Splunk

Splunk

* Splunk 101
  + Navigating Splunk
    - Messages – System-level Messages
    - Settings – Configure Splunk instance
    - Activity – review progress of jobs
    - Help – misc. tutorials
    - Find – search features
    - App – Drop down
    - Apps Panel – see apps installed for Splunk instance
      * Default App – Search & Reporting
    - Explore Splunk
      * quick links to add data to instance, add new apps, and access documentation
    - Home Dashboard
      * none displayed by default, but can be changed
    - <https://docs.splunk.com/Documentation/Splunk/8.1.2/SearchTutorial/NavigatingSplunk>
  + Splunk Apps
    - Search & Reporting
      * enter Splunk queries to search through data ingested by Splunk
    - Apps > cog > Manage Apps
      * Edit properties
        + to land within a specific app automatically post login, edit user-prefs.conf

C:\Program Files\Splunk\etc\apps\user-prefs\default\user-prefs.conf

/opt/splunk/etc/apps/user-pref/default/user-prefs.conf

* + - * + need to restart splunk from command line for user preferences to take hold

net stop splunkd, net start splunkd

* + - * + Find More Apps
      * Splunk Apps
        + Splunkbase

<https://splunkbase.splunk.com/>

need an account

* + - * + Install app from file
        + To Remove app via Command Line

C:\Program Files\Splunk\bin>splunk.exe remove app app-name -auth splunk-username:splunk-password

* + Adding Data
    - Sources can be event logs, website logs, firewall logs
    - Grouped into categories
      * <https://docs.splunk.com/Documentation/Splunk/8.1.2/Data/Getstartedwithgettingdatain#Use_apps_to_get_data_in>
    - Add Data
      * Windows Event Logs and Sysmon Logs 🡪 Monitor
        + Local Event Logs

Available Items(s)

* + - * Settings > Data Inputs
  + Splunk Queries
    - Enter an asterisk and change the timeframe to search All time
    - Specific source or sourcetype
      * or select source/sourcetype under Selected Fields
    - names (values) of each source and the number of events (count), and the percentage value (%) of all the events of each source
      * start query with Sysmon as the source
        + source="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational”
      * click > to expand event to make it more readable
        + can expand query by adding event IDs such as EventID=12 (RegistryEvent Object Create and Delete) – case sensitive
        + can also search via keyworks such as GoogleUpdate.exe – not case sensitive
      * Can add highlighted values to search from expanded view
        + can do multiple keyworks

ex. \* GoogleUpdate.exe chrome\_installer.exe

* + - * + to search for a phrase, surround phrase with quotes

\* “failed password for sneezy”

* + - * be sure to use “Interesting Fields” sidebar
        + RuleName values can be used to expand query
    - <https://www.splunk.com/pdfs/solution-guides/splunk-quick-reference-guide.pdf>
  + Sigma Rules
    - generic and open signature format that allows you to describe relevant log events in a straightforward manner
    - queries (or rules) can be created in the Sigma format and shared with teams that don't use Splunk. Sigma rules can be shared along with IOCs and YARA rules as Threat Intelligence.
    - written in YAML
    - online tool that does conversions automatically
      * Uncoder.io
        + convert Sigma to Splunk and Visa Versa

ex. sigma: APT29 🡪 CommandLine="\*-noni -ep bypass $\*"

* + - * Github Repo
        + <https://github.com/SigmaHQ/sigma>
  + Dashboard and Visualization
    - To Create a Dashboard in Search App
      * Search > Dashboards > Create New Dashboard > Save
      * Create Query
        + Display Top 5 Sysmon Event IDs

source=”XmlWinEventLog:Microsoft-Windows-Sysmon/Operational” | top limit=5 EventID

* + - * + Visualization

Charts

* + - * Save as Dashboard > Existing
      * Set as Home Dashboard
  + Alerts
    - <https://docs.splunk.com/Documentation/SplunkCloud/8.1.2012/Alert/AlertWorkflowOverview>
      * Cannot be done in free version
    - <https://docs.splunk.com/Documentation/Splunk/latest/Alert/Aboutalerts>
    - <https://docs.splunk.com/Documentation/SplunkCloud/8.1.2012/Alert/Alertexamples>
  + Conclusion
    - SPL and regex
    - <https://docs.splunk.com/Documentation/Splunk/8.1.2/Knowledge/AboutSplunkregularexpressions>
    - <https://www.splunk.com/en_us/training/courses/splunk-fundamentals-1.html?utm_medium=email&utm_source=nurture&utm_campaign=GLOBAL_Enterprise_Trial_Learning_Mar19&utm_content=Splunk-fundamentals-1&elqTrackId=590afdde558446d1b16f45726f1bdbfb&elq=7e380948a4ab47419542ec6b54519247&elqaid=21461&elqat=1&elqCampaignId=15042>

