International Journal of Advanced Research and Review

www.ijarr.org

https://doi.org/10.70914/ijarr.2026.v11.i03.pp11-20

Consumption of technical threat intel indicators

[1] Mr.V.Anil Kumar,[2] Bathali Bhoomika,[3] M.Srija,[4] Pesari Akshara,[5] Guniti Anil,

[1]Assistant Professor, Department of Data Science, Narsimha Reddy Engineering Collage, Maisammaguda(V), Kompally, Telangana.

[2,3,4,5] Student, Department of Data Science, Narsimha Reddy Engineering Collage, Maisammaguda(V), Kompally, Telangana.

Abstract —

More and more, our daily lives rely on the enabling function of IT systems. Consistent with this, the increasing frequency and severity of cyberattacks is a major concern. Both businesses and governments must now take measures to protect themselves against cybercriminals. It is becoming more and more difficult to determine the right reaction to cyber-attacks, despite the fact that proactive defense and quick response are vital for IT security experts. There is a need to identify new forms of attack and to have real-time information that provides details of attackers' Tactics, Techniques, and Procedures and identification of Indicators of Compromise. Helping to fortify defenses with information on new attack channels for cybercriminals, Cyber Threat Intelligence is playing an increasingly important role in assisting with threat detection, analysis, and the creation of suitable responses. The publication focuses on the use of technical threat intelligence indicators. Cybersecurity, Industrial Internet of Things (IoT), MITRE ATT&CK, and Cyber Threat Intelligence

I. INTRODUCTION

Individuals and businesses would struggle to meet the high standards of efficiency and progress set for them if they did not have access to reliable information technology. But there are a lot of dangers that come along with being too reliant on computerized information systems. Cyberspace is seeing an uptick in assaults, and the harm they're wreaking is growing rapidly. As a result, being proactive and responding quickly are now crucial. Experts in information technology security are facing a growing number of challenges, one of the most important being how to respond appropriately to cyberattacks. New types of attacks must be discovered, and real-time data revealing the tactics, techniques, and procedures (TTPs) of the attackers, as well as indicators of compromise (IoCs), must be collected.

By giving information on newly emerging attack vectors, Cyber Threat Intelligence (CTI) helps organizations recognize risks, analyze them, and build suitable solutions. This, in turn, strengthens their defenses. Using technical threat intelligence indications is the subject of this paper.

II. NETWORK INDICATORS

Network security is an important part of cybersecurity's ability to identify and prevent attacks. As a result, this battlefield necessitates the completion of many missions. When it comes to intrusion detection and prevention systems (IDPS), there are usually two primary types of engines in a network security ecosystem. One

kind monitors internal mirrored traffic, while the other is located at the perimeter and faces the internet. Consequently, technical indications are crucial. Indicators of CTI, particularly those fed into the proper places, are the usual focus of this article. Engines from IDPS (Intrusion Detection and Prevention Systems) often consume network indications by comparing datasets of network traffic with the contents of CTI databases and data models. One of the most popular data formats for the organized description and exchange of cybersecurity information is STIX, which stands for Structured Threat Information Expression. The US Department of Defense first released this format; the OASIS nonprofit subsequently took over its management, along with that of the TAXII and CybOX technologies. Standards for the organized administration of cybersecurity data and the incorporation of independent formats that are considered appropriate are overseen by the Cyber Threat Intelligence Technical Committee (CTI TC) inside OASIS. Participating in this forum, prominent CTI system developers from across the globe help shape the future of the STIX format in response to user feedback and industry trends. Many Network Traffic Objects are necessary for STIX to function. The IP addresses, domains, signatures, vulnerability information, and other particular workarounds are examined in this article.

