



Tossing Splunk in Your PAN

Ninja's Guide to the Galaxy of Splunk and Palo Alto Networks

Kevin Gonzalez | Security Operations Manager

September 25, 2017 | Washington, DC

Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2017 Splunk Inc. All rights reserved.

Agenda

1. Utilizing Splunk Enterprise Security to:
 - Reduce alert chaos
 - Tame your PANW Threat Intelligence Feeds
2. Saving time with a Splunk/PANW API Fusion
3. Knowing the “who” at all times by populating PANW’s User-ID
4. Utilizing the Splunk Universal Forwarder to fix all of your problems

Splunk Enterprise Security & Palo Alto Networks



► Incident Response

- Single pane of glass
- Adaptive response based on correlated information

► Threat Intelligence

- Centralized repository
- Easy to maintain
- Deduplication and content filtering



Splunk Enterprise Security & Palo Alto Networks

Integrating PANW into Enterprise Security

The screenshot displays the Splunk Enterprise Security Incident Review interface. At the top, there are navigation tabs for Security Posture, Incident Review, Investigations, Glass Tables, Security Intelligence, Security Domains, Audit, Search, and Configure. The main content area is titled 'Incident Review' and includes a search bar with the query 'botnet activity detected'. On the left, there is a 'Urgency' filter showing counts for CRITICAL (0), HIGH (0), MEDIUM (64), LOW (537), and INFO (0). Below this are fields for Status, Owner, Security Domain, and Tag, along with a 'Submit' button. A bar chart shows the distribution of 601 events from May 1 to May 22, 2017. Below the chart is a table of matching events, including a 'Botnet Activity Detected' event with a description of a user calling out to a forum. The interface also includes sections for correlation search, history, contributing events, and adaptive responses.

► Create Notable Events geared towards your PAN Firewall

- Auto-Wildfire submissions
- Per-Notable event tagging for PAN Dynamic Block Lists

► Utilize Threat Intelligence Data Models to feed your Palo Alto Network Firewalls:

- De-duplicate and publish filtered threat intel
- Auto-add trusted intel to dynamic block lists via tags

Splunk Enterprise Security & Palo Alto Networks

Correlation Search

Search Name *

Application Context *

Description
Describes what kind of issues this search is intended to detect.

Mode

Search *

```
index=*_pan sourcetype="pan:threat"
category="malware" client_location="10.0.0.0-
10.255.255.255"
| fillnull value="-" src_user
| stats count by url, src_user, dvc, client_ip,
dest_location, filename, dest_ip
| table dvc, src_user, client_ip, dest_ip,
dest_location, filename, url, count
| sort -count
| eventstats sum(count) as sCount by client_ip
```

Time Range

Earliest time
Set a time range of events to search. Type an earliest time using relative time modifiers.

Latest time
Type a latest time using relative time modifiers.

Cron Schedule *
Enter a cron-style schedule. For example `*/5 * * * *` (every 5 minutes) or `0 21 * * * *` (every day at 9 PM). Real-time searches use a default schedule of `*/5 * * * *`.

Scheduling
Controls the way the scheduler computes the next execution time of a scheduled search.
This controls the `realtime_schedule` setting. [Learn more](#)

Adaptive Response Actions

+ Add New Response Action

- PAN : Tag to Dynamic Address Group
- Risk Analysis
- PAN : Submit URL to WildFire
- Notable

```
index=*_pan sourcetype="pan:threat" category="malware" client_location="10.0.0.0-10.255.255.255"
| fillnull value="-" src_user
| stats count by url, src_user, dvc, client_ip, dest_location, filename, dest_ip
| table dvc, src_user, client_ip, dest_ip, dest_location, filename, url, count
| sort -count
| eventstats sum(count) as sCount by client_ip
| where sCount > 4
| lookup ut_parse_extended_lookup url
| fields - sCount
```

Splunk Enterprise Security & Palo Alto Networks

Correlation Search

Search Name * PAN -Threat Intel Update

Application Context * Enterprise Security ▾

Description Auto-tags malicious IPs into a dynamic blocklis

Describes what kind of issues this search is intended to detect.

Mode Guided Manual

Search *

```
| from
datamodel:"Threat_Intelligence.IP_Intelligence"
| search threat_key="trusted_source"
| pantag device="10.1.1.1" action="add"
ip_field=ip tag="infected-host"
```

```
| from datamodel:"Threat_Intelligence.IP_Intelligence"
| search threat_key="trusted_threat_source"
| pantag device="x.x.x.x" action="add" ip_field=ip tag="no_bueno"
```

Time Range

Earliest time -24h

Set a time range of events to search. Type an earliest time using relative time modifiers.