Splunk 102

* Dive Into The Data
  + https://github.com/splunk/botsv2
  + SPL command metadata can be used to search for information found in Data Summary with specific index options
    - * returned in EPOCH time – need to use eval command to provide more human-friendly formatting
    - <http://docs.splunk.com/Documentation/Splunk/latest/SearchReference/Metadata>
    - <https://www.splunk.com/blog/2017/07/31/metadata-metalore.html>
  + | metadata type=sourcetypes index=botsv2 | eval firstTime=strftime(firstTime,"%Y-%m-%d %H:%M:%S") | eval lastTime=strftime(lastTime,"%Y-%m-%d %H:%M:%S") | eval recentTime=strftime(recentTime,"%Y-%m-%d %H:%M:%S") | sort – totalCount
* 100 Series Questions
  + HTTP Traffic – Dest and Src Addrs
    - index=”botsv2” amber
      * PAN traffic
        + index=”botsv2” sourcetype=”pan:traffic”
    - index=”botsv2” <IPADDR> sourcetype=”stream:HTTP”
* <https://ivanitlearning.wordpress.com/2020/06/15/hunting-with-splunk-botsv2-qns-1xx/>
* <https://ivanitlearning.wordpress.com/2020/06/12/hunting-with-splunk-botsv2-qns-2xx/>
* <https://ivanitlearning.wordpress.com/2020/06/20/hunting-with-splunk-botsv2-qns-3xx/>
* <https://ivanitlearning.wordpress.com/2020/06/23/hunting-with-splunk-botsv2-qns-4xx/>
* See Splunk Folder For Saved Examples Of Use Cases
* Common Splunk Commands
  + search – queries always begin with this command unless specifically specified
    - rare/top – displays the most/least common values of a field
  + index – storage point for imported data
  + dashboard – allows creation of views that enable consistent pulls of same searches repeatedly
  + dedup – removes copies of same data during imports
  + transactions – included in search to track how long event pairs take
  + can use pipe | command to input results into further commands
  + timechart/chart – used to plot occurrences of events tracked over time
  + stats – provides general statistical information about a search
  + fields – data imported into splunk is categorized into these columns
  + host/source/source type – point of origination for data uploaded into splunk
    - sourcetype (syntax) – classify points of origin and group together
  + eval – command prior to evaluation of performed functions on data
  + rex – command to use regex
  + pivot table – create subsets and specific views for less technical Splunk users
  + \_time – proper name for time date field
  + head – specifically include only the first few values found within search
  + reverse – flip the order that results are returned in
  + lookup – include within a search to rename fields using user-provided tables of values
  + bucket – collect events into specific time frames for further processing
  + span – define data into specific sections of time to be used within chart commands
  + count – number occurrence of an event
  + <https://splunkbase.splunk.com/> - splunk apps location
  + BOTS – Boss of the SOC
  + CIM – Common Information Model
  + splunk forums – answers.splunk.com
* Splunk Quick Reference Guide
* BOTS
  + <https://www.splunk.com/blog/2017/09/06/what-you-need-to-know-about-boss-of-the-soc.html>
* Overview
  + Dealing with website defacement, ransomware infection
  + Lockheed Martin’s Kill Chain
    - 1 – Recon
    - 2 – Weaponization
    - 3 – Delivery
    - 4 – Exploitation
    - 5 – Installation
    - 6 – Command and Control (C2)
    - 7 – Actions on Objectives
* APT review in Splunk
  + Reviewing All Data Available
    - SPL command ‘metadata’ can be used to search for same info found in Data Summary with additional context options
      * returned in EPOCH time
    - <http://docs.splunk.com/Documentation/Splunk/latest/SearchReference/Metadata>
    - metadata command
      * | metadata type=sourcetypes index=<index> (ex. botsv1)
  + Building Kill Chain and Traffic Flow Diagrams
    - * <http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf>
    - Finding the IP Scanning Your Web Server - Reconnaissance
      * Understanding Different Sourcetypes
        + Identify sourcetypes Associated with Search Values

