# Winnti analysis

github.com/br-data/2019-winnti-analyse/

br-data

br-data/**2019-winnti-analyse**

Scripts and rulesets for analysing the Winnti malware

1 Contributor · 0 Issues · 24 Stars · 1 Fork

For a number of years now, a group of professional hackers has been busy spying on businesses all over the world: Winnti. It is believed to be a digital mercenary group controlled by China. For the first time, in a joint investigation, German public broadcasters BR and NDR are shedding light on how the hackers operate and how widespread they are.

Read the full article on hackers for hire, conducting industrial espionage, here:

> **BR24**: Attacking the Heart of the German Industry.

## Background

The search for affected company networks is mostly build around so-called **campaign identifiers**. In some instances, Winnti operators wrote the names of their targets directly into the malware, obfuscated with a rolling XOR cipher. In a first step, we tried to verify the information we were provided with, using a (not very good) python script. We then used yara rules to hunt for Winnti samples. The yara rules we used are included in this repo, hopefully they prove useful to other researchers.

Another way of finding networks with Winnti infections is this Nmap script by the Thyssenkrupp CERT.

## Analysis

An execellent script for extracting the configuration details from a Winnti sample was written by Moritz Contag. He thankfully allowed us to share it. Here is how to use it:

## Requirements

The script requires `lief` in version 0.9 to be installed and thus is currently tied to Python 2.7. The dependency can be installed running `pip` on the command line:

```
pip2 install -r requirements.txt
```

## Usage

To extract the configuration of multiple Winnti samples, simply pass the directory to the script. The script will also recurse into subdirectory and blindly try to parse each file it encounters.

The script does not try to identify Winnti samples and might produce incoherent output if the sample looks too different. Currently, it tries to parse configuration information stored in the executable's *overlay* as well as *inline* configurations indicated by a special marker. Further, it also tries to repair broken or "encrypted" files before processing them.

It is recommended to name the samples according to their, e.g., SHA-256 hash for better identification.

To scan a directory called `samples`, simply invoke the script as follows:

```
$ python2 parse.py ./samples

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

./9c3415507b38694d65262e28f73c3fade5038e455b83d41060f024403c26c9ee: Parsed
configuration (overlay).

- Size:    0x50E
- Type:    exe
- Configuration:

        +0x000:  ""
        +0x304:  "1"
        +0x324:  "shinetsu"
        +0x356:  4B A0 D6 05
        +0x3C2:  "HpInsightEx.dll"
        +0x3E2:  "kb25489.dat"
        +0x402:  "HPSupportService"
        +0x442:  "HP Insight Extension Support"
        +0x50A:  A9 A1 A5 A6

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

./585fa6bbc8bc9dbd8821a0855432c911cf828e834ec86e27546b46652afbfa5e: Parsed
configuration (overlay).

- Size:    0x048
- Type:    dll exe
- Exports: #3
           GetFilterVersion
           HttpFilterProc
           TerminateFilter

- Configuration:

        +0x000:  "DEHENSV533-IIS"
        +0x020:  "de.henkelgroup.net"
        +0x044:  99 DE DF E0
```

## Acknowledgments

- Moritz Contag for writing the great script and allowing us to share it
- Silas Cutler who helped us a great deal to corroborate our findings

## Contact

BR Data is a data-driven investigative unit at the German public broadcaster Bayerischer Rundfunk. We are a team of journalists, developers and data scientist. We specialize in data- and document-driven research and interactive storytelling.

Please send us your questions and feedback:

- Twitter: @br_data
- E-Mail: data@br.de