Latest time now

Type a latest time using relative time modifiers.

Cron Schedule * 00 01 ***

Enter a cron-style schedule. For example */5 ***

(every 5 minutes) or 0 21 * * * (every day at 9 PM).

Real-time searches use a default schedule of */5 ***.

Scheduling Real-time Continuous

Controls the way the scheduler computes the next execution time of a scheduled search.

This controls the realtime_schedule setting. [Learn more](#)



pantag

<http://pansplunk.readthedocs.io/en/latest/commands.html#pantag>

pantag

The `pantag` command shares context with the firewall by tagging IP addresses found in Splunk into [Dynamic Address Groups](#).

Command added in App version 4.1. New parameters added in App version 5.0.

Syntax:

```
pantag device=<hostname>|panorama=<hostname>
[serial=<serial-of-device-in-panorama>] [vsys=<vsys#>]
[action=<add|remove>] [ip_field=<field-containing-IPs>]
tag=<tag>|tag_field=<field-containing-tags>
```

Parameter	Default	Added in	Usage
device		4.1	IP or hostname of firewall
panorama		5.0	IP or hostname of Panorama
serial		5.0	Serial of firewall (required if using panorama parameter)
vsys	vsys1	5.0	VSYS ID (eg. vsys2)
action	add	4.1	Add or remove the tag
field	src_ip	4.1	Same as ip_field parameter (deprecated in 5.0, use ip_field)
ip_field	src_ip	5.0	Log field containing IP address to tag
tag		4.1	Tag for the IP, referenced in the Dynamic Address Group
tag_field		5.0	Log field containing the tag for IP address in the same log

Splunk SDK for Python

<https://github.com/splunk/splunk-sdk-python>

splunk / splunk-sdk-python Watch 84 Star 243 Fork 127

Code Issues 13 Pull requests 10 Projects 0 Wiki Insights

Splunk Software Development Kit for Python <http://dev.splunk.com>

967 commits 4 branches 16 releases 23 contributors Apache-2.0

Branch: master New pull request Find file Clone or download

shakeelmohamed Update changelog Latest commit F32374a on Dec 20, 2016

docs	Add KV Store support to the Python SDK	2 years ago
examples	Update version number	7 months ago
splunklib	Update version number	7 months ago
tests	Add test coverage for unknown input types	7 months ago
utils	Updated copyright date from 2011-2014 to 2011-2015	2 years ago
.gitattributes	Updated .gitattributes to prevent carriage return linefeed processing...	2 years ago
.gitignore	update gitignore	2 years ago
.pylintrc	Splunk Python SDK 0.1.0 Release	6 years ago
.travis.yml	Remove Splunk 6.4 from Travis settings	a year ago
CHANGELOG.md	Update changelog	7 months ago
CONTRIBUTING.md	Updated CONTRIBUTED.md to reflect the current release of Splunk and t...	2 years ago
Commands.conf.spec.xlsx	Check in custom search commands v2	2 years ago
LICENSE	Splunk Python SDK 0.1.0 Release	6 years ago
MANIFEST.in	Splunk Python SDK 0.1.0 Release	6 years ago
README.md	Update version number	7 months ago
setup.py	Add Travis CI support for the Python SDK	2 years ago
sitecustomize.py	Updated copyright date from 2011-2014 to 2011-2015	2 years ago
splunkrc.spec	Updated splunkrc.spec file from version=5.0 to version=6.3	2 years ago

PAN-OS XML API

<https://www.paloaltonetworks.com/documentation/71/pan-os/xml-api>

Home > Technical Documentation > PAN-OS 7.1 Documentation > PAN-OS® and Panorama™ 7.1 XML API Usage Guide

PAN-OS® and Panorama™ 7.1 XML API Usage Guide

Search



About the PAN-OS XML API

Get Started with the PAN-OS XML API

PAN-OS XML API Use Cases

PAN-OS XML API Request Types

PAN-OS XML API Error Codes

DOWNLOAD PDF

Next >

About the PAN-OS XML API

The PAN-OS and Panorama XML API allows you to manage firewalls and Panorama through a programmatic XML-based API. Use this API to access and manage your firewall through a third-party service, application, or script.

The PAN-OS XML API uses a tree of XML nodes to map firewall or Panorama functionality. To make an API request, you must specify the [XPath](#) (XML Path Language) to the XML node that corresponds to a specific setting or action. XPath allows you to navigate through the hierarchical XML tree structure for firewalls and Panorama.

