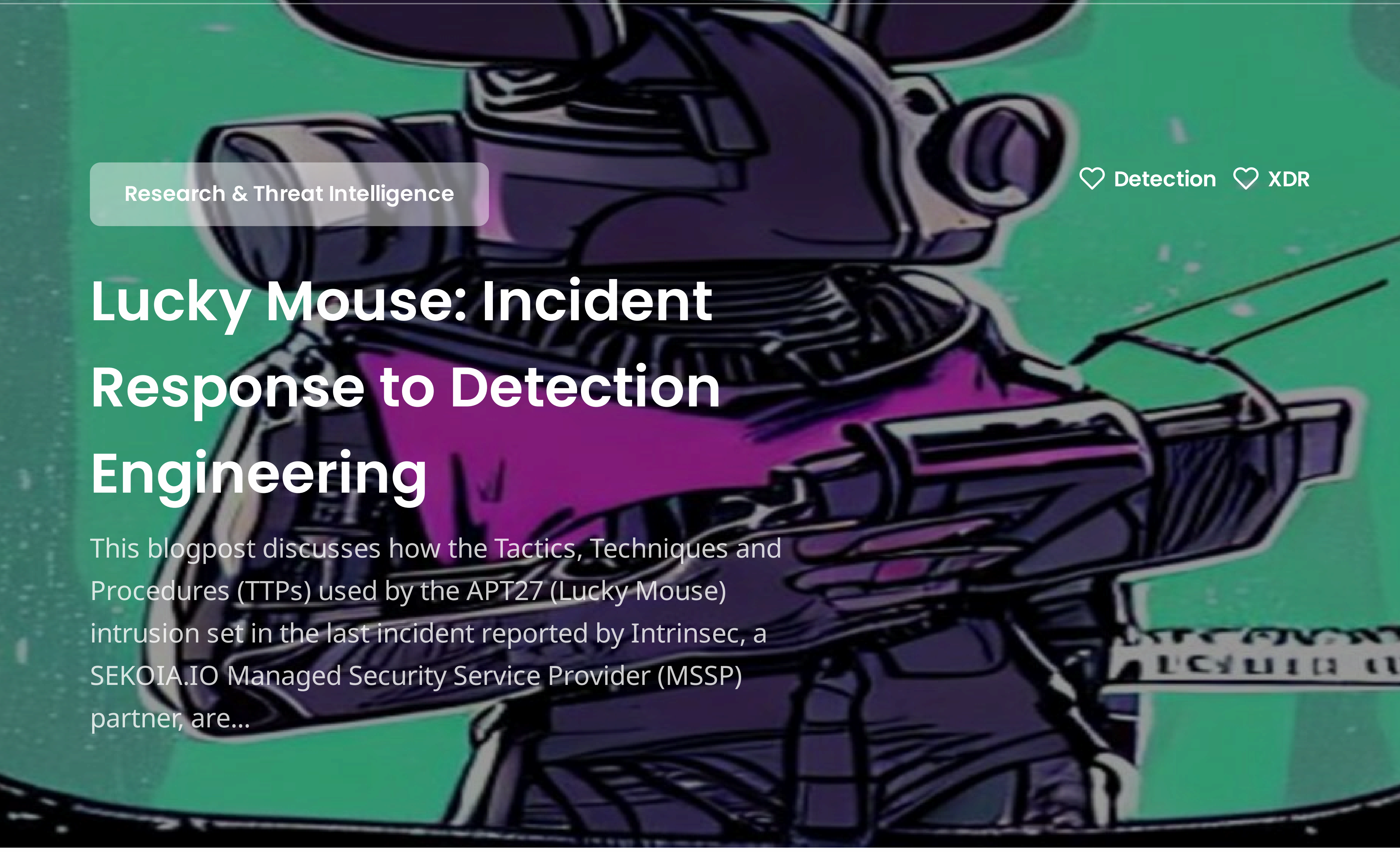

Research & Threat Intelligence

Detection XDR

Lucky Mouse: Incident Response to Detection Engineering

This blogpost discusses how the Tactics, Techniques and Procedures (TTPs) used by the APT27 (Lucky Mouse) intrusion set in the last incident reported by Intrinsec, a SEKOIA.IO Managed Security Service Provider (MSSP) partner, are...

Guillaume C. and Sekoia TDR
December 1 2022

Read it later 10 minutes reading

Table of contents

- Interactive shells & process spawning techniques
- Windows Defender Exclusions
- Exchange mailboxes export
- Usage of SysInternals tools
- Ntdsutil to dump ntds.dit database
- RAR to compress and encrypt folders & files

This blogpost discusses how the Tactics, Techniques and Procedures (TTPs) used by the APT27 (Lucky Mouse) intrusion set in the last incident reported by Intrinsec, **a SEKOIA.IO Managed Security Service Provider (MSSP) partner**, are detected using SEKOIA.IO. We therefore publicly share a few detection rules available in our platform, and explain the logic behind each of them. For that purpose, we specifically focus on our telemetry and how we sometimes had to adapt the rules to make them more specific or more generic. As we use SIGMA, the rules can be easily adapted to another environment, the only specificity we have is the use of Elastic Common Schema (ECS) for the detection rules fields. For that reason, if a SEKOIA.IO rule is relatively similar to an existing SIGMA rule, we will link the SIGMA rule instead.

