Splunking with Sysmon Part 4: Detecting Trickbot

By Hurricane Labs

Published: 2020-11-12 · Archived: 2026-04-05 15:56:34 UTC

Trickbot and Ryuk

With the recent outbreak of Ryuk in hospitals, detecting the precursors to the ransomware has become a more visible priority. Ryuk has a history of being deployed after an enterprise has been compromised by Trickbot. The problems with detecting Ryuk is that once it is detected, it is often too late to save anything. The key is to detect Trickbot or any other malware attackers use before your data starts being encrypted.

This Splunk tutorial will cover the methodology I used to develop and test the detections as well as how to implement and tune them. Also, in case you missed the previous parts of my Splunking with Sysmon tutorial series, make sure to check out parts 1, 2, and 3 too!

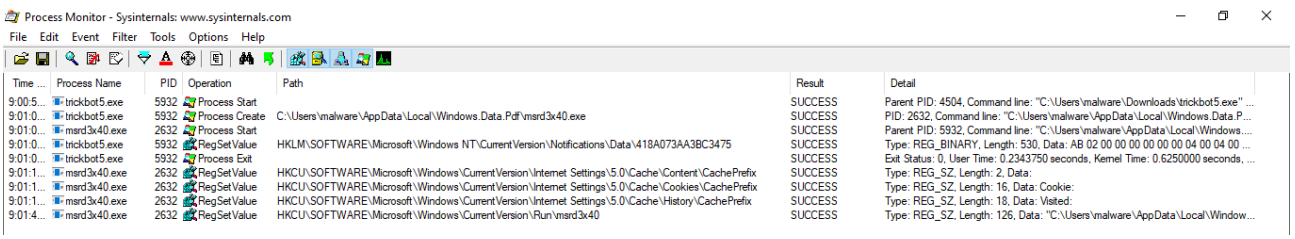
Trickbot Hunting

Finding Trickbot samples is not hard to do; there are many sources and samples available. I tested 7 different .exe samples that all had been submitted within 3 days of my testing. I ran each sample on my home lab with access to the internet enabled and Sysinternal Process Monitor (procmon) running to monitor what the executable was doing. I segregated my home lab from my personal network to reduce the risk of any malware spreading; please be safe if you want to recreate my testing.

To determine how I'd approach a detection, I divided the analysis of the Trickbot samples that I tested into two different categories. The first category included the samples that fully executed and established a persistence mechanism. The second category's samples were more evasive, but they did not establish any form of persistence.

Persistent Trickbot

The Trickbot samples I analyzed that established persistence had a few different ways that they executed, but they always used Registry Run Keys to establish a persistent hold on the infected system. The simplest sample wrote a file to the users Local Appdata folder and created a run registry key to execute that file on boot. It also did a time stomp to change the file creation time on the executable.



The keys I took away from the procmon evaluation was that the initial process creates a file, possibly timestomped, sets a registry run key on it, and works from the child process–not the initial execution. These keys can be seen when tracking the process in Splunk logs as well.

Evasive Trickbot

It was much harder to detect what the evasive trickbot was doing. The process would start, and then each one quickly moved into wermgr.exe and wrote to a file while making outbound connections. The wermgr.exe process never created any child processes or moved into another process; it would create an outward network connection every 5 or so minutes, but appeared to be waiting for more instructions.

Detecting Initial Execution

Detecting the Initial Execution of Trickbot was more reliable than the persistence, but it also required a little more work. All of the Trickbot samples I tested had an OriginalFilename in the PE Header that did not match the file that was executed. Since these processes are created from executables in the User folders, I started looking at Process Creation events where the file_name did not match the OriginalFilename, which fits the Mitre Technique

T1036 (Masquerading). Evaluating those files does result in a decent number of False Positives, but with a little bit of time to exclude them via a lookup, you can detect the programs that should not be running in your environment. This can be somewhat noisy, but it also detected every sample I encountered.

Copy to Clipboard

```
source=xmlwineventlog:microsoft-windows-sysmon/operational EventCode=1 process_path="C:\\Users\\*"
    NOT (OriginalFileName="-" OR OriginalFileName="?" OR [|inputlookup renamed_tools.csv])
| where process_name!=OriginalFileName
```

```
| table OriginalFileName, process_name, process_path, CommandLine, Hashes, Computer _time
```

Tuning the renamed_tools.csv lookup is most easily done by running the search, deduped by OriginalFileName and process_name, over a week to add all processes to the lookup table. After that take some time to go through the lookup and ensure that all entries are expected and clean. If you are unsure you can run the hash through a OSINT tool to determine more information about the process. Then remove all columns from the lookup table but the fields you intend to exclude by.

Detecting Persistence

Only 3 of the 7 samples I tested were able to establish persistence on my lab machine. Each one that did, did so via the same method. They added a Registry run key, Mitre Technique T1547.001 (Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder), which would cause the executable to run when the user logs on to the machine. This detection is similar to the initial execution detection, where it looks for Processes that originate from the User folder that modifies Registry Run Keys. This also needs a little tuning to reduce the number of False Positives, but not as much as above.

Copy to Clipboard

```
source="xmlwineventlog:microsoft-windows-sysmon/operational" EventCode=13
```

```
TargetObject="*SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\*"
```

```
NOT ([| inputlookup registry_run.csv])
```

```
| table RuleName, Image, TargetObject, Details, Computer, _time
```

Tuning the registry_run.csv lookup is similar to the renamed_tools lookup. Deduping will not work as well, and you may need to add wildcards to the lookup table as the TargetObject will contain the user SID and Details and Image may contain the user name. You could also add the process_name field to the lookup and dedup via it, but there will be more risk to commonly named processes. Make sure to remove all columns except Image and TargetObject (and possibly Details if you intend to exclude by it) from the lookup table when tuning is finished.

Conclusion

This was honestly a lot of fun and interesting to watch the execution of a variety of Trickbot samples and see how they initially run. While there is a little work needed to get the detections to an alertable state, it is well worth it if you can catch Trickbot or other malicious processes before they have a chance to cause more damage.

About Hurricane Labs

Hurricane Labs is a dynamic Managed Services Provider that unlocks the potential of Splunk and security for diverse enterprises across the United States. With a dedicated, Splunk-focused team and an emphasis on humanity and collaboration, we provide the skills, resources, and results to help make our customers' lives easier.

For more information, visit www.hurricanelabs.com and follow us on Twitter @hurricanelabs.
