在CPS中从错误状态恢复CRD的过程

目录

简介

本文档介绍从BAD状态恢复思科策略套件(CPS)自定义参考数据(CRD)表的过程。

先决条件

要求

Cisco 建议您了解以下主题：

- Linux
- CPS
- MongoDB

思科建议您必须具有权限访问：

- 对CPS CLI的根访问
- “qns-svn”用户对CPS GUI（策略生成器和CPS中心）的访问

使用的组件

本文档中的信息基于以下软件和硬件版本：

- CPS 20.2
- MongoDB v3.6.17
- UCS-B

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

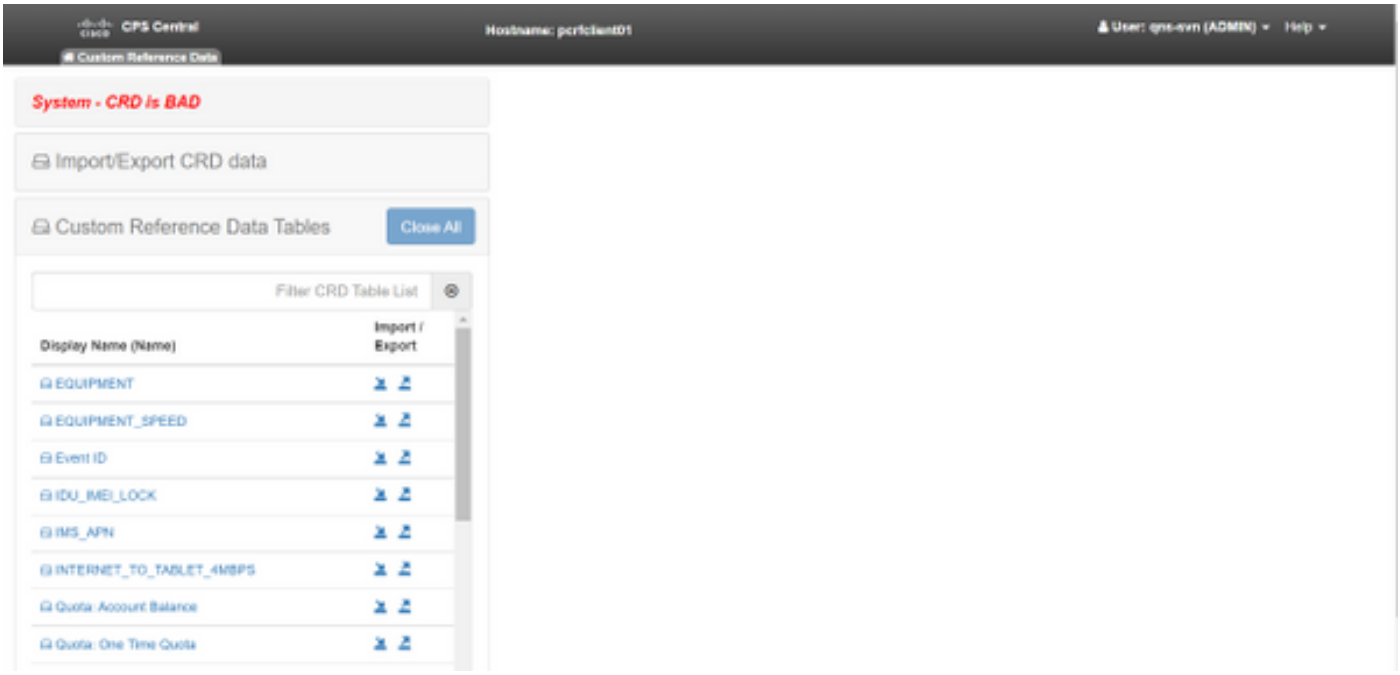
背景信息

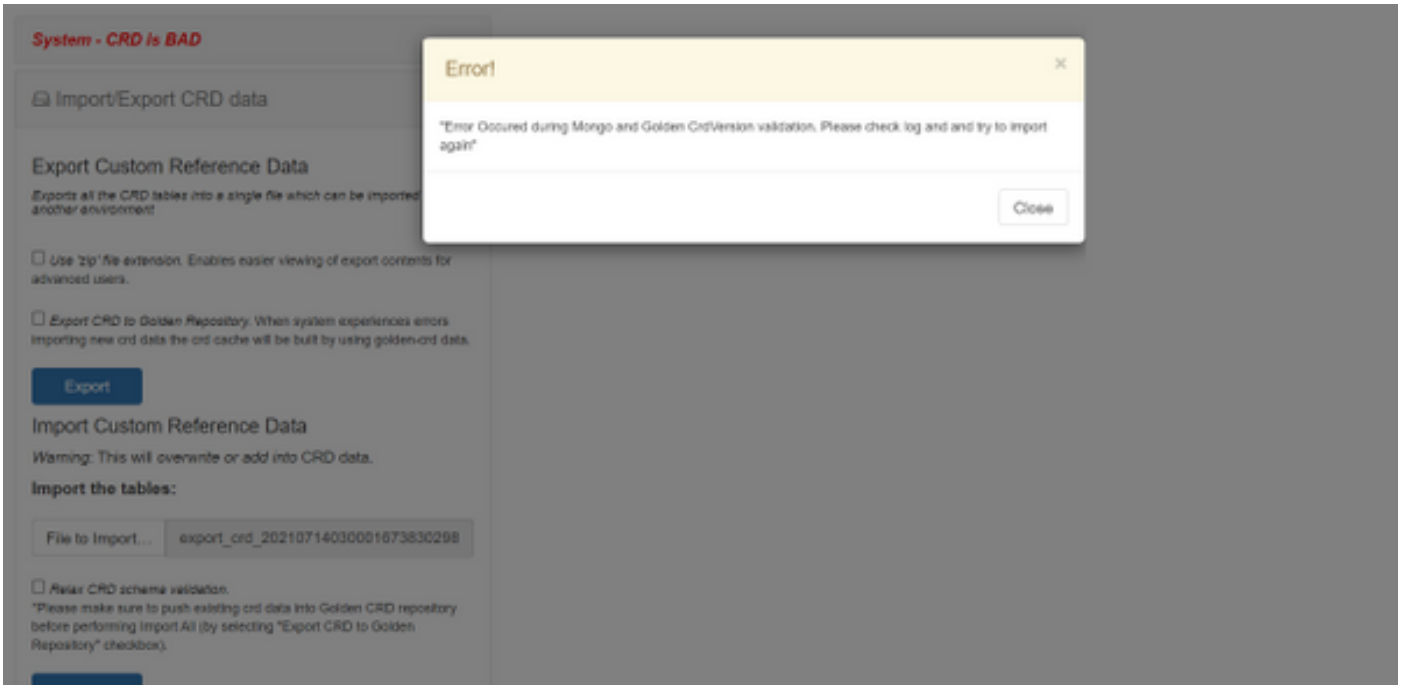
在CPS中，CRD表用于存储自定义策略配置信息，这些信息从策略生成器发布，并与CRD DB关联，该CRD DB存在于托管于会话管理器上的MongoDB实例中。在CRD表中通过CPS Central GUI执行导出和导入操作，以处理CRD表数据。

问题

如果在执行导入所有操作时出现任何错误，则CPS会停止进程，将系统设置为BAD状态并阻止CRD API的执行。CPS向客户端发送错误响应，表明系统处于BAD状态。如果系统处于BAD状态，并且您重新启动Quantum Network Suite(QNS)/用户数据通道(UDC)服务器，则使用Golden-crd数据构建CRD缓存。如果系统BAD状态为FALSE，则使用MongoDB构建CRD缓存。

以下是CPS中心错误图像供参考。





如果CRD系统坏，则：

1. CRD操作被阻止。您只能查看数据。
2. CRD API（_import_all、_list、_query除外）被阻止。
3. QNS重新启动从黄金卡位置提取CRD数据。
4. QNS/UDC的重新启动既不修复系统BAD状态也不修复呼叫丢弃，它只从golden-crd构建CRD缓存。
5. 使用黄金线数据构建的CRD缓存。如果系统BAD状态为FALSE，则使用MongoDB构建crd缓存。

以下是CPS qns.log中的关联消息：

```
qns02 qns02 2021-07-29 11:16:50,820 [pool-50847-thread-1]
INFO c.b.c.i.e.ApplicationInterceptor - System -
CRD is in bad state. All CRD APIs (except import all, list and query),
are blocked and user is not allowed to use.
Please verify your crd schema/crd data and try again!
qns02 qns02 2021-07-28 11:33:59,788 [pool-50847-thread-1]
WARN c.b.c.i.CustomerReferenceDataManager -
System is in BAD state. Data will be fetched from svn golden-crd repository.
qns01 qns01 2021-07-28 11:55:24,256 [pool-50847-thread-1]
WARN c.b.c.i.e.ApplicationInterceptor - ApplicationInterceptor: Is system bad: true
```

从BAD状态恢复CRD的过程

方法1.