Interactive shells & process spawning techniques

A few general techniques can be detected based on Intrinsec's observations.

The first one is the probable use of **wmiexec.py** from the Impacket suite (or a tool that produces similar patterns), used by APT27 to execute most commands:

```
cmd.exe /Q /c [………] 1> \\127.0.0.1\ADMIN$\__[UNIX_EPOCH_DATETIME] 2>&1
```

Although this pattern can be detected in different ways, Sekoia.io provides one detection rule specific to WMI related activities, based on the *"wmiprvse.exe"* parent process. It is tough to build a good generic rule here without having several false positives, which is why our rule has several conditions. Each condition is rather specific but the rule covers most of the tool's execution even when it is not executed "as is".

Based on Sekoia.io telemetry, the rule raises no false positive (FP) and requires no specific configuration other than logging command-line activity, hence the Effort Level set to "elementary".

A second technique used by APT27 throughout their attack is the following:

```
call create cmd
```

It is a simple, well-known technique to create a new process using "wmic".

Sekoia.io provides a simple rule solely based on the *"process call create"* command line parameters. The only FP observed in our telemetry on Windows was the process *"crashpad_handler.exe"*. Since we are vendor-agnostic in our rules as well, we also had to filter out *"/bin/rm"* that triggered several false positives. While the filter on these two false positives are still a weakness in our rule, it avoids lots of alerts and prevents alert fatigue of SOC analysts.

You may be aware of similar SIGMA, publicly available, rules to detect this activity and wonder why we did not link them instead. Most rules have two major blind spots: the process name is "hardcoded" in the rule and there are whitespaces before or after at least one of the command line parameters. This is great to avoid FPs and most of them definitely matches APT27 current activity as documented by Intrinsec. However, the "hardcoded" process name means the rule relies on an unrenamed *"wmic.exe"* process or "OriginalFileName" always catching *"wmic.exe"* being renamed. The whitespaces could lead to missing a non-default usage of wmic commands (usage of quotes before and after each parameter for instance).

Although the attack described by Intrinsec could be detected mainly with these two rules, we will still study each part of the attack in depth to try detecting each technique.

Windows Defender Exclusions

A classic technique, APT27 used PowerShell to disable Windows Defender, more specifically to exclude a folder from Windows Defender scope, using the following command:

```
ExclusionPath C:\Windows\temp
```

There are many ways to disable or tamper Windows Defender. The good news is that there are also many ways to detect that, which means multiple rules can be triggered.

The rule named "Windows Defender Disabled" is triggered based on command line specific strings of or if any change occurs in the Windows Defender registry key. Of note, it does not look at ScriptBlockText, as this would raise significantly more FPs. More specific rules related to ScriptBlock logging suspicious keywords are available in the Sekoia.io platform. Furthermore, we provide other rules for Windows Defender being tampered and for the same commands as this rule but base64 encoded – the detection capability uses the built-in SIGMA "|base64" value modifier.

Depending on your environment, for instance, if using PyCharm or some Anti-Virus / EDRs solutions (BitDefender for example), a few FPs can still occur, but those are easy to filter which is why the Effort Level is "intermediate".

The SIGMA rule titled "Windows Defender Exclusions Added"[1] (similar to our rule) is based on the Windows Defender EventID 5007, narrowed down to specific values. This rule is great since it directly relies on Defender itself, not the command lines.

Exchange mailboxes export

According to Intrinsec, APT27 used the following command to export mailboxes from Exchange servers:

```
powershell -exec bypass -command Add-PSSnapin
Microsoft.Exchange.Management.PowerShell.SnapIn;Get-Mailbox | format-table
Name,WindowsEmailAddress
```

s been used by several intrusion-sets.

ith no observed false positives to date. However this
jex for the ScriptBlockText pattern and could be
54 encoding for instance. Therefore we have other rules
ng for this command line as well, just like our Defender

Usage of SysInternals tools

In the described activity, APT27 used two SysInternals tools. The first one is PsLoggedOn, to identify where specific users are logged in. While this tool is well-known in the community as being used for offensive purposes, Sekoia.io analysts did not see this tool being used by intrusion sets before. Therefore not much research time has been spent on this tool and we are unable to provide an efficient rule, except just detecting the process name or the accepteula parameter which are both easy to bypass, for the time being. We will update our GitHub repository if we find a proper way to detect this tool.

The second Sysinternal tool used by APT27 is ProcDump, often used by intrusion-sets to dump LSASS's process memory to then extract credentials. The command used is :

```
C:\Windows\Temp\error.log -accepteula -ma lsass.exe
c:\windows\temp\error.dmp
```

As you may notice, it was renamed but still has the same command-line parameters. Therefore we have two rules to detect such behaviour. The first rule detects *"accepteula"* combined with *"-ma"* and a *".dmp"* file or *"lsass"* as it is often used. It can be bypassed but raises no false positives based on our telemetry, which is a huge quick win.

