Phone:
+1 703-483-6383

Fax:
+1 703-991-5341

Email:
support@controlcase.com

Web:
www.controlcase.com

ControlCase Plugin Development

Edition 1.3

Plugin Development Methodology

Plugin development is a necessary step in implementing L&M for a device or application. Output of the plugin can be fed to the SIEM tool for monitoring and alerting purpose. ControlCase has out-of-the-box plugins available for popular devices and applications. For custom applications plugins need to be developed from scratch to make it compatible with unknown type and format of logs. Plugin development has multiple phases: parser development, signature identification and Define correlations

Parser Development

Step1: Identify unique logs

1. Collect sample logs of the device/application, below are the example of sample logs.

```
Mar  1 00:01:01 192.168.10.1 2017: 03:01-02:30:05 [redacted]sg135 ulogd[4824]: id="2001" severity="info"
sys="SecureNet" sub="packetfilter" name="Packet dropped" action="drop" fwrule="60001" initf="eth0"
mark="0x3441" app="1089" srcmac="08:94:ef:2d:94:87" dstmac="00:1a:8c:4b:4e:e8" srcip="192.168.12.10"
dstip="192.168.12.255" proto="17" length="78" tos="0x00" prec="0x00" ttl="128" srcport="137" dstport="137"
Mar  1 00:01:01 192.168.10.1 2017: 03:01-02:30:33 [redacted]5 httpproxy[23755]: id="0001"
severity="info" sys="SecureWeb" sub="http" name="http access" action="pass" method="POST"
srcip="192.168.10.201" dstip="195.88.18.14" user="" group="" ad_domain="" statuscode="200" cached="0"
profile="REF_DefaultHTTPProfile (Default Web Filter Profile)" filteraction="REF_HttCffNewinterna (new-internal)"
size="22" request="0xdd3bf000" url="http://api.silverstreet.com/send.php" referer="" error="" authtime="0"
dnstime="717" cattime="191" avscantime="4966" fullreqtime="460147" device="0" auth="0" ua="Apache-
HttpClient/4.3.6 (java 1.5)" exceptions="" sandbox="-" content-type="application/octet-stream"
Mar  1 22:36:00 192.168.10.1 2017: 03:02-01:06:00 [redacted]smtpd[9276]: SCANNER[9276]:
id="1000" severity="info" sys="SecureMail" sub="smtp" name="email passed" srcip="127.0.0.1" from="do-not-
reply@fw-notify.net" to="[redacted]" subject="[r[redacted]g135][INFO-723] Daily Executive
Report" queueid="1cj7hQ-0002Pc-Fg" size="422540"
```

2. Find out unique logs from the device/application sample logs.
    - Unique logs can be retrieved by using a unique number within a log or a unique keyword.
    - E.g. below samples has **name="Packet dropped", name="http access"** etc., so value of that variable name is a unique identifier.

Mar 1 00:01:01 192.168.10.1 2017: 03:01-02:30:05 [redacted]-sg135 ulogd[4824]: id="2001" severity="info" sys="SecureNet" sub="packetfilter" name="Packet dropped" action="drop" fwrule="60001" initf="eth0" mark="0x3441" app="1089" srcmac="08:94:ef:2d:94:87" dstmac="00:1a:8c:4b:4e:e8" srcip="192.168.12.10" dstip="192.168.12.255" proto="17" length="78" tos="0x00" prec="0x00" ttl="128" srcport="137" dstport="137"
Mar 1 00:01:01 192.168.10.1 2017: 03:01-02:30:33 [redacted] httpproxy[23755]: id="0001" severity="info" sys="SecureWeb" sub="http" name="http access" action="pass" method="POST" srcip="192.168.10.201" dstip="195.88.18.14" user="" group="" ad_domain="" statuscode="200" cached="0" profile="REF_DefaultHTTPProfile (Default Web Filter Profile)" filteraction="REF_HttCffNewinterna (new-internal)" size="22" request="0xdd3bf000" url="http://api.silverstreet.com/send.php" referer="" error="" authtime="0" dnstime="717" cattime="191" avscantime="4966" fullreqtime="460147" device="0" auth="0" ua="Apache-HttpClient/4.3.6 (java 1.5)" exceptions="" sandbox="-" content-type="application/octet-stream"
Mar 1 22:36:00 192.168.10.1 2017: 03:02-01:06:00 [redacted] mtpd[9276]: SCANNER[9276]: id="1000" severity="info" sys="SecureMail" sub="smtp" name="email passed" srcip="127.0.0.1" from="do-not-reply@fw-notify.net" to="[redacted]" subject="[redacted]135][INFO-723] Daily Executive Report" queueid="1cj7hQ-0002Pc-Fg" size="422540"

3. Once we find out the unique logs, we need to find how many unique patterns are required to parse all logs.
4. After finding the unique patterns, next step is to create **regular expression** to capture those logs.

