CYBER ATTACK ON UKRANIAN POWER GRID AND YARA TOOL

A REPORT

Submitted by

MATSA BHARGAV

[RA2111030010153]

Under the Guidance of

Dr. D. Deepika

Assistant Professor, Department of Networking and Communications

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

with specialization in CYBER SECURITY

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603203

APRIL 2024

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603203

BONAFIDE CERTIFICATE

Certified that this project report **"Cyber Attack on Ukranian Power Grid and YARA Tool"** is the bonafide work of "**Matsa Bhargav**" of III Year/VI Sem B. Tech (CSE) who carried out the mini project work under my supervision for the course 18CSE386T PENETRATION TESTING AND VULNERABILITY ASSESSMENT in SRM Institute of Science and Technology during the academic year 2023-2024(Even sem).

SIGNATURE	SIGNATURE
Dr. D. Deepika	Dr. Annapurani Panaiyappan K
Assistant Professor	Professor and Head
Networking and Communications	Networking and Communications

CASE STUDY ON "CYBER ATTACK ON UKRANIAN POWER GRID AND YARA TOOL"

EVEN Semester (2023-2024)

Course Code & Course Name: 18CSE386T – Penetration Testing and Vulnerability Assessment

Year & Semester : III/VI

Report Title : Cyber Attack on Ukranian Power Grid and YARA Tool

Course Faculty : Dr. D. Deepika

Student Name : Matsa Bhargav[RA2111030010153]

Evaluation:

S. No	Parameter	Marks
1	Problem Investigation & Methodology Used	
2	Tool used for investigation	
3	Demo of investigation	
4	Uploaded in GitHub	
5	Viva	
6	Report	
	Total	

Date:

Staff Name:

Signature:

TABLE OF CONTENTS

Introduction

The Cyber attack on the Ukrainian power grid stands as a stark reminder of the vulnerabilities inherent in critical infrastructure systems and the potential consequences of cyber warfare. On December 23, 2015, Ukraine experienced a historic cyber assault that left hundreds of thousands of people without electricity in the midst of winter. This unprecedented event marked one of the first instances of a cyber attack causing significant physical damage to a nation's infrastructure.

The attack targeted three regional power distribution companies, utilizing sophisticated malware to infiltrate their systems and remotely manipulate critical infrastructure controls. This led to widespread power outages across several regions of Ukraine, affecting residential areas, businesses, and essential services.

Attribution of the attack pointed towards Russian state-sponsored hackers, although Russia denied involvement. The incident highlighted the potential for cyber attacks to disrupt essential services, sow chaos, and create geopolitical tensions.

This case study explores the timeline, tactics, and implications of the cyber attack on the Ukrainian power grid, shedding light on the evolving landscape of cyber warfare and the imperative for nations to bolster their cybersecurity defenses.

Scope

Background and Context: Provide an overview of the Ukrainian power grid, its significance, and the geopolitical context surrounding the cyber attack. Discuss the importance of critical infrastructure protection and the increasing threat of cyber attacks on such systems.

Attack Overview: Detail the timeline of the cyber attack on the Ukrainian power grid, including the initial infiltration, propagation of malware, and the resulting power outages. Discuss the affected regions and the duration of the disruption.

Attack Tactics and Techniques: Analyze the methods used by the attackers to compromise the power grid infrastructure, such as the use of malware, phishing, or other forms of cyber intrusion. Discuss the sophistication of the attack and any unique characteristics.

Attribution and Motivation: Explore the attribution of the attack, including the suspected perpetrators and their potential motives. Discuss any geopolitical tensions or conflicts that may have contributed to or resulted from the attack.

Impact and Consequences: Assess the immediate and long-term impact of the cyber attack on the Ukrainian power grid, including economic, social, and political consequences. Discuss the challenges faced by affected individuals, businesses, and government agencies in responding to the attack.

Response and Recovery Efforts: Evaluate the response efforts by Ukrainian authorities, energy companies, and international partners in mitigating the attack and restoring power. Discuss the effectiveness of the response strategies and any lessons learned.

Lessons Learned and Recommendations: Identify key lessons learned from the cyber attack and provide recommendations for improving cybersecurity measures, enhancing resilience, and mitigating the risk of future attacks on critical infrastructure.

Global Implications and Future Outlook: Discuss the broader implications of the attack on the global cybersecurity landscape, including its influence on international norms, policies, and strategies for defending against cyber threats to critical infrastructure

Objective

The objective of this case study on the cyber attack on the Ukrainian power grid is to:

Understand the Attack: Analyze the tactics, techniques, and procedures (TTPs) used by the attackers to compromise the Ukrainian power grid, including the methods of infiltration, malware deployment, and control manipulation.

Assess Impact: Evaluate the immediate and long-term impact of the cyber attack on the Ukrainian power grid, including economic, social, and political consequences. This includes the effects on infrastructure, services, and public confidence.