要清除系统状态，您需要从策略生成器导入有效且正确的CRD架构，该架构涉及从CPS中心导入有效CRD数据，如果导入全部成功，则会清除系统状态，并且所有CRD API和操作都会解除阻止。

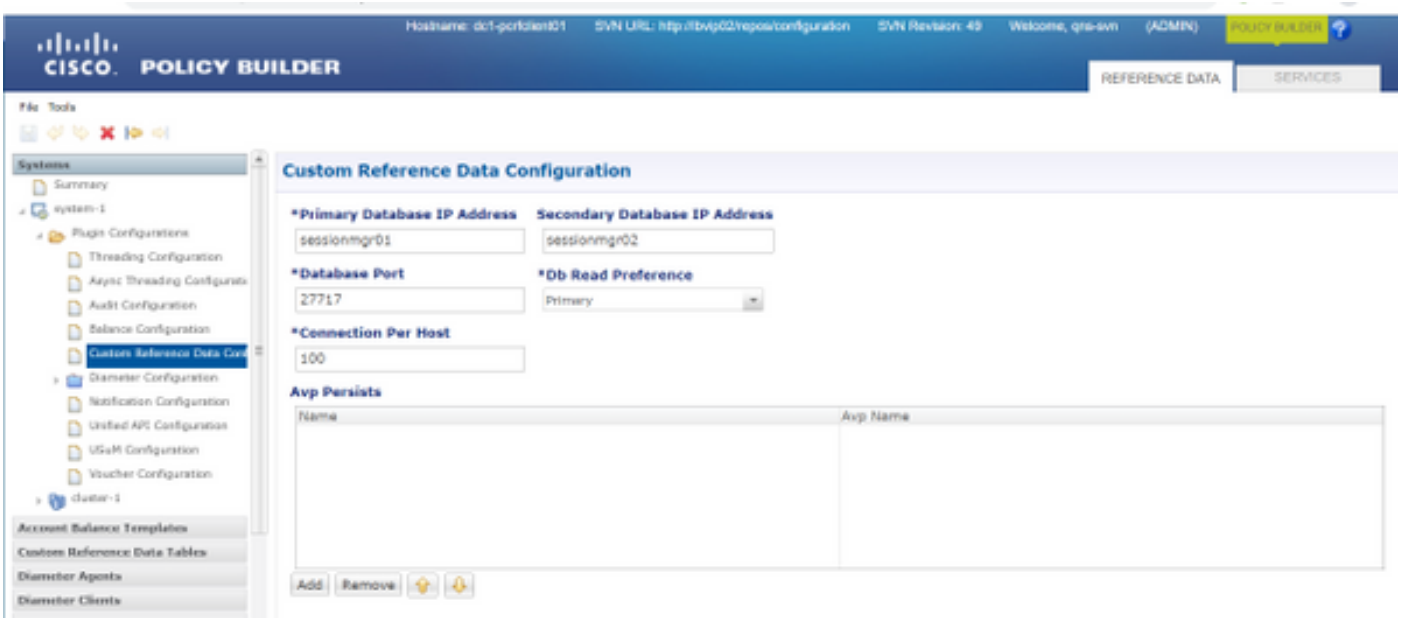
以下是详细步骤：

步骤1.将此命令运行到备份CRD数据库。

```
Command template:
#mongodump --host <session_manager> --port <cust_ref_data_port>
--db cust_ref_data -o cust_ref_data_backup

Sample command:
#mongodump --host sessionmgr01 --port 27717 --db cust_ref_data -o cust_ref_data_backup
```

注意：对于CRD DB主机和端口，请参阅PB中的自定义参考数据配置，如本图所示。



步骤2.使用此过程删除CRD表（整个DB）。

步骤2.1.登录到存在CRD数据库的mongo实例。

```
Command template:
#mongo --host <sessionmgrXX> --port <cust_ref_data_port>

Sample command:
#mongo --host sessionmgr01 --port 27717
```

步骤2.2.运行此命令以显示mongo实例中存在的所有数据库。

```
set01:PRIMARY> show dbs
admin 0.031GB
config 0.031GB
cust_ref_data 0.125GB
local 5.029GB
session_cache 0.031GB
sk_cache 0.031GB
set01:PRIMARY>
```

步骤2.3.运行此命令以切换到CRD数据库。

```
set01:PRIMARY> use cust_ref_data
switched to db cust_ref_data
set01:PRIMARY
```

