrfclient01 -o ConnectTimeout=2 /usr/bin/facter.`

```
++ grep svn_slave_list
```

The script logs into pcrfclient01 and checks the **svn_slave_list** from the **factor** command output, which is not completely executed.

Also, you can log in to pcrfcleint01 and check if the factor command runs properly and gives desired output.

<#root>

```
[root@pcrfclient01 ]#
```

```
facter | grep eth
```

```
[root@installer ~]# ^C
```

When you check the load average of pcrfclient01, it is observed as very high.

<#root>

```
[root@pcrfclient01 pacemaker]#
```

```
top
```

```
top - 15:34:18 up 289 days, 14:55, 1 user, load average:
```

```
2094.68, 2091.77, 2086.36
```

Verify the facter-related processes are stuck and result in a high load average.

<#root>

```
[root@pcrfclient01 ~]#
```

```
ps -ef | grep facter | wc -l
```

```
2096
```

Solution

The ultimate solution to clear these stuck processes and reduce the load average is to reboot the pcrfclient01 VM. The procedure to clear stuck processes of facter and resolve hung issue of diagnostics.sh execution:

Step 1. Log in to the pcrfclient node and execute reboot command.

<#root>

```
[root@pcrfclient01 ~]#
```

```
init 6
```

Step 2. Verify the pcrfcleitn01 VM is Up and stable.

<#root>

```
[root@pcrfclient01 ~]#
```

`uptime`

```
10:07:15 up 1 min, 4:09, 1 user, load average: 0.33, 0.33, 0.36
[root@pcrfclient01 ~]#
```

Step 3. Verify the load average of the pcrfclient01 is normal.

<#root>

```
[root@instapcrfclient01ller ~]#
```

`top`

```
top - 10:07:55 up 1 min, 4:10, 1 user, load average: 0.24, 0.31, 0.35
```

Step 4. Run diagnostics.sh and verify the script execution is finished.

<#root>

```
[root@instapcrfclient01ller ~]#
```

`diagnostics.sh`