A. IP

During incident management, blocking rogue IP addresses is often a simple answer. It is recommended to do this in the event that an alert is generated or if it is determined during the investigation that such an IP is connecting with an internal asset. They usually block them at the perimeter firewall, which incident response teams may do in one of three ways: by entering the IP address directly into the device, by reading from a list, or by using the API (if incident handlers only have access to particular functionalities). Although there may be millions of IP addresses used for host sweep scanning, the average number of entries that may be posted to a blocklist is between 200,000 and 300,000. This is okay, however, since malicious IP addresses are not constants. Importantly, the right context is determined by looking at the IP address's activity history and the retention policy, which is usually 30 days. This is when the need to make use of the CTI database, which houses this historical data, whether derived via crowdsourcing or study, comes into play.[4] The TOR exit nodes are an absolute ban list feed component of the CTI feed. Though this is also an evolving list, their estimated population is close to 1,500. Since the likelihood of genuine people connecting from these is low, it is advised to ban them without inquiry. Because they act as entry points to a site on the dark web, these exit nodes are usually dangerous.[5] Investigative applications of IP addresses may include proactive rather than reactive measures, such as threat hunting. Working with source and destination IP addresses created inside the protected network and a wider threat intelligence database of IP addresses is a common need for retrospective analysis with big volumes of data. Since the CTI database is just required to serve as a lookup table in this scenario, its size might be rather substantial. Checking for communications with known malicious IP addresses is part of the procedure. The next step is to confirm or disprove these theories based on these results. On the other hand, the IP address CTI data model faces additional problems as a result of the aforementioned method. As an example, this may involve: • virtual private network (VPN) access points; The IP address is a proxy, which includes transparent proxies, HTTP/HTTPS, SSL, SOCKS, and CONNECT protocols; All cloud providers and IP addresses that are associated with datacenters are considered to be part of the same category. Can help identify bot or artificial traffic; For anonymity, it's either is_tor or is_proxy. IP addresses are known to be the source of harmful behavior, including attacks, malware, botnet activity, and more. Known abusers include IP addresses that are known to send spam, gather data, activate bots, and other unwanted bots; [6] When an IP address is bogus, the value is true. are control-list-limited (ACL) on routers or blackholed by BGP; IP reputation ratings plotted on a map; • Geo: Location based on IP address. It matters if an IP usually fulfills a dropper or C&C role, however there is a lack of staged threat intelligence and this is the main concern. Common ones, like STIX, aren't even up to the task. This demonstrates how data models are playing an ever-growing role in the CTI realm.

B. Domain

Since domain names, unlike IP addresses, are immutable, there's no need to establish and enforce retention restrictions in order to prohibit them. Additionally, they may be blocked at the perimeter by the security device or via sinkholing on the DNS server. Comparable to routing to a null route, a DNS sinkhole behaves as a dead end or black hole. If we want to stop traffic from going to malwarexyzabc.com, a malicious website that is known to be hosted at 1.1.1.1, we can tell our DNS server to reroute all requests for that domain to 127.0.0.1, which is the loopback address, so blocking the traffic altogether. It may be routed to a logging server's IP address, which would allow for the tracking and investigation of computers trying to visit this malicious domain. As a key component of cybersecurity incident response, domains are a potential vector for several forms of attack, including phishing and malware dissemination [7.]. Domain Generating Algorithms (DGAs) are often used by malware to obfuscate kill switches. With a single piece of virus, DGAs may produce thousands of domains. In order to evade detection or disrupt their malware's transmission, developers often use this technique. Malware infestations would be brief if IP addresses or domains were hardcoded for contacting home and receiving orders. By proactively identifying the