Examine Response Efforts: Investigate the response and recovery efforts by Ukrainian authorities, energy companies, and international partners to mitigate the attack and restore power. Assess the effectiveness of these efforts and identify any shortcomings.

Explore Attribution: Examine the attribution of the attack, including the suspected perpetrators and their motives. Assess the evidence supporting attribution and the geopolitical implications of the attack.

Identify Lessons Learned: Identify key lessons learned from the cyber attack, including vulnerabilities in critical infrastructure systems, gaps in cybersecurity defenses, and challenges in response and recovery.

Provide Recommendations: Offer recommendations for improving cybersecurity measures, enhancing resilience, and mitigating the risk of future attacks on critical infrastructure. These recommendations should address

technical, policy, and operational aspects.

Examine Global Implications: Discuss the broader implications of the attack on the global cybersecurity landscape, including its influence on international norms, policies, and strategies for defending against cyber threats to critical infrastructure.

By achieving these objectives, this case study aims to deepen understanding of the cyber attack on the Ukrainian power grid and provide valuable insights for cybersecurity practitioners, policymakers, and stakeholders involved in protecting critical infrastructure from cyber threats.

About the tool and the application chosen

Tool: YARA

YARA is an open-source tool specifically designed for identifying and classifying malware. It enables users to create custom rules that describe patterns or characteristics found in malware samples. These rules can then be used to scan files, processes, memory, or network traffic to detect the presence of malware.

Key Features:

Rule-Based Detection: YARA operates on a rule-based system where users define rules to identify malware based on specific patterns, characteristics, or behaviors. These rules can range from simple string matches to more complex expressions and logic.

Pattern Matching: YARA allows users to define patterns using strings, byte sequences, or regular expressions. This flexibility enables the creation of rules that can accurately detect a wide range of malware variants.

Customizable Rulesets: Users can create and customize rulesets tailored to their specific needs or the malware they are targeting. Rules can be organized into groups and combined to create comprehensive detection strategies.

Flexible Syntax: YARA's syntax is designed to be intuitive and flexible, allowing for the creation of complex rules with ease. It supports logical operators, wildcards, meta information, and more.

Scalability: YARA is highly scalable and efficient, making it suitable for

analyzing large datasets of files or network traffic. It can handle thousands of rules and process millions of samples quickly.

Integration: YARA can be integrated into existing security workflows and tools. It can be used in conjunction with intrusion detection systems (IDS), antivirus solutions, security information and event management (SIEM) systems, and sandbox environments.

Community Support: YARA benefits from an active user community that contributes to the development of rulesets and provides support through forums, online resources, and collaborative projects.

Application Chosen: Malware Detection

YARA's primary application is in malware detection. By creating custom rulesets, security analysts can effectively scan files, processes, memory, or network traffic to identify known malware or suspicious patterns indicative of malicious activity.

How YARA is Applied in Malware Detection:

Rule Creation: Security analysts create YARA rules that describe specific characteristics or behaviors of malware they want to detect. These rules can be based on unique strings, byte sequences, or behavioral patterns associated with known malware families.

YARA rule:

```
rule Emotet_Malware {
    meta:
        description = "Detects Emotet malware"
        author = "Security Analyst"
    strings:
        $string1 = "Emotet" wide ascii
        $string2 = "powershell" nocase
    condition:
        $string1 and $string2
}
```

Rule Optimization: Analysts may optimize rules to reduce false positives and improve detection accuracy. This may involve refining rule patterns, adding additional context, or adjusting matching thresholds.

Rule Deployment: The created YARA rulesets are deployed within security systems, such as endpoint protection platforms, network intrusion detection systems (NIDS), or email gateways. YARA can also be integrated into automated malware analysis workflows or incident response playbooks.

Scanning: YARA scans files, memory, network traffic, or other data sources using the deployed rulesets. It matches the patterns defined in the rules against the scanned data to identify potential malware infections or suspicious behavior.

Alerting and Response: When YARA detects a match with a defined rule, it generates an alert or triggers a response action according to the configured security policies. This may include quarantining the affected file, blocking network traffic, or notifying security personnel for further investigation.

Advantages of Using YARA for Malware Detection:

Flexibility: YARA's flexible syntax allows for the creation of highly specific and tailored rules to detect a wide range of malware variants.

Customization: Security analysts can create and customize rulesets to meet the specific needs and threat landscape of their organization.

Scalability: YARA can efficiently process large volumes of data, making it suitable for analyzing extensive datasets and high-traffic environments.

Community Support: YARA benefits from an active user community that contributes to the development of shared rulesets and provides support for rule creation and optimization.

Real-Time Detection: YARA can provide real-time detection of malware, enabling organizations to respond quickly to emerging threats and security incidents.