index=botsv1 imreallynotbatman.com

index=<index choice> <url>

* + - * + Finding Source Addresses

index=<index> <url>

time picker set to All Time or specific time frame if dataset is large

* + - * + Selecting a sourcetype and Searching for Source Address

index=<index choice> <url> sourcetype=stream:http

check for web data that is being seen

Stream is a free app for Splunk that collects wire data for protocols

check what IP addresses are associated with domain

* + - Validating the IP that is Scanning Your Web Server – Reconnaissance
      * Take IP addresses found with stream data and change source type to IDS to view logged events coming from suspect IP address
      * index=<index> <url> src=<suspect IP> sourcetype=<IDS>
        + ex. index=botsv1 imreallynotbatman.com src=40.80.148.42 sourcetype=suricata
      * view web signatures to confirm IP address is scanning website
        + look for web scanner tools, such as Acunetix
      * Another Option: Search Stream sourcetype and Count the src-IP
        + index=<index> <url> sourcetype=<src> | stats count(src\_ip) as Requests by src\_ip | sort – Requests
        + ex. index=botsv1 imreallynotbatman.com sourcetype=stream\* | stats count(src\_ip) as Requests by src\_ip | sort – Requests
    - Identify the Web Vulnerability Scanner - Reconnaissance
      * Looking at src\_headers
        + index=<index> src=<suspect IP> sourcetype=<src>
        + ex. index=botsv1 src=40.80.148.42 sourcetype=stream:http

look for scanner information

* + - * Looking at http\_user\_agent Strings
        + index=<index> src=<suspect IP> sourcetype=<src>
        + ex. index=botsv1 src=40.80.148.42 sourcetype=stream:http

check for anomalous/unusual agent strings

* + - Determining Which Web Server is the Target
      * Research system applications to determine logs of critical review importance
      * Identify IP address of Victim System
        + index=<index> src=<suspect IP> sourcetype=<src>
        + ex. index=botsv1 src=40.80.148.42 sourcetype=stream:http

look for server IP with majority traffic

* + - * Digging into the URI
        + index=<index> dest=<victim IP> sourcetype=<src>
        + index=botsv1 dest=192.168.250.70 sourcetype=stream:http

review URLs and URIs to determine content and directory structures being targeted

* + - * Looking for Confirmation
        + index=<index> dest=<victim IP> sourcetype=<src> status=<status code>
        + ex. index=botsv1 dest=192.168.250.70 sourcetype=stream:http status=200 | stats count by uri | sort – count
      * Finding the Answer with IIS

index=botsv1 sourcetype=iis sc\_status=200 | stats values(cs\_uri\_stem)

* + - * + Review IIS logs for Host, URI strings, response code
    - Identifying Where a Brute Force Attack Originated
      * Looking in Wire Data – stream:http
        + index=<index> sourcetype=<src>
        + ex. index=botsv1 sourcetype=stream:http

top src values, view IPs associated with http events

* + - * Refine Search with Web Server Address
        + index=<index> sourcetype=<src> dest=”<victim IP>”
        + ex. index=botsv1 sourcetype=stream:http dest="192.168.250.70"

look for traffic going to victim address

external addresses vs internal 192 addresses

* + - * Same Search, Different Port
        + index=<index> sourcetype=<src> dest=”<victim IP>”
        + ex. index=botsv1 sourcetype=stream:http dest="192.168.250.70"

look at http\_method requests

number of POSTs vs GETs = higher POSTs 🡪 indicate Brute Forcing

* + - * Adding an HTTP Method to Narrow Results
        + index=<index> sourcetype=<src> dest=”<victim IP>” http\_method=<POST or GET>
        + ex. index=botsv1 sourcetype=stream:http dest="192.168.250.70" http\_method=POST
      * Finding Passwords in HTTP Wire Data
        + index=<index> sourcetype=<src> dest="victim IP" http\_method=POST form\_data=\*username\*passwd\*
        + ex. index=botsv1 sourcetype=stream:http dest="192.168.250.70" http\_method=POST form\_data=\*username\*passwd\*

using wildcards to look for values that contain strings username and passwd

use | table form\_data to tabulate view into an easy to read format

ensure time picker is set to narrow to limit exhaustive search

* + - * Identifying the First Password Attempted in a Brute Force Attack
        + What was the first brute force password used?