hardcoded domains, security workers might prevent infected machines in a victim's architecture from sending outbound communication by feeding them into a network blacklisting appliance. This is why DGAs circumvent it by creating hundreds—if not thousands—of new domains every day, with just a fraction of those domains being utilized to communicate with the C&C server. Mr. DGThe domains created by the virus and the ones created by the command and control server are almost always the same since both are symmetric. When creating these domains, the algorithms make use of a seed of some kind. As an example, domain generation in older CryptoLocker malware variants relied on the current date. Here is a Python version of the DGA that was utilized by those early CryptoLocker versions. Using a date as input, the method does several bitwise operations on the date's integer components before transforming them into the domain name's letters.[8] The text files may grow to several hundred gigabytes in size for each virus family, thus it's not recommended to utilize this as a threat feed. This has the potential to cause counterproductive productivity by exhausting computing resources. Activating local detection capabilities is the answer for detection. The detection technique checks the protected network's DNS records against a local lookup table. The measure of the degree of unpredictability of variables, Cloud Shannon entropy, may be used for this purpose. Word randomness cannot be shown by it, although character randomness may be detectable. We suggest detecting capabilities based on neural networks for this study, while there are many more techniques. Overall, the solutions need to be able to detect suspicious behavior by a domain that classification systems generally miss by detecting the telemetry indicated before. Domains used for malware distribution may be effectively removed using the aforementioned techniques; however, phishing assaults cannot. To do this, I suggest using a local algorithm that compares message trace logs with a lookup table. Typosquatting detection might be a great fit for this method. The method for detecting typosquatting involves checking a protected domain list against each incoming email sender's domain name. Included in this protected list are all the domain names—including those of the corporation and its affiliated businesses—that we want to safeguard against typosquatting attempts. Within the comparison, the Gestalt pattern matching result and the weighted average of the normalized Jaro-Winkler edit distance are used. The detection provides both the original log and the domain that was hit by the typosquatting assault if this value is greater than a preset threshold. This solution's defining characteristics are: Important traits: Extensive library of powerful domain fuzzing algorithms; • Domain names that use Unicode (IDN); • Extra domain permutations retrieved from dictionaries; • Efficient allocation of tasks among several threads; Real-time identification of phishing websites; Fuzzy hashes (ssdeep/tlsh) for comparing HTML documents; • Visual resemblance with perceptual hashes (pHash) in screenshots; • Identifying malicious MX hosts (which intercept redirected emails); 2. Geographic IP address [9.]

C. Signatures

Network signatures' primary function is to detect suspicious or harmful activity in network traffic. Both internal and external traffic may be used to accomplish this. Typically, signatures are implemented in Intrusion Prevention System (IPS) mode on perimeter security devices. This mode requires a more careful selection of the ruleset to ensure that legitimate operations are not hindered. As an additional layer of network security, monitoring internal traffic is another option that is now practically required. We distinguish between traffic going east and west, and traffic going north and south. For the most part, traffic going north to south is what you'd expect from a firewall or switch, but traffic going east to west is more common within something like a virtualization cluster. Since methods like lateral movement could not be seen in East-West traffic, both sets of data are essential. Among IDPS engines, SNORT and SURICATA are the most well-known. In most cases, SURICATA is responsible for internal traffic IDS while SNORT is in charge of perimeter IPS, but this is not always the case. Both use Deep Packet Inspection and are open-source technologies for analyzing network traffic for harmful material. An benefit of SURICATA is that it is naturally multi-threaded, which allows for much faster performance and can be accelerated using CUDA. Its capacity to conditionally store network traffic as PCAP files is another key benefit.
The abundance of JSON (EVE) in SURICATA's output makes it ideal for use in NSM (Network Security Monitoring) tasks. Here, the signature is the source of threat intelligence; the most common sources are the Emerging Threats (ET) database and the Snort rules database. To facilitate the change, Suricata may use most Snort rules with minor tweaks. It has its own set of rules that it continuously updates. Having the option to construct bespoke alarms is vital in detecting engineering procedures, however. The best course of action, for instance, is to draft a payload detection rule in the event that the exploit is detected in the network, especially in cases when patches are not available for older system vulnerabilities. in number ten. Zeek is a great addition to NSM capabilities as it provides data sources for a lot of IoCs. Zeek creates a large amount of logs that detail network activity. These logs include application-layer transcripts and a complete record of all connections seen on the wire. the eleventh.