By applying YARA in malware detection, organizations can strengthen their cybersecurity posture, identify and mitigate threats, and protect their systems and data from malicious activity.

Tool installation procedure

Getting started

YARA is a multi-platform program running on Windows, Linux and Mac OS X.

Compiling and installing YARA

Download the source tarball and get prepared for compiling it:

```
tar -zxf yara-4.4.0.tar.gz
cd yara-4.4.0
./bootstrap.sh
```

Make sure you have `automake`, `libtool`, `make` and `gcc` and `pkg-config` installed in your system. Ubuntu and Debian users can use:

```
sudo apt-get install automake libtool make gcc pkg-config
```

If you plan to modify YARA's source code you may also need `flex` and `bison` for generating lexers and parsers:

```
sudo apt-get install flex bison
```

Compile and install YARA in the standard way:

```
./bootstrap.sh
./configure
make
sudo make install
```

Run the test cases to make sure that everything is fine:

```
make check
```

Some of YARA's features depend on the OpenSSL library. Those features are enabled only if you have the OpenSSL library installed in your system. If not, YARA is going to work fine but you won't be able to use the disabled features.
The `configure` script will automatically detect if OpenSSL is installed or not. If you want to enforce the OpenSSL-dependent features you must pass `--with-crypto` to the `configure` script. Ubuntu and Debian users can use `sudo apt-get install libssl-dev` to install the OpenSSL library.

The following modules are not compiled into YARA by default:

- cuckoo
- magic

- dotnet

If you plan to use them you must pass the corresponding `--enable-<module name>` arguments to the `configure` script.

For example:

```
./configure --enable-cuckoo
./configure --enable-magic
./configure --enable-dotnet
./configure --enable-cuckoo --enable-magic --enable-dotnet
```

Modules usually depend on external libraries, depending on the modules you choose to install you'll need the following libraries:

- **cuckoo:**

  Depends on Jansson for parsing JSON. Some Ubuntu and Debian versions already include a package named `libjansson-dev`, if `sudo apt-get install libjansson-dev` doesn't work for you then get the source code from its repository.

- **magic:**

  Depends on *libmagic*, a library used by the Unix standard program file. Ubuntu, Debian and CentOS include a package `libmagic-dev`. The source code can be found here.

Installing with vcpkg

You can also download and install YARA using the vcpkg dependency manager:

```
git clone https://github.com/microsoft/vcpkg.git
cd vcpkg
./bootstrap-vcpkg.sh
./vcpkg integrate install
vcpkg install yara
```

The YARA port in vcpkg is kept up to date by Microsoft team members and community contributors. If the version is out of date, please create an issue or pull request on the vcpkg repository.

Installing on Windows

Compiled binaries for Windows in both 32 and 64 bit flavors can be found in the link below. Just download the version you want, unzip the archive, and put the `yara.exe` and `yarac.exe` binaries anywhere in your disk.

Download Windows binaries

To install YARA using Scoop or Chocolatey, simply type `scoop install yara` or `choco install yara`. The integration with both *Scoop* and *Chocolatey* are not maintained their respective teams, not by the YARA authors.

Installing on Mac OS X with Homebrew

To install YARA using Homebrew, simply type `brew install yara`.

Installing `yara-python`

If you plan to use YARA from your Python scripts you need to install the `yara-python` extension. Please refer to https://github.com/VirusTotal/yara-python for instructions on how to install it.

Running YARA for the first time

Now that you have installed YARA you can write a very simple rule and use the command-line tool to scan some file:

```
echo "rule dummy { condition: true }" > my_first_rule
yara my_first_rule my_first_rule
```

Don't get confused by the repeated `my_first_rule` in the arguments to `yara`, I'm just passing the same file as both the rules and the file to be scanned. You can pass any file you want to be scanned (second argument).

If everything goes fine you should get the following output:

```
dummy my_first_rule
```

Which means that the file `my_first_rule` is matching the rule named `dummy`.

If you get an error like this:

```
yara: error while loading shared libraries: libyara.so.2: cannot open shared
object file: No such file or directory
```

It means that the loader is not finding the `libyara` library which is located in `/usr/local/lib`. In some Linux flavors the loader doesn't look for libraries in this path by default, we must instruct it to do so by adding `/usr/local/lib` to the loader configuration file `/etc/ld.so.conf`:

```
sudo sh -c 'echo "/usr/local/lib" >> /etc/ld.so.conf'
sudo ldconfig
```

On newer Ubuntu releases such as 22.04 LTS, the correct loader configuration is installed via dependencies to `/etc/ld.so.conf.d/libc.conf`. In this case, the following command alone is sufficient to configure the dynamic linker run-time bindings.

sudo ldconfig

If you're using Windows PowerShell as your command shell, `yara my_first_rule my_first_rule` may return this error:

```
my_first_rule(1): error: non-ascii character
```

You can avoid this by using the `Set-Content` cmdlet to specify ascii output when creating your rule file:

```
Set-Content -path .\my_first_rule -Value "rule dummy { condition: true }" -Encoding Ascii
.\yara my_first_rule my_first_rule
```

Steps of ethical hacking that you have done on your application using the chosen tool

YARA for malware detection on an application:

1. Reconnaissance:

- Gather information about the application to understand its architecture, technology stack, and potential vulnerabilities.