Use the PAN-OS XML API to automate tasks such as:

- create, update, and modify firewall and Panorama configurations
- execute operational mode commands, such as restart the system or validate configurations
- retrieve reports
- manage users through User-ID
- update dynamic objects without having to modify or commit new configurations

Because PAN-OS XML API functionality mirrors that of the web interface and CLI, familiarize yourself with both. Reading relevant portions of the [PAN-OS Administrator's Guide](#) will help you get a better understanding of firewall functionalities that the API can access. You should also be knowledgeable about web service APIs, HTTP, XML, and XPath.

▲ [PAN-OS XML API Components](#)

▲ [Structure of a PAN-OS XML API Request](#)

Updating PAN-OS Address Objects

```

1 #!/usr/bin/env python
2 import splunklib.client as client
3 import requests
4 import re
5
6 # function to rename address objects
7 def pan_name_update (adDict, apiKey):
8     url = None
9     tagStripper = re.compile("\<ip\>\<netmask\>(\d{1,3}\.){3}\d{1,3}\</ip\>\</netmask\>")
10    for dictObj in dictList:
11        host = dictObj['host']
12        ip = tagStripper.sub('', dictObj['ip'])
13        url = "https://10.1.1.1/api/?type=config&action=rename&xpath=/config/shared/address/entry[@name='%s']&newname=%s" % (ip,
14        requests.get(url=url, verify=False)
15
16 # function to set address objects with pan
17 def pan_tag_update (adDict, apiKey):
18     url = None
19     host = adDict['host']
20     ip = adDict['ip']
21     tags = adDict['tags']
22     spaceStripper = re.compile("\s+")
23     # validate whether address object already exists, if not create it
24     url_prelim = "https://10.1.1.1/api/?type=config&action=show&xpath=/config/shared/address/entry[@name='%s']&as" % (host, apiKey)
25     response = requests.get(url=url_prelim, verify=False)
26     response_data = response.content.split('\n')
27     if re.search("No\susuch\snode", response_data[0]):
28         url = "https://10.1.1.1/api/?type=config&action=set&xpath=/config/shared/address/entry[@name='%s']&element=%s&as" % (host
29         requests.get(url=url, verify=False)
30     else:
31         # iterate through current tags
32         current_tags = None
33         desc = ""
34         for resp_obj in response_data:
35             match_ip = re.search("\<ip\>-", resp_obj)
36             if match_ip:
37                 pan_ip = spaceStripper.sub('', resp_obj)
38                 print pan_ip
39             match_desc = re.search("\<description\>", resp_obj)
40             if match_desc:
41                 desc = spaceStripper.sub('', resp_obj)
42                 print desc
43             match_mem = re.search("\<member\>", resp_obj)
44             if match_mem:
45                 tag_member = spaceStripper.sub('', resp_obj)
46                 if current_tags == None:
47                     current_tags = "%s" % tag_member
48                 else:
49                     current_tags = "%s" % (current_tags, tag_member)
50             current_tags = "<tag-%s/>" % current_tags
51             print current_tags
52             if ip != pan_ip or tags != current_tags:
53                 url = "https://10.1.1.1/api/?type=config&action=edit&xpath=/config/shared/address/entry[@name='%s']&element=entry_name
54                 requests.get(url=url, verify=False)
55
56 # connect to splunk
57 service = client.connect(
58     username = 'admin',
59     password = 'password',
60     host = '10.1.1.2',
61     port = '8089' )
62
63 # define static arguments for splunk and pan
64 apiKey = 'key=AF3HIAEIAHF10110F0Q1J0FASFJ010J0F1JAFKJ3R3ASIAJ5FAJFLKFKLJ1F0Q1F9KLA9FRIE0QPPMA'
65 kwargs_oneshot = {'count': 0}
66
67 # oneshot for ip to host address group mappings with tags
68 search = 'search index=scripts sourcetype=script:winsocreset "server" earliest=-24h | eval host=upper(host) | rex field=_row "05'
69 results = service.jobs.oneshot(search, **kwargs_oneshot)

```

► Address object:

- Standardization
- Creation
- Modification

► Splunk query that contains all necessary data

► + crontab

Updating User-ID With RADIUS Logs

```
index=meraki mac=* src=10.* src!=0.0.0.0
| eval src_mac=upper(replace(mac,":","-"))
| rex field=src "(?<src_ip>(\d{1,3}\.){1,3}\d{1,3})"
| dedup src_mac src_ip
| table src_mac src_ip
| outputlookup meraki_src.csv
```

```
index=wineventlog host=radius1 Client_IP_Address=* Security_ID!=*$ src_ip=*
| mvexpand Security_ID
| mvexpand src_ip
| dedup Security_ID src_ip
| search Security_ID!="NULL SID"
| rename Security_ID as user
| panuserupdate panorama=x.x.x.x serial=001234567890
| fields user src_ip status
```

- ▶ X amount of incomplete sources can depict a full picture
- ▶ Wireless AP Logs
 - Device MAC
 - Device IP
- ▶ RADIUS Logs
 - User
 - Device MAC
 - Automatic Lookups

panuserupdate

The `panuserupdate` command synchronizes user login events with Palo Alto Networks User-ID.

