DCAP CENTRAL

jSONAR Simplifying Security

tech note 1803

Ingesting Apache Ranger Entitlement Data into jSonar's DCAP Central

DCAP Central can consume data related to Apache Ranger policies. This Technote will outline the steps necessary to integrate the two systems.

SYSTEM INTEGRATION:

Integration of Ranger entitlement data is done using a "pull" where DCAP Central uses Ranger REST APIs to pull JSON documents that are parsed and processed using SonarGateway. Ranger policy data is further processed by DCAP Central pipelines.

STEPS TO INTEGRATE THE SYSTEMS:

1. SSH into the jSonar environment
2. Add support for Ranger entitlement records by:
   a. vi /etc/rsyslog.d/sonargateway.conf and write the line $IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/ranger.conf

```
$MaxMessageSize 32768
$IncludeConfig /etc/rsyslog.d/sonar/gateway/include/*.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/cassandra.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/cloudera.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/cloudwatch_aurora.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/cosmosdb.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/eventhub.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/helloworld.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/horton.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/kafka_consumer.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/mapr.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/mssql.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/mssql_eventhub.conf
$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/okta.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/oracle.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/qradar.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/redjack.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/service_now.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/sonar_audit_log.conf
$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/ranger.conf
#$IncludeConfig /etc/rsyslog.d/sonar/gateway/rulesets/windows.conf
```

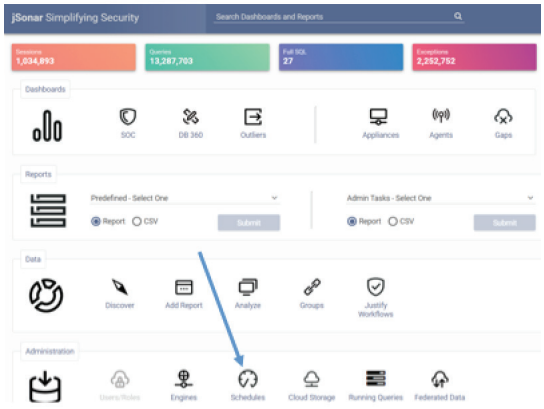
3. Get port number from rulesets configuration file.
   a. vi /etc/rsyslog.d/sonar/gateway/rulesets/ranger.conf

```
input(type="imptcp" port="15068" keepalive="on" ruleset="ranger")

template(name="ranger_rawmsg" type="list") {
    property(name="rawmsg")
    constant(value="D3L1M1T3R")
}

ruleset(name="ranger") {
    action(type="omprog"
           binary="/usr/lib/sonar/gateway/sonargateway --config /etc/sonar/gateway/ranger.json"
           template="ranger_rawmsg")
    stop
}
```

4. Configure the dispatcher web service section for Apache Ranger.
   a. vi /opt/sonarfinder/sonarFinder/dispatcher.conf
   b. add lines for Ranger configuration
      i. [ranger]
      ii. download_type = JSON
      iii. Username = your ranger username
      iv. Password = your ranger password
      v. SonarGateway_Address =<IP of Sonar machine>
      vi. SonarGateway_Port =<port from ranger.conf>
   c. example configuration below

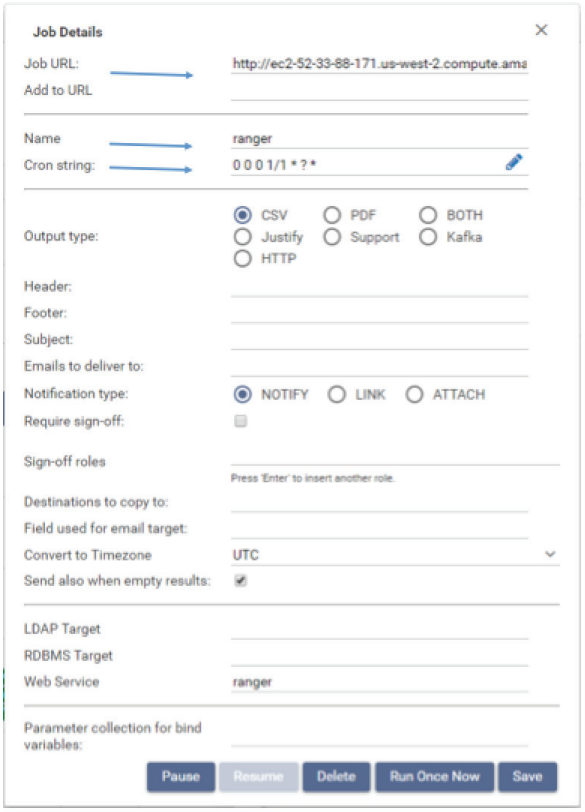
```
[ranger]
download_type = JSON
username = admin
password = admin
#authURL =
SonarGateway_Address = 192.168.50.205
SonarGateway_Port = 15068
#service_date_format = %Y-%m-%dT%H:%M:%SZ
#apikey =
#api_paging_pattern
```

5. Restart sonardispatcher and rsyslog
   a. systemctl restart sonardispatcher rsyslog
6. In the SonarG UI, schedule the ranger
   a. Click schedules



   b. Click anywhere that is not an existing job to open the scheduler
   c. Configure for ranger
      i. Enter the job URL: http://<your-host>:<your-port>/service/plugins/policies/download/<your-instance-name>
      ii. Enter the Name: ranger
      iii. Enter desired cron string. The suggested interval is daily.
      iv. Enter Web Service: ranger
      v. Click save

7. To test that the pull is working
   a. SSH to the jSonar environment
      i. sudo tail -f /opt/sonarfinder/sonarFinder/dispatcher.log
   b. In the GUI, open the ranger job in schedules that was just created
      i. Click Run Once Now at the bottom of the job



   c. Check the tail of the dispatcher.log. The process should run with no errors, example below

8. Now you should see ranger data in DCAP Central in the ranger collection in the sonargd database.
9. After getting the ranger data in DCAP Central the data is in the Ranger format. The final step is to convert the Ranger data using a pipeline that generates documents that are easier to use and aggregate.
10. Schedule the Ranger conversion pipeline to run after the ranger webservices pull to populate converted data to the working collection. Typically if the policy pull is done at 1am once a day the conversion pipeline should be scheduled to run at 2am once a day. The ranger conversion pipeline is called ranger_policy_extract and is published by the user lmrm__ae.

Final output:


```
{
  "_id": {
    "$oid": "5adf6718e138231ca3258e24"
  },
  "update_time": {
    "$date": "2018-04-11T19:31:46.000Z"
  },
  "create_time": {
    "$date": "2018-04-11T19:15:35.000Z"
  },
  "description": "Policy for all - path",
  "id": 1,
  "isauditenabled": true,
  "isenabled": true,
  "policies_name": "all - path",
  "isallowed": true,
  "type": "read",
  "delegate_admin": true,
  "username": "hdfs",
  "isexcludes": false,
  "isrecursive": true,
  "path": "/*",
  "service": "uryranger_hadoop",
  "version": 1,
  "as_of": {
    "$date": "2018-04-26T00:00:00.000Z"
  }
}
```


Note that the final data maintains history so that you can now what policy was in-effect at what time allowing you also to perform historical analysis of the data.