2. Scanning:

- Use YARA to scan the application's files, binaries, or memory for known malware signatures or suspicious patterns.
- Create YARA rules that target specific malware families or behaviors relevant to the application's environment.

3. Enumeration:

- Enumerate the application's endpoints, APIs, or services to identify potential attack vectors.
- Ensure that YARA rules cover areas where malware could potentially be introduced or exploited.

4. Vulnerability Analysis:

- Analyze the results of YARA scans to identify any vulnerabilities or weaknesses in the application's security defenses.
- Investigate any matches found by YARA to determine the

severity and impact of potential malware infections.

5. Exploitation:

- If any vulnerabilities are discovered, attempt to exploit them to gain further access or control over the application.
- Use YARA to monitor for any indicators of compromise (IOCs) resulting from the exploitation.

6. Post-Exploitation:

- Conduct post-exploitation activities to gather additional information, escalate privileges, or maintain access to the application.
- Continuously monitor the application with YARA to detect any malicious activity or persistence mechanisms left behind by attackers.

7. Reporting:

- Document the findings of the ethical hacking process, including details of any vulnerabilities, exploits, or malware detections.
- Provide recommendations for mitigating the identified risks and improving the application's security posture.

8. Remediation:

- Work with the application's developers or administrators to address and remediate the vulnerabilities or malware infections

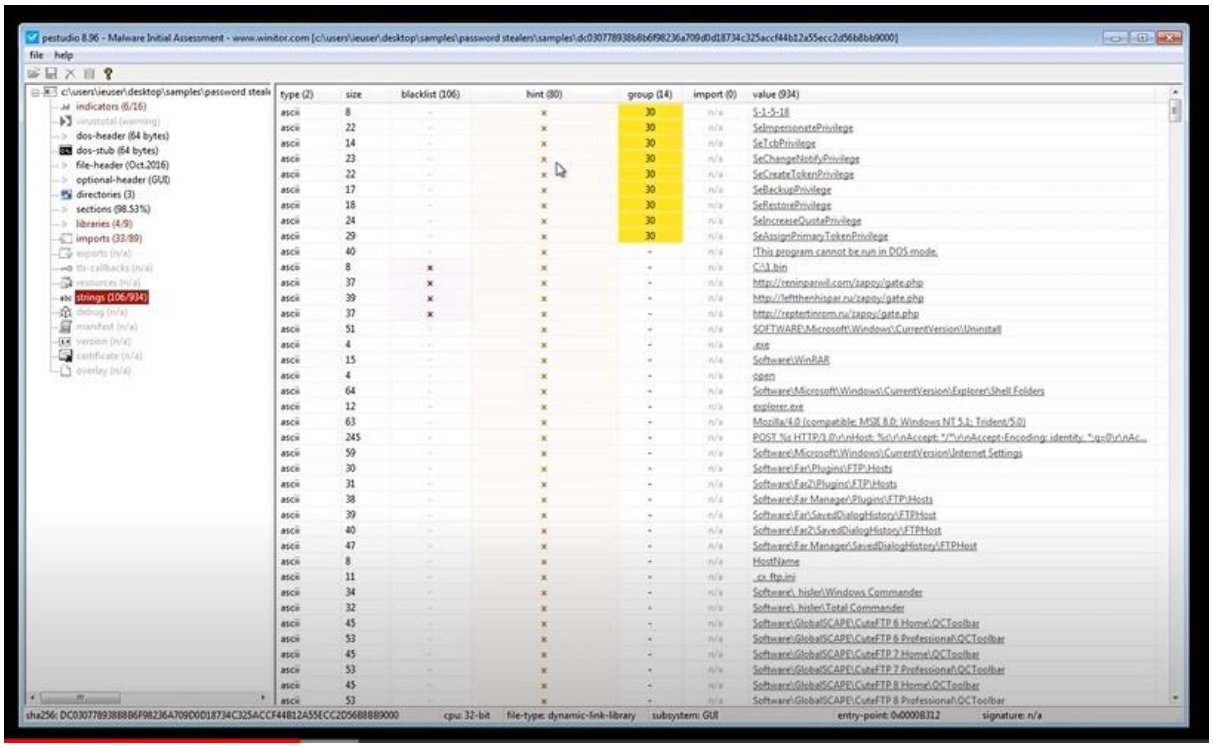
identified during the ethical hacking process.

