URL Evaluator: Semi-automatic evaluation of suspicious URLs from honeypots

Michaela Novotná
Brno University of Technology
Brno, Czech Republic
xnovot2i@stud.fit.vutbr.cz

Václav Bartoš
CESNET
Prague, Czech Republic
bartos@cesnet.cz

***Abstract*—Botnets often rely on malicious URLs to distribute malware payloads over HTTP. Identifying these URLs is critical for network defense, as it enables the detection or blocking of access from within the network, thereby preventing potential malware infections. A promising approach for uncovering URLs used for malware distribution involves analyzing data from SSH honeypots. However, not every URL observed in a honeypot log is necessarily malicious. In this paper, we present the ”URL Evaluator” system, which automates the extraction and analysis of suspicious URLs from SSH honeypot data. It employs a semi-automated evaluation process, which leverages multiple data sources and methods and escalates to human operators only when necessary. Confirmed malicious URLs are then used in a network monitoring system to detect any accesses to such URLs from within the defended network. Any such access is automatically reported to the responsible administrator or security team. Additionaly, the system contributes newly found malicious URLs to a large community blacklist. The paper describes the system architecture, key components, and its operational results.**

I. Introduction

The spread of malware and botnets often involves downloading a malicious payload over HTTP. Therefore, and important aspect of security operations is identifying the URLs hosting malware, so any attempt to access such an URL from within the defended network can be blocked or at least detected and reported as a potential compromise of a device. While there are public lists of such malicious URLs (e.g. URLHaus by abuse.sh[1]) that can be used for this purpose, no such list contains all active malicious URLs and it takes time for a new URL to be submitted to the list. Therefore, it is always beneficial to look for other sources.

One promising approach for uncovering malicious URLs involves the deployment of SSH honeypots, which simulate vulnerable systems to attract attackers. These honeypots capture extensive session data from malicious actors, often containing URLs that attackers use to fetch malicious software to the compromised system. However, not all URLs extracted from SSH honeypot logs are necessarily malicious. Some may point to benign content (e.g. a script to measure connection speed), or to content that might be used in unwanted ways, but is not inherently malicious (e.g. a cryptomining software), making it essential to evaluate each URL's potential risk.

Given the volume of data collected, manually analyzing these URLs is impractical. Therefore, a systematic, largely automated process for evaluating URLs is required.

In this short technical paper, we introduce such a system called the *URL Evaluator*, which was developed at CESNET, the operator of the Czech National Research and Education Network (NREN), in collaboration with Brno University of Technology. The system extracts suspicious URLs from data of several honeynet[2] projects, semi-automatically evaluates them, and outputs those URLs confirmed to be hosting malware. The system is accompanied by a set of tools to detect and report attempts to access such URLs within our network.

Source codes of the system are available on GitHub[3].

II. System overview

The architecture of the URL Evaluator system is depicted in Figure 1. The core of the system is represented by a database containing information about each evaluated URL. SQLite is used for this purpose due to its simplicity.

The input consists of several modules which gather suspicious URLs from different sources (detailed in Section III). They write new URLs into the database and mark them for evaluation.

The main part is the evaluation process. It takes new URLs from the database and tries to automatically assess them by looking them up in a blacklist, downloading the URL content and looking up its hash, etc. The process is described in detail in Section IV. It also periodically checks each URL whether it is still active (i.e. the server responds and a content can be downloaded) or not.

In case none of the automatic evaluation methods is able to determine whether the URL is malicious or not, it is marked as unclassified and the decision is left for a human analyst – a member of the Security Operations Centre (SOC). The analyst accesses the system via a web interface, which is further described in Section V.

The URLs classified as malicious are stored into an instance of MISP [3] (where they can be combined with data from other sources) and then used within a network monitoring system to detect and report any attempt to access a malicious

[1] https://urlhaus.abuse.ch/

[2] Honeynet is a network of honeypots – decoy systems designed to attract attackers and record their activities on the simulated system.

