**Stuart Hohm**

**Personal Lab**: Setting up VMs, throwing, and detecting attacks.

**Required Software**:

VMWare Workstation (used a free trial for this project) - [link](https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html)

Windows free VM - [link](https://developer.microsoft.com/en-us/windows/downloads/virtual-machines/)

Ubuntu 22.04.1 - [link](https://releases.ubuntu.com/22.04.1/ubuntu-22.04.1-live-server-amd64.iso)

LimaCharlie - [link](https://limacharlie.io/)

**Set-up and Config VMs Procedure:** [link](https://blog.ecapuano.com/p/so-you-want-to-be-a-soc-analyst-part)

Document these during VM set up

Subnet IP: 192.168.5.0

Gateway IP: 192.168.5.2

Linux VM IP: 192.168.5.128/24

Set up an easy username and password since this is just for lab purposes

Server name: Attack

Username:user

Password: password

After completing the Set-up and Configures Procedure take a snapshot of both machines to reset incase of error.

**Generating the C2 Payload Procedure:** [link](https://blog.ecapuano.com/p/so-you-want-to-be-a-soc-analyst-part-ea2)

1. Generation of a C2 or Command and Control payload is done by downloading sliver.
2. A mock server address is created on the Linux VM using the VMs IP and a simple python script.
3. Then on the windows vm in the command prompts as an admin a download request is made to download the C2 payload from the Linux hosted IP and is instructed the download location.
4. Back in the Linux VM I check to see if the connection is active between the two VMs.
5. Then we do some snooping from the Linux VM, getting a look at where I was installed in the system, what threat protection software they have enabled, as well as who the user is via whoami.
6. Since I know what to look for because I made the payload, I am able to find it both on the Windows VM in the downloads folder, and through LimaCharlie.
7. Additionally some of the snooping I was doing was flagged under SENSITVE\_PROCESS\_ACCESS.
8. Then I ran a credential dump from the linux server to generate an event to use as a template to build a detection rule via LSASS.exe.
9. The credential dump called by LSASS.exe appeared in the LC timeline and was used to create an alert for LSASS access.
   1. The rule is not proof to false positives, but does trigger on a second attempt at LSASS.exe and the rule detection is as follows:

event: SENSITIVE\_PROCESS\_ACCESS  
op: ends with  
path: event/\*/TARGET/FILE\_PATH  
value: lsass.exe

1. The next malicious thing I want to test is a rule that not only detects an event, but also kills the event.
   1. These need to be very robust against false positives because you do not want to be canceling critical functions across your company due to poor implementation and rule construction.
   2. For this reason the rule will be targeting the command vssadmin.exe delete shadows /all
   3. This is chosen because it is commonly used in ransomware attacks and rarely sees use otherwise so the likelihood of a false positive is low.
   4. To give a template, the command is again run from the Linux VM through the C2 payload on the Windows VM.
   5. The event appears in the timeline and is noticed by LimaCharlie by default, but no action is taken other than the alert.
   6. The event is used to build a D&R rule to terminate this event and the task calling the action due to it likely being a payload.
   7. The Response is as follows:  
      - action: report  
       name: vss\_deletion\_kill\_it  
      - action: task  
       command:  
       - deny\_tree  
       - <<routing/parent>>
2. When the attack is run again, the detection and response successfully fire and the payload task is terminated, breaking the connection between the infected windows VM and my linux server.
3. To continue a connection needs to be reinstated between the two VMs.
4. This is done by simply executing the C2 payload .exe on the infected machine, meaning a malicious actor could have this payload insert itself amongst start up applications so that it fires up any time the machine is started.