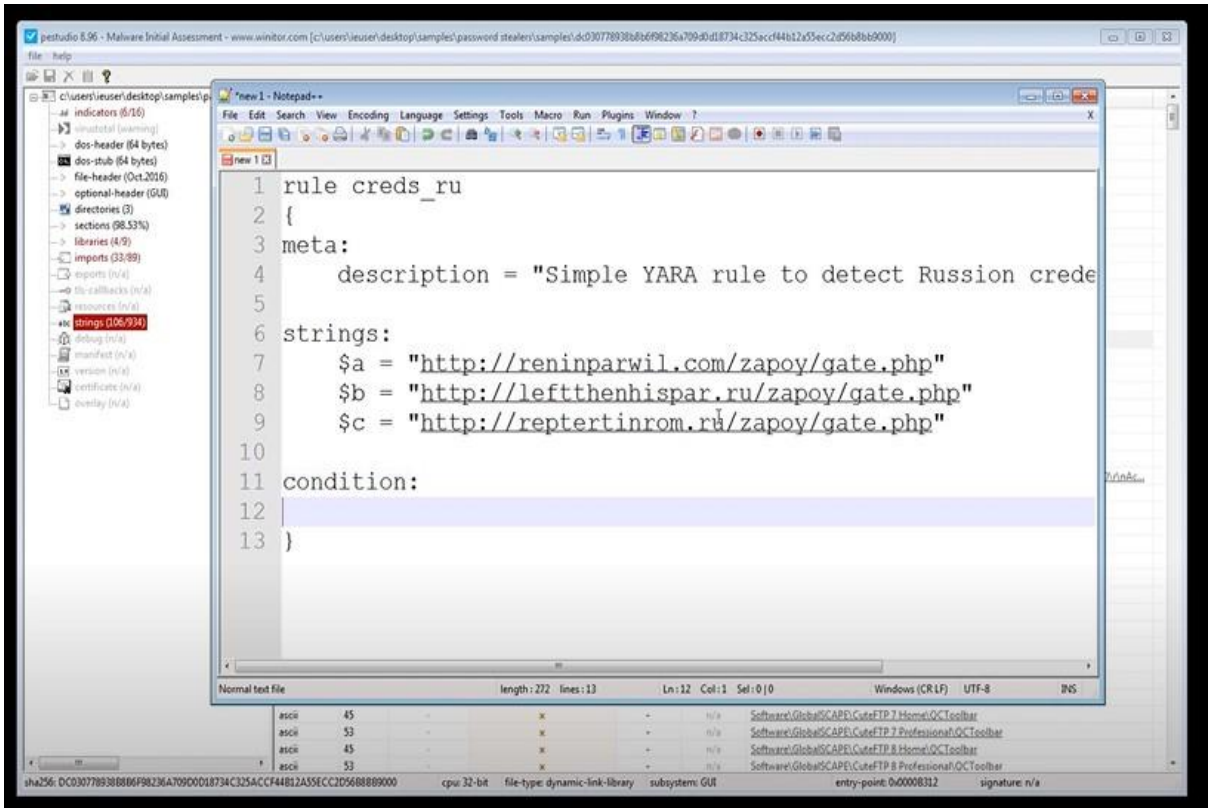
- Develop and deploy countermeasures, such as applying patches, updating security configurations, or implementing additional security controls.

By following these steps of ethical hacking and utilizing YARA for malware detection, organizations can proactively identify and address security issues within their applications, helping to protect against potential cyber threats and malicious attacks.