[3] https://github.com/CESNET/URL_Evaluator

978-3-903176-66-9 ©2024 IFIP

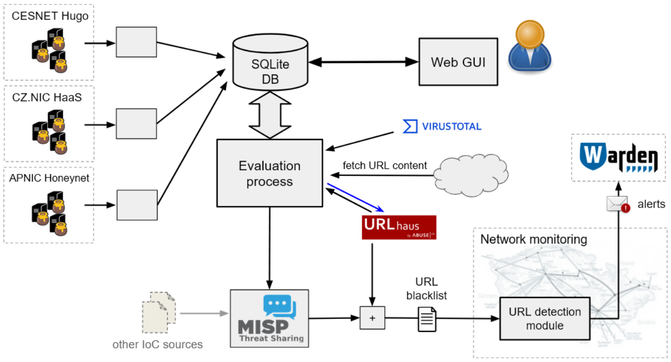


Fig. 1. Architecture of the URL Evaluator system

URL from a device in the monitored network. Also, all newly identified malicious URLs are published by sending them to the URLhaus blacklist.

The following sections detail individual parts of the system.

III. Sources of suspicious URLs

The system gathers data from the following three honeynet projects, which all operate multiple SSH honeypots:

- Hugo honeynet by CESNET: This project operates about a dozen of honeypots, mostly in university networks in the Czech Republic. They simulate different types of services, including SSH. All incoming connections are reported to CESNET's alert-sharing system Warden [2], from which the input module of the URL Evaluator reads them. The reports contain complete SSH session logs, i.e. lists of commands the attacker tries to perform on the server. The module analyzes these logs and extracts URLs that occur within the `wget` or `curl` commands.
- CZ.NIC HaaS project[4]: Volunteers can install a *HaaS proxy*, which forwards all traffic on port 22 (SSH) to a central server, operated by the CZ.NIC organization, which acts like a honeypot. Data about all such incoming connections, including full session logs, are published as daily dumps on the project website. An input module downloads these dumps and extracts the URLs from sessions in the same way as in the previous case.
- APNIC Community Honeynet Project: A honeynet project led by APNIC [4]. This project does not publish raw data of the honeypot connections and sessions; instead a set of feeds[5] with processed and aggregated data are provided. One of the feeds contains the URLs from which attackers downloaded some content. A module of URL Evaluator downloads this feed every day and submit those URLs for evaluation.

Examples of URLs obtained this way are shown below:

```
http://121.40.85.244/ns3.jpg (malicious)
http://pen.gorillafirewall.su/lol.sh (malicious)
https://ipinfo.io/org (harmless)
```

When available, full session logs are also stored in the URL Evaluator database, so they can be shown to the analyst in the web interface if needed.

Thanks to the modular architecture, it is easy to add any other source of suspicious URLs (not necessarily from honeypots) in the future. It is also possible to add URLs manually using the web interface.

IV. Semi-automatic evaluation

The goal of the evaluation is to assign to each URL one of the following categories:

- *Malicious* - The URL has been proven to contain harmful or malicious content.
- *Harmless* - The URL contains a content that is not inherently malicious.
- *Unreachable* - No content can be downloaded from the URL. Either the server does not respond or it returns an error.
- *Unclassified* - This is used when the evaluation process is not able to reliably assign any of the other classes. Such URLs require manual classification.
- *Invalid* - The string passed is not a valid URL.

The evaluation process is semi-automatic – it first tries to classify an URL automatically by multiple methods. If it does not lead to a conclusive result, manual evaluation is performed through a web interface.

The automatic classification process consists of several steps. If any of them assigns a class, the process stops there, otherwise it proceeds with the next step.

1) First, the URL format is checked. If the string passed for evaluation does not match the required format, it is classified as *invalid*.
2) Then, the URL is checked against an URL blacklist. Currently, the URLhaus blacklist is used, as it is probably the largest of its kind and, according to our experience, reliable. More blacklists can be easily added in the future, if needed. The list is re-downloaded every 15 minutes to keep it up-to-date. If the URL is found on the list, it is automatically classified as *malicious* and further processing is skipped.
3) Next, the URL is checked on VirusTotal[6] via its API. VirusTotal is an online service used by security practitioners to scan suspicious files by multiple antivirus engines and aggregate their results. Since it is possible to submit a file via its URL, VirusTotal also has information about URLs and allows to search them and get assessment of their content – which is what we do in this step. The results contain statistics about the number of detection engines which classified the URL/file as malicious, harmless, unknown, etc. If over 80 percent of them classifies the URL as malicious, we also assign it the *malicious* class and stop further processing.
4) If none of the URL lookups lead to a result, the URL Evaluator attempts to download the content of the URL.

