David Van Aulen

Brian Batinchok

Jordan Painter

IDPS Systems Intro Homework

1. **Identify a file or resource that can be used by Splunk for getting login information. Give both the name of the file and path.**

File Name: Security

File Path: %SystemRoot%\System32\Winevt\Logs\Security.evtx

We then converted this .evtx file to a .txt file for viewing in Splunk.

1. **What other files can be found in the path and for what?**

The System32 Logs directory contains numerous logs for many Windows applications. Some examples of log files found in the Logs directory include PowerShell, Windows Defender, Remote Desktop Services, and Network Access Protection.

1. **Setup ingestion for the file in question 1 into Splunk.**
2. **Make a report of login/logout attempts and the session duration. Use the command rex to extract fields.**

IDTasks: login is 4624, logoff is 4647

index=\* OR index=\_\* sourcetype=splunk | rex field=\_raw "(?ms)^(?:[^\\t\\n]\*\\t){3}(?P<IDTask>\\d+)" | search IDTask=4647 OR IDTask=4624

1. **Find if a session went into admin rights. You may use eval to create a new column using if.**

index=\_\* OR index=\* sourcetype=splunk ElevatedTokenValue=Yes

"Running with a full administrator access token," sometimes referred to as "running with an elevated access token," means that an application is allowed to use the user's full administrator access token, which includes the administrator security identifiers (SIDs).

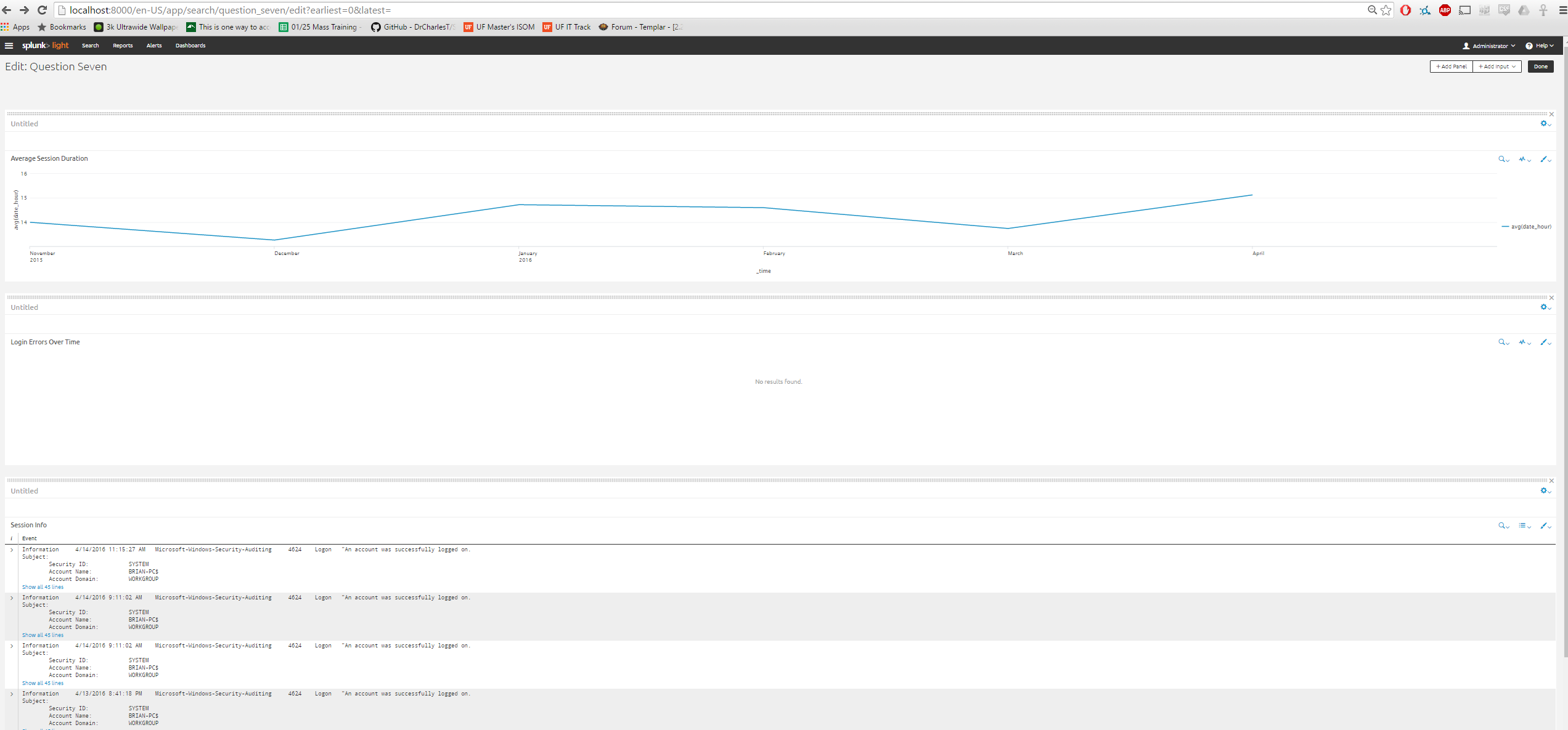
<https://technet.microsoft.com/en-us/library/dd759094.aspx>