Step 2: Write regular expression

1. Capture the fields from the unique logs which are essential with respect to compliance such as login success/failure, access to card holder data table etc.
   You can get help from below web sites on how to write regular expression and test it:
   https://pythex.org/
   https://regex101.com/
2. Give each regex a **Rule No** so that it becomes easy in identifying which logs are being parsed through which rule.
3. Keep the no of regex as less as possible, as it increases the parsing time and may affect the performance.
4. There must be a **generic regular expression** which can be at the end of the plugin file for each plugin in case if there are some logs which are not being captured by any of the **regex** then it will be parsed through this generic rule
5. The regular expression should be able to capture the below information:

   - **username:** Username from which the action was performed
   - **date:** Date of the logs when they were generated
   - **device:** Capture the device name of the logs
   - **plugin_sid:** Capture the signature id of the log which must be unique
   - **src_ip:** Every log should have a valid source IP address. Make sure you use resolve function for this. If the action is performed on the same host then source IP address is the IP address of the device/host IP address

- **dst_ip:** Every log should have a valid destination IP address. DO NOT RESOLVE DST_IP. If the action is performed on the same host then destination IP address can be set to 0.0.0.0
- **src_port:** If source port is present parse it by resolve function
- **dst_port:** If destination port is present, parse it by resolve function
- **userdata1 to userdata9:** in case something custom needs to be captured.

Step 3: Create plugin file

- Please refer plugin_template.cfg for template
- Once done with making regular expression for all the logs start with the plugin by creating <devicename>.cfg file in any text editor. Replace <devicename> with proper name of the device or application
    - Find the below image for reference:

Step 3.a: Create plugin with translation:

- See attached translation.cfg for template
- Create a plugin with translation only when the plugin_sid captured is character and needs to be translated, i.e. it's not integer value. And plugin_sid only supports integer value hence we need to translate it.
  below is the example of plugin with translation

```
[DEFAULT]
plugin_id=934014

[config]
type=detector
enable=yes

source=log
location=/var/log/devices[redacted].log
create_file=true

process=
start=no
stop=no
restart=no  ; restart plugin process after each interval
startup=
shutdown=

[translation]
DNS request=1
Packet dropped=2
strict TCP state=3
Authentication successful=4
web request blocked, forbidden category detected=5
http access=6
_DEFAULT_=99999

[100]
#Mar  1 00:01:01 192.168.10.1 2017: 03:01-02:30:05 [redacted]-hq-sg135 ulogd[4824]: id="2001" severity="info" sys="SecureNet" sub="packetfilter" name="Packet dropped" action="drop" fwrule="60001"
initf="eth0" mark="0x3441" app="1089" srcmac="08:94:ef:2d:94:87" dstmac="00:1a:8c:4b:4e:e8" srcip="192.168.12.10" dstip="192.168.12.255" proto="17" length="78" tos="0x00" prec="0x00" ttl="128"
srcport="137" dstport="137"
event_type=event
regexp='^(?P<date>\w+\s+\d+\s+\d+:\d+:\d+).*?(?P<device>\S+).*\d+:\d+:\d+\s+\S+.*?\s+name="(?P<sid>.*?)".*?\s+srcip="(?P<src_ip>\d+.\d+.\d+.\d+)".*?\s+dstip="(?P<dst_ip>\d+.\d+.\d+.\d+)".*?\s+srcport="(?P
<src_port>\d+)".*?\s+dstport="(?P<dst_port>\d+)".*'
date={normalize_date($date)}
device={resolv($device)}
plugin_sid={translate($sid)}
src_ip={resolv($src_ip)}
dst_ip={resolv($dst_ip)}
src_port=($src_port)
dst_port={($dst_port)}
userdata8={$sid}
userdata9="100"
```

1. Provide the plugin a unique plugin_id which must be greater than 9002000 (eg : plugin_id=9002001)
2. Keep the config part as it is in the image with type=detector & enable=yes
3. Keep source=log (file format), create_file=true (destination file create flag) & location should be the input log file location (E.g. location=var/log/device/devicename.log)
4. Write Translation for each uniquely identified plugin_sid and assign a unique no to it (E.g. DNS request=1,Packet dropped=2)
5. At the end of translation always mention (_DEFAULT_=99999)

Step 3.b: Create plugin without translation:

- See attached no_translation.cfg for template
- Create a plugin without translation when the plugin_sid captured is numeric and unique for each log, below is the example of plugin without translation

```
[DEFAULT]
plugin_id=900026

[config]
type=detector
enable=yes

source=log
location=/var/log/devices/[redacted].log,/var/log/devices/[redacted].log
create_file=true

process=
start=no
stop=no
restart=no  ; restart plugin process after each interval
startup=
shutdown=

[110]
#Jul 26 17:13:11 [redacted]-10.85.21.143 (Location -> 'C:/Program Files (x86)/cchids/logs/[redacted]log'): 2016 Jul 26 17:18:24 WinEvtLog: Application: INFORMATION(9000079): CCHIDS: Previous
SHA256: '09bfd4762d995eb096dea2659c77365d93b690437821c209eb5d999d4df06280'; Current SHA256: 'd4f68743333f627cc01ed366803e31582f2768f8837ea52b5a12541de49efa95'; Integrity checksum changed for:
'HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\ACPI\Parameters\WakeUp::GenericEventStatus (6th time)'

event_type=event
regexp="^(?P<date>\w+\s+\d+\s+\d+:\d+:\d+)\s+(?P<device>\S+)\s+.*?INFORMATION\((?P<sid>\d+)\).*?[redacted]\:\s+Previous\s+SHA256\:\s+\'(?P<sha1>.*?)\'.*?Current\s+SHA256:\s+\'(?P<sha2>.*?)\'.*?for:\s+'(?P<filen
ame>.*?)\s+\(\S+\s+time\)\'$"
date={normalize_date($date)}
#device={resolv($device)}
plugin_sid={$sid}
src_ip={resolv($device)}
userdata1={$sha1}
userdata2={$sha2}
filename={$filename}
userdata8="AV6"
userdata9=110


event_type=event
regexp="^(?P<date>\w+\s+\d+\s+\d+:\d+:\d+)\s+(?P<device>\S+)\s+.*?INFORMATION\((?P<sid>\d+)\).*?$"
date={normalize_date($date)}
#device={resolv($device)}
plugin_sid={$sid}
src_ip={resolv($device)}
userdata8="AV6"
userdata9=140
```

1. Provide the plugin a unique plugin_id which must be greater than 9002000 (E.g. plugin_id=9002002)
2. Keep the config part as it is in the image with type=detector & enable=yes
3. Keep source=log (file format), create_file=true (destination file create flag) & location should be the input log file location (E.g. location=var/log/device/devicename.log)

Step 4: Create Rule:

```
[100]
#Mar  1 00:01:01 192.168.10.1 2017: 03:01-02:30:05 [redacted]-hq-sg135 ulogd[4824]: id="2001" severity="info" sys="SecureNet" sub="packetfilter" name="Packet dropped" action="drop" fwrule="60001"
initf="eth0" mark="0x3441" app="1089" srcmac="08:94:ef:2d:94:87" dstmac="00:1a:8c:4b:4e:e8" srcip="192.168.12.10" dstip="192.168.12.255" proto="17" length="78" tos="0x00" prec="0x00" ttl="128"
srcport="137" dstport="137"
event_type=event
regexp='^(?P<date>\w+\s+\d+\s+\d+:\d+:\d+).*?(?P<device>\S+).*\d+:\d+:\d+\s+\S+.*?\s+name="(?P<sid>.*?)".*?\s+srcip="(?P<src_ip>\d+.\d+.\d+.\d+)".*?\s+dstip="(?P<dst_ip>\d+.\d+.\d+.\d+)".*?\s+srcport="(?P
<src_port>\d+)".*?\s+dstport="(?P<dst_port>\d+)".*'
date={normalize_date($date)}
device={resolv($device)}
plugin_sid={translate($sid)}
src_ip={resolv($src_ip)}
dst_ip={resolv($dst_ip)}
src_port=($src_port)
dst_port={($dst_port)}
userdata8={$sid}
userdata9="100"
```

1. Provide the regex a rule number (i.e [100])

   This number plays important role, as it's the order in which regex will be executed.

2. After the rule number set event_type=event.
3. Enter your regex which captures all the essential fields
4. Date field should always be in normalized format (E.g. date={normalize_date($date)})
5. Device field should be resolved in case it captures hostname (E.g. (device={$device_ip}))
6. scr_ip and dst_ip field should also be resolved in case it captures hostname as shown above
7. src_port and dst_port to be captured as shown above
8. In case we need to capture extra fields, it can be stored from userdata1 to userdata9

Signature Identification

Once we have the regular expressions ready and we can parse the incoming logs, we need to identify the signatures. With Signatures, we can identify the nature of the log whether it's an information, warning or threat alert. For commercial devices/application, the vendor usually provides the audit logging details such as event number, name and description. For custom applications, define your own signatures based on event type and severity. Use attached template Signature_Template.xlsx to provide signature details. Primary rows are examples.
Please leave the columns Plugin ID, Signature ID, Category and Subcategory blank in case you are not sure what to write there. CCREPORT column will be filled by ControlCase. Reliability is a number from 1 to 10 which defines how frequently the event will occur. E.g. an event with Reliability 6 means it's likely that he event occurs every day. Priority is a number between 1 to 10 which defines how critical the event is. E.g. an event with Priority 7 means it's a high priority event.

Define Correlations

With correlations, an event can be converted to an alarm. E.g. if we are seeing numerous login failures event in a very short period of time, we can conclude that an attack is in progress and with a correlation between the number of events and time we can generate an alarm.
Please use attached template Signature_Correlation.xlsx to provide the correlations. Primary rows are examples.
In this template, use Threshold column to define on how many occurrences of that event, an alarm needs to be generated.