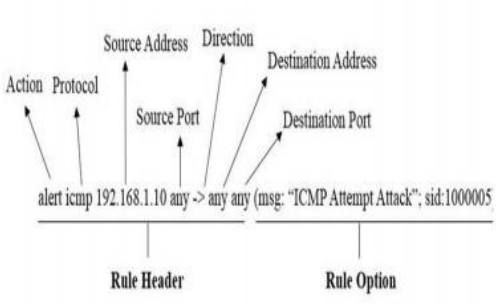

Fig. 1. Example of Snort IDS Rule. Source: N Khamphakdee, N Benjamas, S Saiyod (2015)

As a result of Zeek's ability to parse network data into files.log, we may implement YARA rules—usually applied to client-side security—on the network itself. The YARA framework, which stands for "Yet Another Ridiculous Acronym," is based on rules and is used for large-scale pattern matching. The goal of these YARA rules is to detect and categorize malware samples by using textual or binary patterns to build families of malware. YARA regulations are available in a wide variety of databases, both public and private. Following these guidelines, which are based on the Detection as Code philosophy, is highly recommended. Look at this example to see how YARA rules may be used as a CTI source:

```
rule IDDQD_God_Mode_Rule {
  meta:
    description = "Detects a wide array of cyber threats, from malware and ransomware to advanced persistent threats (APTs)"
    author = "Florian Roth"
    reference = "Internal Research - get a god mode rule set with THOR by Nextron Systems"
    date = "2019-05-15"
    modified = "2024-01-12"
    score = 60
  strings:
    $ = "sekurlsa::logonpasswords" ascii wide nocase /* Mimikatz Command */
    $ = "ERROR kuhl" wide xor /* Mimikatz Error */
```

```
    $ = " -w hidden " ascii wide nocase                    /*
Power Shell Params */

    $ = "Koadic." ascii                                    /* Koadic
Framework */

    $ = "ReflectiveLoader" fullword ascii wide xor         /*
Generic - Common Export Name */
```

```
    $ = "%s as %s\\%s: %d" ascii xor                       /*
CobaltStrike indicator */

    $    =    "[System.Convert]::FromBase64String("    ascii
/* PowerShell - Base64 encoded payload */

    $ = "/meterpreter/" ascii xor                          /* Metasploit
Framework - Meterpreter */

    $              =              /              -[eE][decoman]{0,41}
['"]?(JAB|SUVYI|aWV4I|SQBFAFgA|aQBlAHgA|cgBlAG)/
ascii wide  /* PowerShell encoded code */

    $ = /  (sEt|SEt|SeT|sET|seT)  / ascii wide             /*
Casing Obfuscation */

    $ = ");iex " nocase ascii wide                         /*
PowerShell - compact code */

    $ = "Nir Sofer" fullword wide                          /* Hack
Tool Producer */

    $ = "impacket." ascii                                  /* Impacket
Library */

    $  =  /\[[\+\-!E]\]  (exploit|target|vulnerab|shell|inject)/
nocase  /* Hack Tool Output Pattern */
```

Florian Roth's preceding example exemplifies a universal rule with full detection capacity. The 'God Mode Rule' is an experimental YARA rule that aims to detect various security concerns. It has a number of malware indications, including Mimikatz, Metasploit Meterpreter, PowerShell obfuscation, encoded payloads, and other hacking tools. Shadow copy deletion orders and crypto mining routines are among the ransomware behaviors that this rule aims to prevent. It's much better now at spotting APTs, obfuscation methods, and distinctive strings from popular hacking frameworks and tools. [12.]

```
    $ = "0000FEEDACDC}" ascii wide                    /*
Squiblydoo - Class ID */

    $ = "vssadmin delete shadows" ascii nocase        /*
Shadow Copy Deletion via vssadmin - often used in
ransomware */

    $ = ".exe delete shadows" ascii nocase            /*
Shadow Copy Deletion via vssadmin - often used in
ransomware */

    $ = " shadowcopy delete" ascii wide nocase        /*
Shadow Copy Deletion via WMIC - often used in ransomware
*/

    $ = " delete catalog -quiet" ascii wide nocase    /*
Shadow Copy Deletion via wbadmin - often used in
ransomware */

    $ = "stratum+tcp://" ascii wide                   /* Stratum
Address - used in Crypto Miners */

    $                                                 =
/\(Debug|Release)\\(Key[lL]og|[Ii]nject|Steal|By[Pp]ass|A
msi|Dropper|Loader|CVE\-)/  /* Typical PDB strings found
in malware or hack tools */

    $  =  /(Dropper|Bypass|Injection|Potato)\.pdb/  nocase
/* Typical PDP strings found in hack tools */

    $ = "Mozilla/5.0" xor(0x01-0xff) ascii wide       /*
XORed Mozilla user agent - often found in implants */
```