1. **Find the average duration of a session, and the average user session duration. For this you will need to use stats avg.**

index=\* OR index=\_\* sourcetype=splunk | rex field=\_raw "(?ms)^(?:[^\\t\\n]\*\\t){3}(?P<IDTask>\\d+)" | search IDTask=4647 OR IDTask=4624 | stats avg(date\_hour)

Filters IDTask by login and logoff and then applies stats avg operation on the date\_hour field

1. **Make a dashboard which has a graph of: average session duration time chart, login errors time chart, event table with session info (\_time, user name, successful?, went into root?). Provide a screenshot of the dashboard and the queries used to generate each of the parts.**



**Average Session Duration:**

index=\* OR index=\_\* sourcetype=splunk | rex field=\_raw "(?ms)^(?:[^\\t\\n]\*\\t){3}(?P<IDTask>\\d+)" | search IDTask=4647 OR IDTask=4624 | timechart avg(date\_hour)

**Login Errors Timechart (this is zero because I do not have a password to log in):**

**4625 IDTask is a login error**

index=\* OR index=\_\* sourcetype=splunk | rex field=\_raw "(?ms)^(?:[^\\t\\n]\*\\t){3}(?P<IDTask>\\d+)" | search IDTask=4625| timechart avg(IDTask)

**Event Table of Session Info:**

index=\* OR index=\_\* sourcetype=splunk (ElevatedTokenValue=No OR ElevatedTokenValue=Yes) Username="BRIAN-PC$" (IDTask=4624 OR IDTask=4625)

1. **Explain what settings or information you could use from this dashboard to create alerts to detect potential intruders. Which IPtables command(s) would you use to react if the alert fires? Assume that you can get all the information needed from the log file event.**

--Login errors over time can track the amount of unsuccessful logins by a user, alerting to a possible brute force attack.

--Average session duration can show the amount of time that a user is logged in for extended periods, possibly leaving their stations vulnerable to intrusion from the station (someone coming up and accessing sensitive data).

--Session Info can show the name of a user, successful/unsuccessful logins, and the domain on which they reside. Abnormal amounts of login attempts can alert an administrator of potential hacking of information.

IPtables commands to react to alerts:

* The DROP command can be used to DROP the MAC address associated with the alert from the table to stop the attacks
* The LIMIT command can be used to limit the time the machine is allowed access. This, at the very least, can shorten the attack until there is a solution to block/stop the attack

1. **Use IPtables MAC address filter to allow login only from specific local machines. You may write any valid/non-valid MAC address in your command.**

--Iptables -A INPUT -s “AB.09.B8.C9” -j DROP will turn away a single machine that is not valid or wanted

--Iptables -A INPUT -s “C1.33.61.67” -j ACCEPT will allow for that local machine to go through, would have to input every MAC address desired for allowed login

--Iptables -I *chain* -j ACCEPT will allow for that local chain to be added to the table for allowed access

--Iptables -t *table name* -F as is can flush all allowed rules and using the accept line to input the MAC addresses desired will allow for only those MAC addresses to have access. Flush can also be used with -L *list name* after the -F to delete the list that is not wanted instead of flushing the entire table

1. **Specify a logrotate that could handle this file to rotate every day, keeping only one week of files. Anything older must be zipped into a tar file. Once past one month it should be deleted.**

/var/log/btmp {

weekly //weekly

missingok //allow for initial null

rotate 4 //rotate every 4 weeks, aka a month

compress

compressext tar //compress to tar

}