More information: [User-ID with Splunk](#)

Added in App version 5.0. For previous versions, refer to the [panupdate](#) command.

Syntax:

```
panuserupdate device=<hostname>|panorama=<hostname>
[serial=<serial-of-device-in-panorama>] [vsys=<vsys#>]
[action=<login|logout>] [ip_field=<field-containing-IPs>]
user_field=<field-containing-usernames>
```

Parameter	Default	Usage
device		IP or hostname of firewall
panorama		IP or hostname of Panorama
serial		Serial of firewall (required if using panorama parameter)
vsys	vsys1	VSYS ID (eg. vsys2)
action	login	Tell the firewall user logged in or logged out
ip_field	src_ip	Log field containing IP address
user_field	user	Log field containing the username

panuserupdate

<http://pansplunk.readthedocs.io/en/latest/commands.html#panuserupdate>

Updating User-ID With SIP and “Best Guess” Scenarios

```
index=*_pan url="*/?sipuri=*"
| rex field=url "\/\/?sipuri\=(?<user>[\w\-\.\@]+)"
| mvexpand user
| dedup user src_ip
| search user!=unknown
| panuserupdate panorama=x.x.x.x serial=001234567890
| fields user src_ip status
```

```
index=*_pan (user=unknown OR user="no user") ("@domain1.com" OR "@domain2.com")
| eval tmp=lower(_raw)
| rex field=tmp "\=(?<userTmp>[a-z\.]+"@(\domain1|\domain2)\.com)"
| rex field=userTmp "(?<pre>[a-z\.]+"@(?<suf>(\domain1|\domain2)\.com)"
| eval user=pre+"@"*"+suf
| dedup user src_ip
| panuserupdate panorama=x.x.x.x serial=001234567890
| fields user src_ip status
```

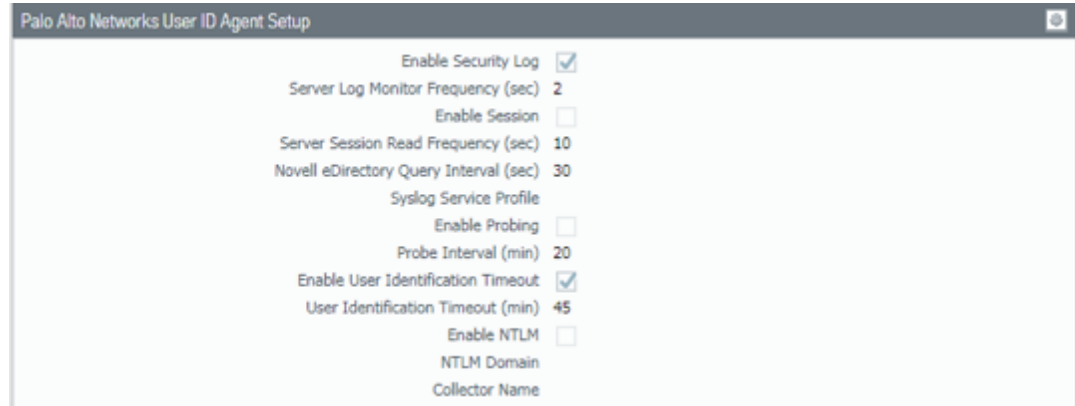
- ▶ Know your apps
- ▶ Know the data from those apps
- ▶ Now apply that knowledge
- ▶ Examples:
 - SIP Data from Lync/Skype
 - Raw URL Logs passing user info