D. Vulnerabilities

The CTI component's vulnerability information may be either repository-based or human-readable, such in a disclosure or white paper. There is an emphasis in this study on repository-based vulnerability information. When carried out properly, vulnerability detection is an essential component of cybersecurity operations as it limits the attacker's capacity to move about. Because of this, a proper vulnerability management program is crucial. This encompasses the right equipment, competent personnel, and protocols. Cyber Threat Intelligence (CTI) offers a wealth of options for valuable vulnerability enrichment.
With the encrypted data in hand, the internally specified IDS engines may be able to compile a comprehensive dynamic inventory of the host system and all of its installed software, down to the version numbers. We suggest building a lookup table and comparing it to a database that contains vulnerabilities using this inventory as the main dataset. Its ability to use an API to access the NIST National Vulnerability Database is an ideal answer to this problem. The NVD has a substantial backlog of paper submissions (~100 days), which should be taken into consideration. When active scanning is not an option, as is often the case with control networks, this workaround might be useful.

```
    $ = "amsi.dllATVSH" ascii xor                    /* Havoc
C2 */

    $ = "BeaconJitter" xor                           /* Sliver */

    $ = "main.Merlin" ascii fullword                 /* Merlin
C2 */

    $                                                =
"\x48\x83\xec\x50\x4d\x63\x68\x3c\x48\x89\x4d\x10" xor /*
Brute Ratel C4 */

    $ = "}{0}\"-f" ascii wide                        /* PowerShell
obfuscation - format string */

    $ = "HISTORY=/dev/null" ascii                    /* Linux
HISTORY tampering - found in many samples */

    $ = " /tmp/x;" ascii                             /* Often used
in malicious linux scripts */

    $ = /comsvcs(\.dll)?[, ]{1,2}(MiniDump|#24)/     /*
Process dumping method using comsvcs.dll's MiniDump */

    $ = "AmsiScanBuffer" ascii wide base64           /*
AMSI Bypass */
```

```
    $ = "AmsiScanBuffer" xor(0x01-0xff)                    /*
AMSI Bypass */

    $ = "%%%%%%%%%%%%######%%%#%%####%
&%%**#" ascii wide xor /* SeatBelt */

  condition:

    1 of them

}
```

You may also get the CVEs (Common Vulnerabilities and Exposures) from MITRE, however using the Exploit Prediction Scoring System (EPSS) is highly recommended. The industry continues to face challenges with vulnerability management, despite substantial expenditures in information security technology and research over the previous several decades. To be more precise, vulnerability management tools rely on a combination of severity ratings and subjective expert judgment to prioritize remedial activities. With more and more vulnerabilities that the typical business has to fix, this problem is becoming worse. Predicting the likelihood of a vulnerability being exploited in the wild during the first twelve months following public disclosure, this article presents the first open, data-driven approach for evaluating vulnerability risks. This scoring system's user-friendliness belies its ability to provide precise exploitation estimates (ROC AUC = 0.838) without requiring practitioners to own specialist tools or software. The system is also adaptable enough to receive upgrades as new and improved data becomes available. The Exploit Prediction Scoring System is the name we give to this system. (Part 13) A validation step is necessary since the end product will be a lengthy report without a prioritized action plan. This is why it would be a good idea to include Exploit DB so that discovered vulnerabilities may be immediately compared to entries in Exploit DB. By using this method, we can identify vulnerabilities with known exploits and fix them right away. The remedial efforts may be better distributed if they are focused on these high-risk vulnerabilities.