The second rule is recent and was built following the Intrinsec blogpost, as we found the logic quite interesting: the usage of *"-accepteula"* with a binary neither originating from the Sysinternals toolsuite, nor known as legitimate. With the provided rule we observed no false positives, based on ~40 days of telemetry.

In addition to our detection rules, we retrieve hashes for ProcDump releases automatically, which allows us to detect its usage (if not modified), even when renamed. We only advise to do that when the tools are not often used for legitimate purposes.

Ntdsutil to dump ntds.dit database

us ntdsutil to dump *"ntds.dit"* database on Domain

and:

```
full c:\\windows\\temp\\winstore\\ quit quit
```

While the command line itself could be detected, there is a specific Windows EventID for when the *"ntds.dit"* database is created: EventID 325, from the Event Provider ESENT. This EventID also raises events when other databases are created therefore a looking for the string *"ntds.dit"* is necessary.

To detect *"ntdsutil"* usage, a general rule can be made on the usage of *"ntdsutil.exe"* process, like this SIGMA rule. It's fine to have this rule, but having the other SIGMA rule with the EventIDs is really necessary. Even if EventIDs can be disabled, we think it is always best to rely on them rather than command lines or process name since it is even easier to rename ntdsutil.exe before using it. With the two rules, both scenarios are covered.

Although legitimate backup operations can sometimes trigger these rules, we rarely observe false positives and strongly advise to activate the rules.

RAR to compress and encrypt folders & files

According to Intrinsec, RAR was used to compress and encrypt folders & files using a password with several commands, such as:

```
rar.exe a -r -y -hp[PASSWORD] -df c:\windows\temp\error.rar
c:\windows\temp\winstore\
```

Sekoia.io provides a simple but efficient rule, relying on whitespaces before and after the *"a"* command line argument to avoid false positives but, as written earlier, could be a blind spot

itespaces, the rule Effort Level is set to only

this command being used within a PowerShell script

t needs the default name *"rar.exe"* to prevent too many

g with Chisel

Lucky Mouse also leveraged Chisel, an open-source tunneling tool, renamed *"veamGues.exe"*, to run a Chisel server listening on port 9080 and allowing Chisel clients to access the SOCKS5 proxy:

```
cmd /c c:\Windows\Temp\veeamGues.exe server -p 9080 –socks5
```

Chisel was only renamed but the attackers did not change the default command line, which is quite unique (or at least relative to most SOCKS5 tunneling tools). Therefore, our rule is mainly based on the usage of *"socks4"* or *"socks5"* in the command line, *"socks"* alone giving way too many false positives.

We still have some false positives in our telemetry with this rule, related to the fact that developers used *"chrome.exe"* for instance to set up the SOCKS proxy. When doing so, it triggers the following command:

```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" --user-data-dir="%USERPROFILE%\proxy-profile" --proxy-server="socks5://localhost:9090"
```

Therefore it triggers our rule as well. It could be a True Positive and it's often related to a few, if not single, user(s) therefore we recommend activating this rule and adding some filters depending on your environment. Again, a simple but efficient rule!

Conclusion

Incident response publications like Intrinsec's one are a gold mine for defenders and can be used to improve or create detection rules. This blogpost shows our detection logic when looking at this kind of publication and also gives information on our telemetry to ease the integration of our rules without having unexpected false positives within your environment.

Sekoia.io shared detection rules

es/host/data_compressed_with_rar_with_password.yml

es/host/disable_windows_defender.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/impacket_wmiexec.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/powershell_exchange_snapin_mailbox.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/non_legit_use_eula_parameter.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/procdump_args.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/socks_tunneling_tool.yml

https://github.com/SEKOIA-IO/Community/blob/main/sigma_rules/host/wmic_process_call_create.yml

[1]https://github.com/SigmaHQ/sigma/blob/master/rules/windows/builtin/windefend/win_defender_exclusions.yml. Accessed on december 2022.

You can also read other blog post :

- Hunting for IoCs: from singles searches to an automated and repeatable process
- XDR detection engineering at scale: crafting detection rules for SecOps efficiency
- Engineering detection around Microsoft Defender
- An insider insights into Conti operations – Part Two
- Ideation process at Sekoia.io
- XDR vs Ransomware
- Command & Control infrastructures tracked by Sekoia.io in 2022
- Peeking at Reaper’s surveillance operations

Discover our:

- CTI platform
- XDR platform
- SOC platform
- Tools for SOC analyst
- SIEM solution

What's next

Calisto show interests into entities involved in Ukraine war support

Calisto (aka Callisto, COLDRIVER) is suspected to be a Russian-nexus intrusion set active since at least April 2017. Although...

Felix Aimé, Maxime A. and Sekoia TDR

How to use Sekoia.io indicators in Microsoft Sentinel ?

Since May 20221,2, Sekoia.io indicators can be integrated into Microsoft Sentinel. In this blogpost, we will cover how to...

SEKOIA.IO

The DPRK delicate sound of cyber

This blogpost aims at contextualising and analysing trends pertaining to cyber malicious activities associated to the Democratic People's Republic...

Sekoia TDR

Comments are closed.

Trending topics

tform

Detection Engineering