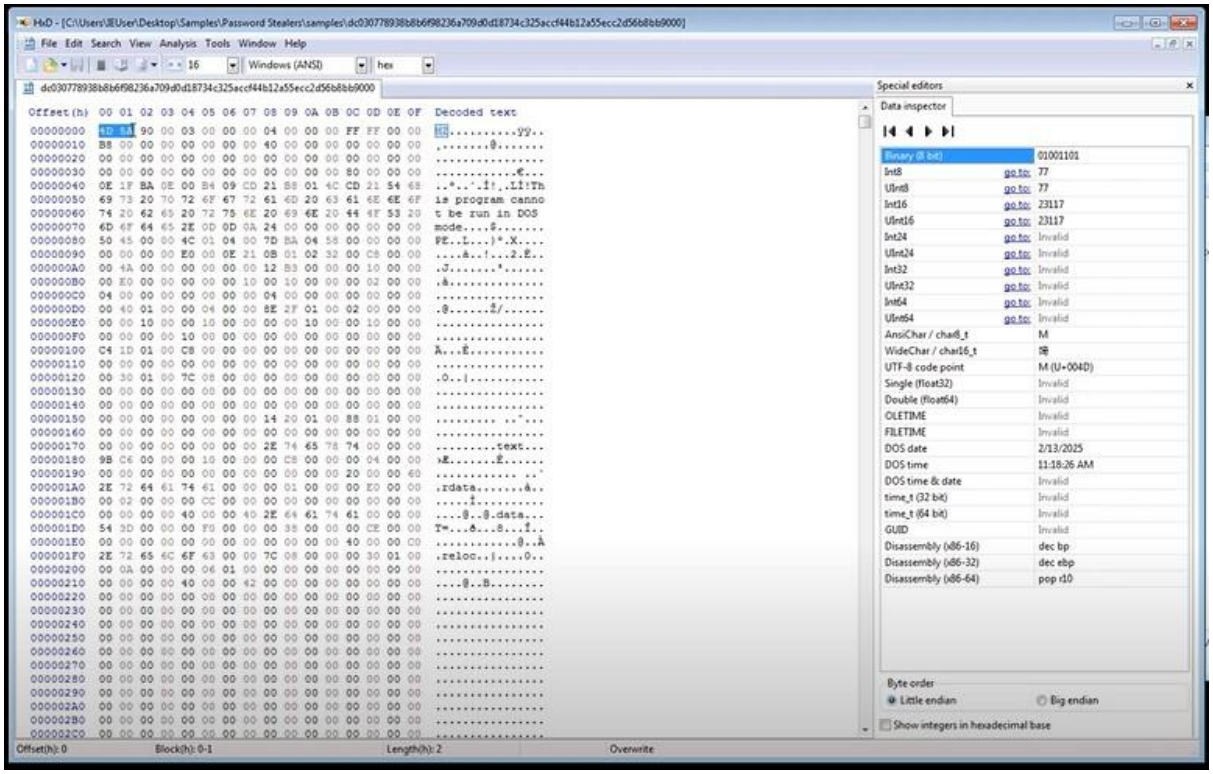
Screenshots of the implementation

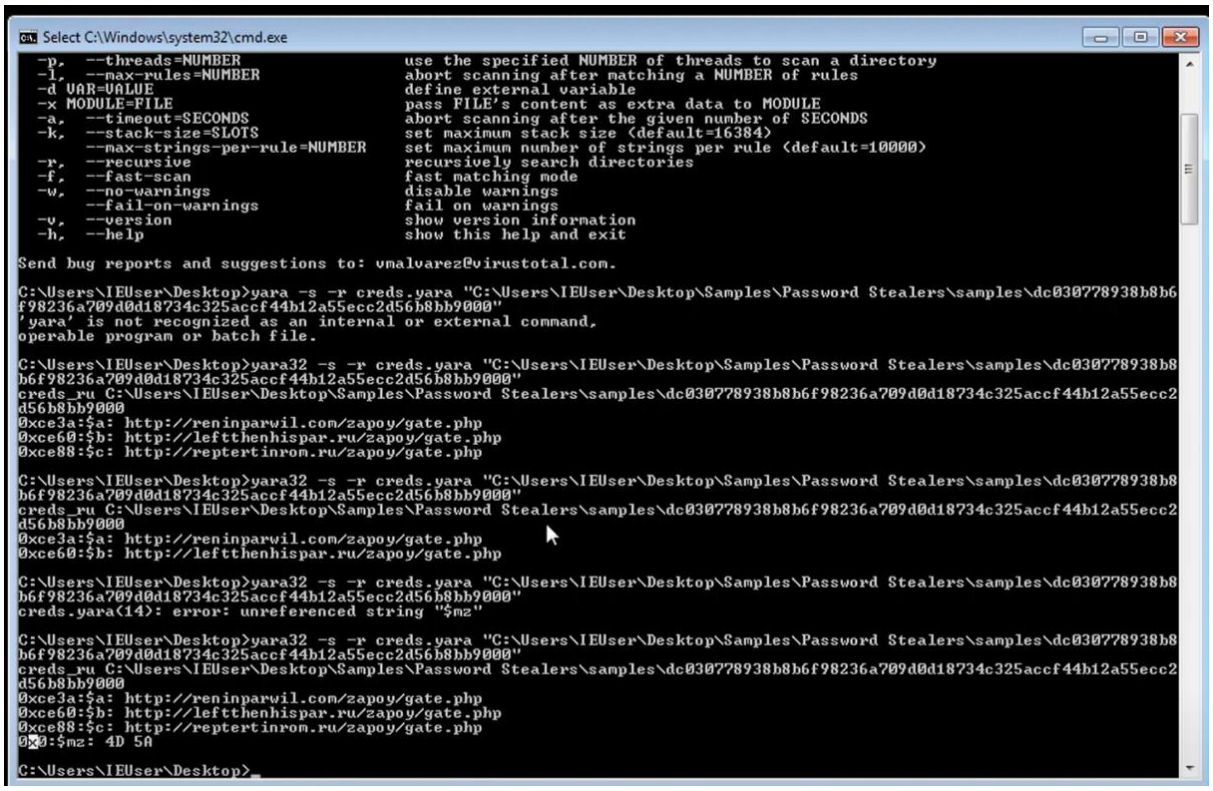
```
rule pdf_1.7_contains_few_links {

meta:
  author = "Sean Whalen"
  last_updated = "2017-06-08"
  tlp = "white"
  category = "malicious"
  confidence = "medium"
  killchain_phase = "exploit"
  description = "A PDFv1.7 that contains one or two links - a common phishing tactic"

strings:
  $pdf_magic = {25 50 44 46}
  $s_anchor_tag = "<a " ascii wide nocase
  $s_uri = /\(http.+\)/ ascii wide nocase

condition:
  $pdf_magic at 0 and (#s_anchor_tag == 1 or (#s_uri > 0 and #s_uri < 3))
}
```






```
C:\Users\IEUser\Desktop>yara32 --help
YARA 3.7.0, the pattern matching swiss army knife.
Usage: yara [OPTION]... [NAMESPACE:]RULES_FILE... FILE | DIR | PID

Mandatory arguments to long options are mandatory for short options too.

  -t, --tag=TAG                      print only rules tagged as TAG
  -i, --identifier=IDENTIFIER        print only rules named IDENTIFIER
  -c, --count                        print only number of matches
  -n, --negate                       print only not satisfied rules (negate)
  -D, --print-module-data            print module data
  -g, --print-tags                   print tags
  -m, --print-meta                   print metadata
  -s, --print-strings                print matching strings
  -L, --print-string-length          print length of matched strings
  -e, --print-namespace              print rules' namespace
  -p, --threads=NUMBER               use the specified NUMBER of threads to scan a directory
  -l, --max-rules=NUMBER             abort scanning after matching a NUMBER of rules
  -d VAR=VALUE                       define external variable
  -x MODULE=FILE                     pass FILE's content as extra data to MODULE
  -a, --timeout=SECONDS              abort scanning after the given number of SECONDS
  -k, --stack-size=SLOTS             set maximun stack size (default=16384)
      --max-strings-per-rule=NUMBER  set maximun number of strings per rule (default=10000)
  -r, --recursive                    recursively search directories
  -f, --fast-scan                    fast matching mode
  -w, --no-warnings                  disable warnings
      --fail-on-warnings             fail on warnings
  -v, --version                      show version information
  -h, --help                         show this help and exit

Send bug reports and suggestions to: vmalvarez@virustotal.com.

C:\Users\IEUser\Desktop>_
```