步骤2.4.运行此命令以删除CRD数据库。

```
set01:PRIMARY> db.dropDatabase()
{
"dropped" : "cust_ref_data",
"ok" : 1,
"operationTime" : Timestamp(1631074286, 13),
"$clusterTime" : {
"clusterTime" : Timestamp(1631074286, 13),
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
```

```
"keyId" : NumberLong(0)
}}}
set01:PRIMARY>
```

步骤3.使用命令show dbs检验是否存在名为cust_ref_data的**db**。

```
set01:PRIMARY> show dbs
admin 0.031GB
config 0.031GB
local 5.029GB
session_cache 0.031GB
sk_cache 0.031GB
set01:PRIMARY>
```

步骤4.使用“qns-svn”用户登录策略生成器并发布有效的CRD架构。

步骤5.从Cluster Manager中，在所有具有restartall.sh的节**点上重新启**动qns进程。

步骤6.检验诊断是否正常，并且CRD表中没有条目。CRD表中必须只存在架构，即没有任何数据。

步骤7.使用“qns-svn”用户登录CPS中心并导入有效的CRD数据。

步骤8.验证是否导入所有返回成功消息和CPS中心中未显示“system - CRD is BAD”错误消息。

步骤9.确认所有CRD API现在都已解除阻止，您现在可以处理CRD数据。

如果第一种方法不起作用，则选择第二种方法。

方法2.

步骤1.使用命令diagnostics.sh —get_r确定ADMIN DB Mongo实例托管在**其中的主机和端口**。

```
[root@installer ~]# diagnostics.sh --get_r
CPS Diagnostics HA Multi-Node Environment
--------------------------
Checking replica sets...
|-------------------------------------------------------------------------------------------------
----------------------------------------|
| Mongo:v3.6.17 MONGODB REPLICA-SETS STATUS INFORMATION Date : 2021-09-14 02:56:23 |
|-------------------------------------------------------------------------------------------------
----------------------------------------|
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC - PRIORITY |
|-------------------------------------------------------------------------------------------------
----------------------------------------|
| ADMIN:set06 |
| Status via arbitervip:27721 sessionmgr01:27721 sessionmgr02:27721 |
| Member-1 - 27721 : - PRIMARY - sessionmgr01 - ON-LINE - -------- - 3 |
| Member-2 - 27721 : - SECONDARY - sessionmgr02 - ON-LINE - 1 sec - 2 |
| Member-3 - 27721 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
|-------------------------------------------------------------------------------------------------
----------------------------------------|
```

步骤2.登录到存在管理数据库的mongo实例。

```
Command template:
#mongo --host <sessionmgrXX> --port <Admin_DB__port>

Sample Command:
#mongo --host sessionmgr01 --port 27721
```

步骤3.运行此命令以显示mongo实例中存在的所有数据库。

```
set06:PRIMARY> show dbs
admin 0.078GB
config 0.078GB
diameter 0.078GB
keystore 0.078GB
local 4.076GB
policy_trace 2.078GB
queueing 0.078GB
scheduler 0.078GB
sharding 0.078GB
set06:PRIMARY>
```

步骤4.运行此命令以切换到ADMIN DB。

```
set06:PRIMARY> use admin
switched to db admin
set06:PRIMARY>
```

步骤5.运行此命令以显示ADMIN DB中存在的所有表。

```
set06:PRIMARY> show tables
state
system.indexes
system.keys
system.version
set06:PRIMARY>
```

步骤6.运行此命令以检查系统的当前状态。

```
set06:PRIMARY> db.state.find()
{ "_id" : "state", "isSystemBad" : true, "lastUpdatedDate" : ISODate("2021-08-11T15:01:13.313Z")
}
set06:PRIMARY>
```

在这里，您可以看到“**isSystemBad**”：**没错**。因此，您必须使用下一步中提供的命令将此字段更新为**false**以清除CRD BAD状态。

步骤7.使用命令db.state.updateOne({_id:"state"},{$set:{isSystemBad:false}}更新字段**“isSystemBAD”**。

```
set06:PRIMARY> db.state.updateOne({_id:"state"},{$set:{isSystemBad:false}})
{ "acknowledged" : true, "matchedCount" : 0, "modifiedCount" : 0 }
set06:PRIMARY>
```

步骤8.运行命令**db.state.find()**，以检查**isSystemBad字**段值是否更改为**false。**

```
set06:PRIMARY> db.state.find()
```

```
{ "_id" : "state", "isSystemBad" : false, "lastUpdatedDate" : ISODate("2021-08-
11T15:01:13.313Z") }
set06:PRIMARY>
```

步骤9.验证所有CRD API现在都已解除阻止，您现在可以处理CRD数据。