“Best Guess” Scenarios and Timeouts...

```

1  #!/usr/bin/env python
2  import splunklib.client as client
3  import requests
4  import re
5
6  # connect to splunk
7  service = client.connect(
8      username = 'admin',
9      password = 'password',
10     host = '10.1.1.2',
11     port = '8089' )
12
13 # define static arguments for splunk and pan
14 apiKey = 'key=ASFJKQNFQNFQIASFIOEJOIFJOKVMQFPIINFPOIQNIPOFUNIPUMNFVAKSKLNFQWQHFQINLKAFNP
15 kwargs_oneshot = {"count": 0}
16
17 # oneshot search
18 search = 'search index=*pan (user=unknown OR user="no user") ("domain1.com" OR "domain2.com")'
19 results = service.jobs.oneshot(search, **kwargs_oneshot)
20
21 # parse into list
22 resultList = ([])
23 for xmlObj in results:
24     resultList.append(xmlObj)
25
26 # begin parse and pan user-id updates
27 count = 0
28 fullEntry = ''
29 rexStripper = re.compile("\s+|\<value\>|\<text\>|\</text\>|\</value\>")
30 for obj in resultList:
31     match = re.search("\<field\s\k=\<'user'\>", obj)
32     if match:
33         user = rexStripper.sub('', resultList[count+1])
34         ip = rexStripper.sub('', resultList[count+4])
35         entry = '<entry name="%s" ip="%s" timeout="360"/>' % (user, ip)
36         fullEntry = '%s%s' % (fullEntry, entry)
37     count += 1
38
39 # append pre and post xml tags for user-id
40 xmlStart = '<uid-message><type>update</type><payload><login>'
41 xmlEnd = '</login></payload></uid-message>'
42 xmlString = '%s%s%s' % (xmlStart, fullEntry, xmlEnd)
43
44 # post user-id info
45 url = "https://10.1.1/api/?type=user-id&vsys=vsys1&cmd=%s&key=%s" % (xmlString, apiKey)
46 requests.get(url=url, verify=False)

```



- ▶ Individualized User-ID timeout
- ▶ Increased customization
- ▶ APIs are your best friend...

Updating User-ID with Custom Scripts

```
$os = (gwmi -Class win32_operatingsystem).caption
if($os -notmatch "server"){
  wmic computersystem get username
  $ipL = gwmi -Class win32_networkadapterconfiguration | ?{$_ .defaultipgateway -ne $null} | select ipaddress
  foreach($a in $ipL){
    $ipA = @()
    $ipA += $a.ipaddress
    foreach($ip in $ipA){
      Write-Host "IP = $ip"
    }
  }
}
```

```
index=*_scripts sourcetype=Script:UserAffinity IP=*
| rex field=_raw "UserName\s+(?<user>\w+[\x5C]\w+)"
| rex max_match=5 field=_raw "IP\s=\s(?<src_ip>(\d{1,3}\.){3}\d{1,3})"
| mvexpand src_ip
| dedup user src_ip
| fillnull value="No User" user
| panuserupdate panorama=x.x.x.x serial=001234567890
| fields user src_ip status
```

- ▶ Endpoints are your true source
 - Contain LAN/WAN IPs
 - Contain User Info
- ▶ The Splunk Universal Forwarder are also your best friend
- ▶ Historical evidence of all User/IP mappings... completed.

Globalprotect Bug Workaround Utilizing The Splunk Universal Forwarder

- ▶ v3.0.x
- ▶ Ubiquitous VPN on Windows
- ▶ Ungraceful Network Disconnects
- ▶ Internal DNS Settings Frozen

```

$checkGP = gwmi -class win32_product | ?{$_name -match "globalprotect"}
if($checkGP -ne $null){
    $n = $checkGP.name
    $v = $checkGP.version
    Write-Host "Name = $n"
    Write-Host "Version = $v"
    wmic computersystem get username

    if (!(Test-Connection -ComputerName validationurl1.com -Count 2 -ErrorAction SilentlyContinue) -and !(Test-Connection -ComputerName validationurl2.com -Count 2 -ErrorAction SilentlyContinue)) {
        Write-Host "External Test Completed. No Connection Found."
        if (!(Test-Connection -ComputerName validationIP1 -Count 1 -ErrorAction SilentlyContinue) -and !(Test-Connection -ComputerName validationIP2 -Count 1 -ErrorAction SilentlyContinue)) {
            Write-Host "Internal Test Completed. No Connection Found."
            C:\windows\system32\ipconfig.exe /all
            C:\windows\system32\netsh.exe int ipv4 reset
            C:\windows\system32\netsh.exe int ipv6 reset
            C:\windows\system32\netsh.exe winsock reset catalog
            C:\windows\system32\netsh.exe interface ip set dns name= "Wireless Network Connection" dhcp
            C:\windows\system32\netsh.exe interface ip set dns name= "Local Area Connection" dhcp
        }
    }
}

```


Thank You

Don't forget to **rate this session** in the
.conf2017 mobile app

splunk> **.conf2017**