HxD - [C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000]
File Edit Search View Analysis Tools Window Help
Windows (ANSI)
hex
Special editors
Data inspector
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Decoded text
00000000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ..........ÿÿ..
00000010 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ¸.......@.......
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000030 00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00 ....€...
00000040 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ..º..´.Í!¸.LÍ!Th
00000050 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno
00000060 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t be run in DOS
00000070 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode....$.......
00000080 50 45 00 00 4C 01 04 00 7D BA 04 58 00 00 00 00 PE..L...}º.X....
00000090 00 00 00 00 E0 00 0E 21 0B 01 02 32 00 C8 00 00 ....à..!...2.È..
000000A0 00 4A 00 00 00 00 00 00 12 B3 00 00 00 10 00 00 .J.......³......
000000B0 00 E0 00 00 00 00 00 00 10 00 10 00 00 00 02 00 .à..............
000000C0 04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00 ....
000000D0 00 40 01 00 00 04 00 00 8E 2F 01 00 02 00 00 00 .@......Ž/......
000000E0 00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00 ....
000000F0 00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 ....
00000100 C4 1D 01 00 C8 00 00 00 00 00 00 00 00 00 00 00 Ä...È...........
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000120 00 30 01 00 7C 08 00 00 00 00 00 00 00 00 00 00 .0..|...........
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000150 00 00 00 00 00 00 00 00 14 20 01 00 88 01 00 00 ....ˆ...
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000170 00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00 ....text...
00000180 9B C6 00 00 00 10 00 00 00 C8 00 00 00 04 00 00 ›Æ.......È......
00000190 00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60 ....`
000001A0 2E 72 64 61 74 61 00 00 00 01 00 00 00 E0 00 00 .rdata.......à..
000001B0 00 02 00 00 00 CC 00 00 00 00 00 00 00 00 00 00 .....Ì..........
000001C0 00 00 00 00 40 00 00 40 2E 64 61 74 61 00 00 00 ....@..@.data...
000001D0 54 3D 00 00 00 F0 00 00 00 38 00 00 00 CE 00 00 T=...ð...8...Î..
000001E0 00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 C0 ....@..À
000001F0 2E 72 65 6C 6F 63 00 00 7C 08 00 00 00 30 01 00 .reloc..|....0..
00000200 00 0A 00 00 00 06 01 00 00 00 00 00 00 00 00 00 ....
00000210 00 00 00 00 40 00 00 42 00 00 00 00 00 00 00 00 ....@..B........
00000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
000002A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
000002B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
000002C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
Data inspector
Binary (8 bit) 01001101
Int8 go to: 77
UInt8 go to: 77
Int16 go to: 23117
UInt16 go to: 23117
Int24 go to: Invalid
UInt24 go to: Invalid
Int32 go to: Invalid
UInt32 go to: Invalid
Int64 go to: Invalid
UInt64 go to: Invalid
AnsiChar / char8_t M
WideChar / char16_t
UTF-8 code point M (U+004D)
Single (float32) Invalid
Double (float64) Invalid
OLETIME Invalid
FILETIME Invalid
DOS date 2/13/2025
DOS time 11:18:26 AM
DOS time & date Invalid
time_t (32 bit) Invalid
time_t (64 bit) Invalid
GUID Invalid
Disassembly (x86-16) dec bp
Disassembly (x86-32) dec ebp
Disassembly (x86-64) pop r10
Byte order
Little endian
Big endian
Show integers in hexadecimal base
Offset(h): 0
Block(h): 0-1
Length(h): 2
Overwrite