III.MITRE ATT&CK

Threat intelligence is useful because it can help stop cyberattacks before they happen by recommending countermeasures. An

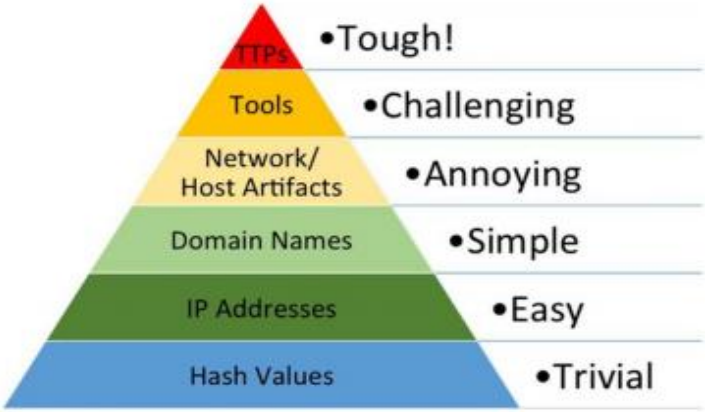

Fig. 2. David J. Bianco's Pyramid of Pain

David Bianco's Pyramid of Pain provides a thought-provoking framework for comprehending this idea. This model shows the relationship between several indications used to identify enemy operations and the difficulty (or "pain") of neutralizing such signs. Instead than focusing on the instruments used by an enemy, we may tackle all of their behaviors by identifying and countering their strategy, tactics, and procedures (TTPs). Because it makes enemies work so hard to adjust to new methods, this all-encompassing method boosts efficacy. So, they have to

learn new abilities, which is very hard, so they can respond quickly to enemy TTPs or prevent them. The results of this assessment, when combined with an examination of sector-specific TTPs, provide practical insights that direct efforts towards areas where they may erect the highest barriers to entry for would-be attackers. Part 2.

IV.HEATMAPS

Security must be nuanced to account for the wide variety of detection environments, each of which is structured and designed differently. Opponents with bad intentions are obviously not a monolith. The strategies and methods used by people who have several clients in mind are radically different from those used by those who have one customer in mind. With the goal of making the best possible suggestions, we will set out on an extensive trip that will lead us through the complexities of systems and give us a better picture of the environment in which we work. In this growing database of threat information, you will find extensive historical records. The most notable dangers that have successfully compromised other organizations are still under scrutiny. Thoroughly document, examine, and organize the TTPs (Tactics, Techniques, and Procedures) used in these occurrences. Using the highly regarded MITRE ATT&CK methodology as a guide, the map reveals a heatmap that shows the methods that each company faces the most danger from. Our defenses need to be flexible since cyber threats are always changing. True resilience is discovered in the ongoing attention and action, but the retrospective TTP technique based on Threat Intelligence (TI) sets the groundwork. In addition to developing and refining detection algorithms using past data, the detection engineering team is tasked with keeping an eye on the dynamic threat environment, finding new entry points for attacks, and coming up with effective countermeasures. The capacity to quickly adjust is crucial for success in the ever-changing field of cybersecurity. When sufficient safeguards are in place, threat intelligence proves its worth by preventing cyberattacks. When covering the whole range of an adversary's actions rather than simply their tools, and when able to identify and mitigate TTPs. This is perfect in terms of efficiency alone. If you can anticipate and respond quickly to adversaries' TTPs, you can make them perform the most labor-intensive thing they can think of: learn new behaviors.[14]

V. SIGMA RULES

Since the previously established heat map is based on the only objective framework that can technically represent detection capabilities according to the MITRE ATT&CK, it offers an ideal roadmap for action. Creating a suitable notification system is the assignment at hand. As a standard for all SIEM systems, Sigma was developed (Florian Roth, 2020a). It provides a community-driven collection of threat detection rules, a Python toolkit for parsing rules and translating them to supported SIEM formats, and a customizable rule structure in YAML format. The project's goal is to make it easier to share event log IoC and avoid vendor lock-in by translating Sigma rules into any vendor's format. Instead of matching patterns or issuing alarms, Sigma serves as a translation layer and platform for exchanging IoC data. For this reason, it is essential to have an alerting capability equipped SIEM system.SIEM systems are notoriously resource-intensive, requiring regular management, license fees, and expensive hardware [15]. Usually, a specialized group has to keep an eye on these systems, make sure they're up-to-date, re-index data when necessary, know about new features and deprecations, deal with support calls, and make sure the supply chain is intact. Contrarily, making one's own tool is often seen as a more difficult choice, particularly in business and military contexts. There are two main goals of this article. To start, I'll make the case that a tiny, purpose-built streaming tool may successfully manage jobs that are normally performed by much bigger databases. Part 16.