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | table \_time form\_data

* + - * + Reversing the Order of Output

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | table \_time form\_data

| reverse

* + - * + A More Elegant Way To View Passwords

index=botsv1 sourcetype=stream:http http\_method=POST

| rex field=form\_data "passwd=(?<userpassword>\w+)"

| search userpassword=\*

| reverse

| head 1

| table userpassword

extracts passwords to a new field containing password string values only when the new field userpassword exists and contains data, then reverses the output, returns the first record in the data set, and outputs only the userpassword field

* + - Extracting Passwords from Events
      * Extracting Password from form\_data

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\*

| rex field=form\_data "passwd=(?<userpassword>\w+)"

| table userpassword

* + - * + extract values from form\_data and look for string that starts with passwd= and capture all alphanumeric characters and placed in a table userpassword
      * Calculating the Length

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | rex field=form\_data "passwd=(?<userpassword>\w+)"

| eval lenpword=len(userpassword)

| table userpassword lenpword

* + - * + look for passwords of a given value/length
      * Finding a password using an external source file

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | rex field=form\_data "passwd=(?<userpassword>\w+)" | eval lenpword=len(userpassword) | search lenpword=6

| eval password=lower(userpassword)

| lookup coldplay.csv song as password OUTPUTNEW song

| search song=\*

| table song

* + - * + this uses a csv file made from copying known coldplays songs and searching for a given output that matches
    - Identifying the password used to gain access

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* dest\_ip=192.168.250.70 | rex field=form\_data "passwd=(?<userpassword>\w+)"

| stats count by userpassword | sort – count

* + - * + password extraction for values used more than once, as successful inputs will be tested again by tools such as hydra
      * stats command provides a wider view

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* dest\_ip=192.168.250.70 | rex field=form\_data "passwd=(?<userpassword>\w+)"

| stats count values(src) by userpassword | sort – count

* + - * + determines where the login came from
      * Collecting additional attributes around the login events

index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* dest\_ip=192.168.250.70 src=40.80.148.42 | rex field=form\_data "passwd=(?<userpassword>\w+)"| search userpassword=\* | table \_time uri userpassword

* + - * + add source address and URI associated with penetration
    - Finding Average Length of the Passwords During the Brute Force Attack

index=botsv1 sourcetype=stream:http http\_method=POST | rex field=form\_data "passwd=(?<userpassword>\w+)" | search userpassword=\*

| eval mylen=len(userpassword)

| stats avg(mylen) AS avg\_len\_http

| eval avg\_len\_http=round(avg\_len\_http,0)

* + - * + determine an average length using stats command or average command
        + round out with eval and round command + function
    - Determining the Elapsed Time Between Events
      * Tabling Logins with Same Password

index=botsv1 sourcetype=stream:http | rex field=form\_data "passwd=(?<userpassword>\w+)" | search userpassword=batman | table \_time userpassword src

* + - * + look for delta in time between successful login events

grabbing \_time and src

* + - * Transaction command

index=botsv1 sourcetype=stream:http | rex field=form\_data "passwd=(?<userpassword>\w+)" |search userpassword=batman

| transaction userpassword | table duration

* + - * + group events together with transaction and return a field called duration that calculates difference between first and last event

round out with eval round

* + - Identifying the Number of Unique Passwords Attempted During the Brute Force Attack
      * Distinct Count
        + index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | rex field=form\_data "passwd=(?<userpassword>\w+)" | stats count(userpassword)
      * Distinct Passwords Attempted
        + index=botsv1 sourcetype=stream:http form\_data=\*username\*passwd\* | rex field=form\_data "passwd=(?<userpassword>\w+)" | stats dc(userpassword)
    - Identifying the Executable Uploaded
      * Search for EXEs in stream:http
        + index=botsv1 sourcetype=stream:http dest="192.168.250.70" \*.exe
      * Search for EXEs in Suricata
        + index=botsv1 sourcetype=suricata dest\_ip=192.168.250.70 .exe
      * Hostnames v IPs
        + index=botsv1 sourcetype=suricata (dest="192.168.250.70" OR dest\_ip="192.168.250.70") .exe
      * When Destination and Destination IP are Different
        + index=botsv1 sourcetype=suricata (dest=imreallynotbatman.com OR dest="192.168.250.70") http.http\_method=POST .exe
      * Capturing the Source of the Executable
        + index=botsv1 sourcetype=suricata dest\_ip="192.168.250.70" http.http\_method=POST .exe
    - Determining the Hash of the Uploaded File
      * What is the MD5 Hash of the Executable Uploaded
        + <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>
        + <https://github.com/SwiftOnSecurity/sysmon-config>
      * What sourcetype Should I Start With
        + index=botsv1 3791.exe

earlier found executable

* + - * Sysmon
        + index=botsv1 3791.exe sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
        + index=botsv1 3791.exe sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational

return values like command issued to start a process execution as well as the parent command line