Select C:\Windows\system32\cmd.exe
-p, --threads=NUMBER use the specified NUMBER of threads to scan a directory
-l, --max-rules=NUMBER abort scanning after matching a NUMBER of rules
-d VAR=VALUE define external variable
-x MODULE=FILE pass FILE's content as extra data to MODULE
-a, --timeout=SECONDS abort scanning after the given number of SECONDS
-k, --stack-size=SLOTS set maximum stack size (default=16384)
--max-strings-per-rule=NUMBER set maximum number of strings per rule (default=10000)
-r, --recursive recursively search directories
-f, --fast-scan fast matching mode
-w, --no-warnings disable warnings
--fail-on-warnings fail on warnings
-v, --version show version information
-h, --help show this help and exit
Send bug reports and suggestions to: vmalvarez@virustotal.com.
C:\Users\IEUser\Desktop>yara -s -r creds.yara "C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000"
'yara' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\IEUser\Desktop>yara32 -s -r creds.yara "C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000"
creds_ru C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000
0xce3a:$a: http://reninparwil.com/zapoy/gate.php
0xce60:$b: http://leftthenhispar.ru/zapoy/gate.php
0xce88:$c: http://reptertinrom.ru/zapoy/gate.php
C:\Users\IEUser\Desktop>yara32 -s -r creds.yara "C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000"
creds_ru C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000
0xce3a:$a: http://reninparwil.com/zapoy/gate.php
0xce60:$b: http://leftthenhispar.ru/zapoy/gate.php
C:\Users\IEUser\Desktop>yara32 -s -r creds.yara "C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000"
creds.yara(14): error: unreferenced string "$mz"
C:\Users\IEUser\Desktop>yara32 -s -r creds.yara "C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000"
creds_ru C:\Users\IEUser\Desktop\Samples\Password Stealers\samples\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000
0xce3a:$a: http://reninparwil.com/zapoy/gate.php
0xce60:$b: http://leftthenhispar.ru/zapoy/gate.php
0xce88:$c: http://reptertinrom.ru/zapoy/gate.php
0x0:$mz: 4D 5A
C:\Users\IEUser\Desktop>

Conclusion

The cyber attack on the Ukrainian power grid, which occurred on December 23, 2015, was a watershed moment in the realm of cyber warfare, marking one of the first instances of a cyber attack causing significant physical damage to a nation's infrastructure. By targeting three regional power distribution companies, the attackers utilized sophisticated malware to infiltrate systems and manipulate critical infrastructure controls, resulting in widespread power outages affecting hundreds of thousands of people. Attribution of the attack pointed towards Russian state-sponsored hackers, although Russia denied involvement. The incident underscored the potential for cyber attacks to disrupt essential services, sow chaos, and escalate geopolitical tensions. In response to the attack, Ukrainian authorities, energy companies, and international partners worked to restore power and investigate the incident.

The implementation of YARA for malware detection offers a proactive approach to defending against similar cyber attacks in critical infrastructure. YARA's rule-based system allows for the creation of custom rulesets tailored to detect specific malware signatures or behaviors. By scanning network traffic, files, and systems, organizations can identify indicators of compromise and potential threats, enabling a swift response to mitigate the impact of cyber attacks. In conclusion, the cyber attack on the Ukrainian power grid underscores the importance of bolstering cybersecurity defenses in critical infrastructure systems. Through the implementation of tools like YARA, organizations can enhance their ability to detect and respond to cyber threats effectively.

References

https://medium.com/@valerie7995/yara-tryhackme-c2d59c33b4a6

https://www.varonis.com/blog/yara-rules

https://www.magnetforensics.com/blog/yara-rule-processing-in-magnet-axiom-cyber/