Process of Elimination

The first steps of an investigation is to think about what the end goal is and how you know about the current situation. Every investigation requires some context to start. If all you know about an incident is “malware infected this computer” there needs to be some context to start such as what time? What machine? What account? What network? What malware? How did the user become aware they were infected.

From these facts you are able to work backwards and build the puzzle. The example I will use here is for the Conti Ransomware lab from THM. In this investigation the situation given to me was the machine was infected with malware and we are aware of the incident due to denial to the Exchange admin portal and Outlook, While investigating this issue the user found a readme.txt which was the ransomware note. This is the piece of evidence and context of the investigation.

**STEPS([tldr](#Theinvestigation))**

1. Evidence being the “readme.txt” was created from the malware. This is the first notable we will search for. Unaware of anything other than this simple name of the file. We search all index’s and source’s for this file name. ( index=\* source=\* “readme.txt”)
2. Upon finding the file we note everything related to it, this includes: Timestamp of alert, timestamp in data(if there is one), machine, sourcetype, source, account, any other notables. This allows us to build a picture
3. Sysmon or eventlogs should contain information about this image(file) being created. We will search for the parent image and the target image. This should create a timeline of how this came to be, again we look for any notables that can correlate puzzle pieces. This could be: timestamps, machines, filepaths, PIDs, GUIDs, accounts  
   This lead us to find cmd.exe had created the file. Cmd.exe was located within “c:\Users\Administrator\Documents\cmd.exe” which is an incorrect file path for a windows binary. This would usually reside in system32. If unsure GOOGLE EVERYTHING
4. This allowed me to trace its parent process which led to a suspicious powershell script. We know if a threat actor has access to powershell there is a range of things that can be done. One of which is persistence, from here we need to follow computer artifacts as we cant rely strictly on names and hunches. I found the proccessguid which helped me follow the trail, command below

index=\* sourcetype="wineventlog:microsoft-windows-sysmon/operational" "72893ba8-1125-6139-5d00-000000000c00"

1. From here It is a good idea to search for windows event logs IDs to attempt to see if suspicious activity has occurred.   
   4688 – process creation “identify execution”

4720 – New user created

4624 – successful logon

4625 – failed logon

These among others allow us to get a better picture of persistence of suspicious activity. We know “net user /add” is the powershell command to create a user.

1. [Lsass.exe](#lsass) allows us to see a range of information click the hyper link to view them. From lsass we could determine where the ransomware had accessed system hashes.
2. The investigation ended with finding the webshell that was deployed. The way I found the webshell was purely from the format of the question and this took and understanding of IIS (inetpub) web servers and purely trying to identify an anomaly. Purely by a hunch I found the webshell which isn’t scientific and would of required googling to identify without context given to me by the format of the question. The /owa/auth was the location of the webshell and googling information after the investigation gave me some context into why it could of effected the exchange server which can be found from the [hyperlink.](#owaauth)
3. The investigation was completed and the methodical steps got me there paired with the format of the questions. The loose methodology was:
   1. Find the notables/evidence given to me from context of the incident, in this case a ransom note
   2. Identify where that was placed(filepath, machine), what made it(process/binary), what it is related to(machine, account, network, time, how many), the timestamp
   3. Work backwards from all discovered notables to find the puzzle pieces

**Important Knowledge expected to be known prior or learnt during the investigation.**

----------------------------------------------------------------

LSASS.exe is a Windows process responsible for managing security policy and enforcing access controls on the operating system. It plays a crucial role in verifying user logons, handling password changes, and creating access tokens. Here are some key functions of LSASS

* **User authentication**: LSASS verifies user credentials (username and password) during logon, ensuring that only authorized users access the system.
* **Security policy enforcement**: It enforces the security policy defined on the system, controlling access to resources and ensuring compliance with organizational security settings.
* **Access token creation**: LSASS generates access tokens for authenticated users, granting them the necessary permissions to access system resources.
* **Windows Security Log**: It writes events to the Windows Security Log, providing a record of security-related activities on the system.

In normal operation, LSASS.exe runs in the system32 directory (c:\windows\system32 or c:\winnt\system32) and is a critical system process. Its high CPU usage can indicate issues with the system’s security configuration or potential malware infections.

---------------------------------------------------------------------

In IIS (Internet Information Services), /owa/auth/ refers to **the authentication endpoint for Outlook Web Access (OWA) virtual directory**.

OWA is a web-based interface for accessing Exchange Server mailboxes, and the /owa/auth/ URL is responsible for handling authentication requests for users trying to access their mailboxes through OWA.

When a user navigates to the OWA login page, the request is routed to the /owa/auth/ endpoint, where the IIS server authenticates the user using the configured authentication methods, such as Forms-Based Authentication (FBA), Windows Authentication, or Basic Authentication.

The /owa/auth/ endpoint is typically configured to use a specific authentication scheme, which is defined in the OWA virtual directory settings. For example, in Exchange Server 2010, you can configure Forms-Based Authentication (FBA) for OWA by setting the BasicAuthentication property to $false and the WindowsAuthentication property to $true.

In IIS Manager, you can access the OWA virtual directory settings by navigating to the Server Configuration > Client Access > Outlook Web App > Properties > Authentication tab.

The /owa/auth/ endpoint plays a critical role in **securing access to Exchange Server mailboxes through OWA and ensuring that only authorized users can access their mailboxes**. Misconfigurations or issues with this endpoint can impact OWA authentication and access.