* + - * Isolating on MD5
        + index=botsv1 3791.exe sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational EventCode=1

find CommandLine and ParentCommandLine values

EventCode field in Sysmon and search for value = 1

* + - * Exploring the Sysmon Event
        + index=botsv1 3791.exe CommandLine=3791.exe

previous found file and corresponding command line input

* + - * + search EventDescription for process creation, directory where it was executed, parent command line, host Sysmon was run on
      * Putting it Together
        + index=botsv1 3791.exe CommandLine=3791.exe
        + | stats values(MD5)

stats command with values function return all matches to a specific field

* + - Identifying the File that Defaced the Web Server
      * Looking at Directional Flow of Data
        + index=botsv1 dest=192.168.250.70 sourcetype=suricata

determine who is communicating with web server

* + - * + index=botsv1 src=192.168.250.70 sourcetype=suricata

treat web server as source of traffic instead of the destination

would a web server initiate outbound traffic

only if an admin logged into web server and surfed the web, among other reasons

* + - * Pivot into Destination IP Addresses to View URLs
        + index=botsv1 src=192.168.250.70 sourcetype=suricata dest\_ip=23.22.63.114

pivot the external destination IP addresses

* + - * Do Web Servers Start A Conversation
        + look for http GET and POST

index=botsv1 imreallynotbatman.com sourcetype=stream:http

web data flow POST v GET

* + - * + Any Similarities between Suricata and stream:http?

index=main src=192.168.250.70 sourcetype=stream:http

pivot into interesting fields like URI

* + - Validating the File that Defaced the Web Server
      * What About Firewall Data?
        + index=botsv1 sourcetype=fgt\_utm "192.168.250.70"

IPs of interest

* + - * What Search to Start With?
        + specify direction of communication path but limit the search using the NOT command

index=botsv1 sourcetype=fgt\_utm "192.168.250.70" NOT dest="192.168.250.70" | stats count

* + - * + index=botsv1 sourcetype=fgt\_utm "192.168.250.70" NOT src="192.168.250.70" |stats count
      * Use Web Site Categorization to Filter
        + index=botsv1 sourcetype=fgt\_utm "192.168.250.70" NOT dest="192.168.250.70"

UTM devices rate/classify websites like web filtering gateway

pivot to view malicious sites

* + - * Firewall Gives Us Confirmation
        + index=botsv1 sourcetype=fgt\_utm "192.168.250.70" NOT dest="192.168.250.70" category="Malicious Websites"

check for recurring files or items

* + - Identify the Fully Qualified Domain Name of the System that Defaced the Web Server
      * <http://blogs.splunk.com/2015/08/04/detecting-dynamic-dns-domains-in-splunk/>
      * <http://10.10.98.171:8000/en-US/static/@962d9a8e1586:0/app/investigate_workshop/APT-http-gettrafficflow.png>
      * Using Found Firewall Events (Fortigate Firewall)
        + index=botsv1 sourcetype=fgt\_utm "poisonivy-is-coming-for-you-batman.jpeg"
      * What Other Data Sets Saw This File?
        + index=botsv1 dest=23.22.63.114 "poisonivy-is-coming-for-you-batman.jpeg" src=192.168.250.70

using file in question, search for other sourcetypes that contain data with file name

* + - * Using stream:http
        + index=botsv1 dest=23.22.63.114 "poisonivy-is-coming-for-you-batman.jpeg" src=192.168.250.70 sourcetype=stream:http

search for events with same URL and FQDN

* + - * What If No Filename Is Found?
        + index=botsv1 answer=23.22.63.114 sourcetype=stream:dns | stats values("name{}")