I. CONCLUSIONS

It is of utmost importance that IT systems maintain an adequate level of cybersecurity. Timely identification of assaults is crucial for preserving company continuity, which is why monitoring is increasingly becoming more important. Internal traffic on IT networks also generates a great deal of technical threat intelligence indications, thus security measures are taken to ensure their protection.

The prompt deployment of safeguards is greatly enhanced by cooperation and the exchange of data within the cybersecurity domain. Here, cybercriminals re-use technological indications that they have already obtained. When attackers' IP addresses, domains, signatures, and vulnerability information are discovered and publicized, it might limit their operating capabilities. Therefore, it is critical to have a vulnerability management program in place that includes the proper resources, staff, and processes. It may be a lengthy process to build new patterns of behavior for attackers, but our software will force them to do it if it works properly and monitors their behavior.

II. REFERENCES

[1.] ATTACKIQ ENTERPRISE (2022): What is the Pyramid of Pain?, Online: https://www.attackiq.com/glossary/pyramid-of-pain/

[2.] Black Cell Ltd. (2023): Sector specific MITRE ATT&CK heatmaps for detection engineering. Online: https://github.com/blackcellltd/Heatmaps

[3.] Oasis (2018) STIX™ Version 2.1 Specification. Online https://docs.oasis-open.org/cti/stix/v2.1/cs02/stix-v2.1-cs02.html.
[4.] AbuseIPDB. Online: https://www.abuseipdb.com/
[5.] Tor list: Online: https://www.dan.me.uk/torlist/
[6.] Ipdata Online: https://docs.ipdata.co/docs/proxy-tor-and-threat-detection
[7.] Catchpoint: DNS sinkhole Online: https://www.catchpoint.com/network-admin-guide/dns-sinkhole
[8.] Black Cell Ltd. (2020): Domain Name Generating Algorithms Detection. Online: https://blackcell.io/wp content/uploads/2023/04/BC_DGADetection_WHITEPAPER.pdf
[9.] Github (2023): Dnstwist.Online: https://github.com/elceef/dnstwist
[10.] Nattawat Khamphakdee, Nunnapus Benjamas, Saiyan Saiyod (2015), Improving Intrusion Detection System Based on Snort Rules for Network Probe Attacks Detection with Association Rules Technique of Data Mining, Journal of ICT Research and Applications.
[11.] Zeek Documentation (2023 Online: https://docs.zeek.org/en/master/about.html
[12.] Github (2023) Online: https://github.com/Neo23x0/god-moderules/blob/master/godmode.yar#L1
[13.] ACM Digital Library (2021) Exploit Prediction Scoring System (EPSS) Online: https://dl.acm.org/doi/pdf/10.1145/3436242
[14.]Bader Al-Sada, Alireza Sadighian, Gabriele Oligeri (2023). Analysis and Characterization of Cyber Threats Leveraging the MITRE ATT&CK Database. Journal of ICT Research and Applications.
[15.] Ashok Manoharan, Mithun Sarker (2022). Revolutionizing Cybersecurity: Unleashing the Power of Artificial Intelligence and Machine Learning for Next-Generation Threat Detection. International Research Journal of Modernization in Engineering, Technology and Science, Vol. 4, Issue 12, December 2022, pp. 2151. e-ISSN: 2582-5208. Online: https://www.irjmets.com.
[16.] Kont, M., & Pihelgas, M. (2020). IDS for logs: Towards implementing a streaming Sigma rule engine, NATO Cooperative Cyber Defence Centre of Excellence (CCDCOE)