because IP of concern is known, search DNS and look for DNS events with same IP

stats command can return values of the name[] field to find domain

* + - Using OSINT to Identify Attacker Infrastructure
      * What IP Address is tied to Domains that are pre-staged to attack
        + <http://www.robtex.com/>
        + <http://www.threatcrowd.org/>
        + <http://www.virustotal.com/>
    - Using OSINT to Create Linkages Between Email and Infrastructure
      * <http://www.virustotal.com/>
      * <http://whois.domaintools.com/>
      * <http://dnswhois.info/>
    - Using OSINT to Identify Associated Malware
      * <http://www.threatminer.org/>
      * <http://www.virustotal.com/>
      * <https://www.hybrid-analysis.com/>
    - Using OSINT to Find Clues Pertaining to the Adversary
      * <http://www.virustotal.com/>
      * <http://www.rapidtables.com/convert/number/hex-to-ascii.htm>
* Ransomware Review in Splunk
  + Overview
    - construct a timeline of events, traffic flow diagram – Kill Chain does not translate well to commercial malware
  + Identifying the IP Address of a Victim System
    - Host Centric Log Sources
      * index=botsv1 we8105desk
        + hostname and index on specific date
        + see which sourcetypes have events that reference the hostname value
    - src v src\_ip v src\_host
      * index=botsv1 we8105desk sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
        + stats command in Sysmon against src and dest to determine IP address of host
        + src provides hostnames and IP addresses, essential to look across fields to ensure max data collected
  + Identifying Removable Media
    - Windows Logs, Registry Logs
    - <https://msdn.microsoft.com/en-us/library/windows/hardware/jj649944(v=vs.85).aspx>
    - <https://answers.splunk.com/answers/450042/finding-usb-and-removable-media-detection.html>
      * index=botsv1 sourcetype=winregistry friendlyname
        + start broad search, specifying index and sourcetype of winregistry
    - Tabular View of Host, User and Friendly Name
      * index=botsv1 sourcetype=winregistry friendlyname | table host object data
        + look at hosts and USB inserts to get an idea of where they are occurring
        + check data field for potential USB name
  + Identifying the Malicious File
    - Finding Sysmon Events for the Infected System on an External Drive
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk "d:\\" | reverse
        + assume malicious file upload reference file paths within D:\, or other paths other than C:\, as it is removable media
    - Refining Search to Find D:\ ONLY in Command and Parent Command Line
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk (CommandLine="\*d:\\\*" OR ParentCommandLine="\*d:\\\*") | table \_time CommandLine ParentCommandLine | sort \_time
        + take results and table field with \_time and sort oldest to newest
  + Identifying Suspicious Processes Executing
    - Finding EXEs in Sysmon
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational \*.exe
        + needs refinement
    - Refining Search
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational \*.exe | eval length=len(CommandLine) | table CommandLine length
        + add eval and table but some events do not have CommandLine field populated
    - Further Improving Search
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational \*.exe CommandLine=\* host=we8105desk EventCode=1 | eval length=len(CommandLine) | table CommandLine length | sort – length
        + added host, EventCode for Process Creation and the CommandLine value needing to present along with sorting output
  + Identifying File Server Connections from Infected Host
    - Identifying Source of Sysmon Events
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk
        + Registry data from Windows Systems as well as Sysmon provide insight into file shares
        + connections created by ransomware should create events
    - Identifying Sysmon Events Originating from Bob’s system
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk src=we8105desk.waynecorpinc.local
        + look at EventDescription field to see all events of Network Connect type
    - Applying Stats Command
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk src=we8105desk.waynecorpinc.local | stats count by dest\_ip | sort – count
        + check to see number of network connections host has made using dest\_ip
    - Clarifying Results by Using Registry sourcetype with suspected IPs
      * index=botsv1 sourcetype=winregistry host=we8105desk fileshare
        + see key\_path field entries that reference one of top internal IP addresses
        + all logs within key\_path field have same key\_path with only one IP address referenced
      * Another way to find the answer
        + index=botsv1 sourcetype=winregistry host=we8105desk explorer | stats count by registry\_key\_name | sort – count

limit search to host in question and focus on internal IP addresses (nature of file shares)

* + - Identifying Hostname of File Server
      * index=botsv1 sourcetype="XmlWinEventLog:Microsoft-Windows-Sysmon/Operational" 192.168.250.20
  + Identifying First Suspect Domain Visited by Victim
    - Isolating on DNS and the Source of the Query
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100
    - Further Filtering on A Records
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A
        + view DNS queries going out to different sites
    - Excluding Well Known Domains From Search
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A NOT (query{}=\*.microsoft.com OR query{}=\*.waynecorpinc.local OR query{}=\*.bing.com) | stats count by query{} | sort - 10 count
        + using conditions like AND OR and NOT
    - Further Refinement
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A NOT (query{}=\*.microsoft.com OR query{}=\*.waynecorpinc.local OR query{}=\*.bing.com OR query{}=\*.windows.com OR query{}=\*.msftncsi.com) | stats count by query{} | sort - 10 count
        + whois found returns
    - Further Elimination
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A NOT (query{}=\*.microsoft.com OR query{}=\*.waynecorpinc.local OR query{}=\*.bing.com OR query{}=isatap OR query{}=wpad OR query{}=\*.windows.com OR query{}=\*.msftncsi.com) | table \_time query{} src dest | reverse
        + google returns that are unknown to eliminate possibilities
  + Identifying Crypto Code Filename and Origin
    - Starting with stream:http
      * index=botsv1 sourcetype=stream:http src=192.168.250.100 | stats count values(url) by dest
        + use stats command to return all values
        + look through destination addresses and URLs

look for recurring suspect sites

suspicious files

* + - Confirm with Suricata
      * index=botsv1 sourcetype=suricata src=192.168.250.100 url=\* | stats count values(url) by dest
    - Corroborate with Google and OSINT
    - UTM Confirmation of Stream and Suricata
      * index=botsv1 sourcetype=fgt\_utm src=192.168.250.100 mhtr.jpg | table \_time src dest msg url action
    - Viewing Subsequent Activity As Seen on UTM
      * index=botsv1 sourcetype=fgt\_utm src=192.168.250.100 app="Cerber.Botnet" | reverse
        + signature and IoC
  + Identifying Parent/Child Processes
    - CommandLine
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational 121214.tmp
        + check Sysmon for specific process or file numbers
    - Relationship Between Process and Parent Process
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational 121214.tmp CommandLine=\* | table \_time CommandLine ProcessId ParentProcessId ParentCommandLine | reverse
        + use table command to see time along with process that was executed
        + reverse to view oldest to newest to view process of execution
  + Determine Which Signatures Specific to the Ransomware Alerted
    - Searching Suricate sourcetype
      * index=botsv1 sourcetype=suricata alert.signature=\*cerber\*
    - Suricata Signatures that Reference Suspect Files
      * index=botsv1 sourcetype=suricata alert.signature=\*cerber\* | stats count by alert.signature alert.signature\_id | sort count
        + use stats to view least or most common flagged events
  + Damage Assessment – Identifying Encrypted Text Files
    - <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon#events>
    - Initial Search
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk \*.txt
        + Target\_Filename value to view text files that are referenced in Sysmon events
    - Sysmon EventCode
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk \*.txt
        + EventCode and EventDescription to find correlation between values
        + File Create Time (Code 2)
    - TargetFilename
      * index=main sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk \*.txt EventCode=2
        + look at location of files, if they are within infected host
        + expand and see files that fall under hostname user and Sysmon
    - Finding Source Addresses
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk EventCode=2 TargetFilename="C:\\Users\\bob.smith.WAYNECORPINC\\\*.txt"
        + look for TargetFilenames that reference user, remove files and sub-directories that are unrelated
    - Stats of Infection
      * index=botsv1 sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational host=we8105desk EventCode=2 TargetFilename="C:\\Users\\bob.smith.WAYNECORPINC\\\*.txt" | stats dc(TargetFilename)
  + Damage Assessment – Identifying Distinct PDFs Encrypted
    - Search with IPs
      * index=botsv1 sourcetype=\*win\* pdf
        + need to use hostname, not IP
    - Searching with Correct Destination
      * index=botsv1 sourcetype=\*win\* pdf dest=we9041srv.waynecorpinc.local
        + hostname for fileshare, including Relative Target Name that includes files and Share Path
    - Identifying sourcetypes Associated with Search Values
      * index=botsv1 sourcetype=\*win\* pdf dest=we9041srv.waynecorpinc.local Source\_Address=192.168.250.100 | stats dc(Relative\_Target\_Name)
  + Identifying Redirection Post Encryption to a Domain
    - <https://splunkbase.splunk.com/app/2734/>
    - Quick Review
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A
    - Filter Out Known Domains
      * index=botsv1 sourcetype=stream:DNS src=192.168.250.100 record\_type=A NOT (query{}=\*.microsoft.com OR query{}=\*.waynecorpinc.local OR query{}=\*.bing.com OR query{}=isatap OR query{}=wpad OR query{}=\*.windows.com OR query{}=\*.msftncsi.com) | table \_time query{} src dest
* Build Timeline

Splunk 3