[4] Honeypot as a Service, https://haas.nic.cz/
[5] https://feeds.honeynet.asia/
[6] https://www.virustotal.com/

If the attempt fails, either because the server does not respond or it returns an error, the URL is classified as *unreachable*.

5) If the content is successfully downloaded, a SHA1 hash is computed from it. The VirusTotal API is then used again, but now to find a file matching the hash. This is because a single malware sample can be hosted on multiple locations, so there is a high chance that even though the URL is new, its content is an already known piece of malware. If the hash is known to VirusTotal, the statistics are evaluated similarly as with URL search – if more than 80 % of engines which classified the content mark it as malicious, we label it as *malicious* as well. If some engine marks is as harmless and none of them as malicious, we assign it the *harmless* class. In all other cases, the results are inconclusive, so we label the URL as *unclassified*, which means the decision is left for a human analyst.
6) If VirusTotal does not know the hash, it is looked up in the database provided by the MalwareBazaar[7] project. This database contains information about various malicious software, including hashes of known samples. If the content's hash of the evaluated URL is found in this database, the URL is classified as *malicious*.
7) Otherwise, it is labeled as *unclassified*.

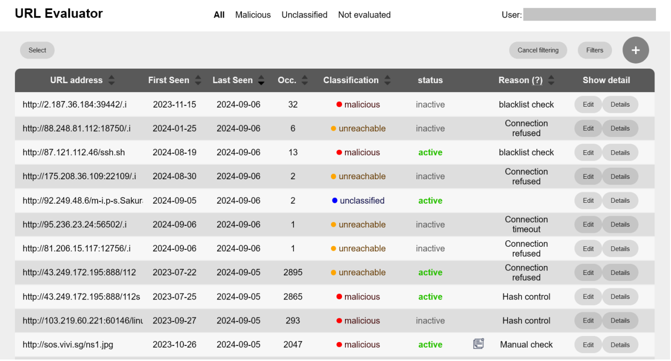
To document which step of the automatic analysis led to the classification of an URL, the system stores a note with the reason for the classification. It also stores some pieces of information obtained during the evaluation process, which may be useful for the manual analysis in case it is needed. This includes the MIME type of the content, its size and SHA1 hash, as well as the results of the VirusTotal queries.

It is quite common that the downloaded content is just a shell script downloading another content from a different URL (so called *downloader*). In such case, URL Evaluator searches the script for URLs in the same way as it processes the sessions from honeypots. Any URLs extracted this way are inserted into the database and queued for evaluation.

Malicious URLs are often active only for a limited time and then become inaccessible. Therefore, besides the automatic evaluation of newly observed URLs, which was described above, the system performs regular checks of each URL to find out whether it is still reachable (active) or not. Every day the system tries to connect to each URL and based on the result it marks it as active or inactive. The date it was last seen active is also stored in the database.

V. Web interface and manual evaluation

Security analysts can access the data in the URL Evaluator database via a simple web interface. Its main purpose is to allow the manual classification, but it can also be used to list, search and filter existing URL records, show details of each record, and to submit new URLs for automatic evaluation. A screenshot of the main page is shown in Figure 2.



URL Evaluator

User: [redacted]

URL address	First Seen	Last Seen	Occ.	Classification	status	Reason (?)	Show detail
http://2.187.36.184:39442/.i	2023-11-15	2024-09-06	32	● malicious	inactive	blacklist check	Edit Details
http://88.248.81.112:18750/.i	2024-01-25	2024-09-06	6	● unreachable	inactive	Connection refused	Edit Details
http://87.121.112.46/ssh.sh	2024-08-19	2024-09-06	13	● malicious	active	blacklist check	Edit Details
http://175.208.36.109:22109/.i	2024-08-30	2024-09-06	2	● unreachable	inactive	Connection refused	Edit Details
http://92.249.48.6/m-i.p-s.Sakur	2024-09-05	2024-09-06	2	● unclassified	active		Edit Details
http://95.236.23.24:56502/.i	2024-09-06	2024-09-06	1	● unreachable	inactive	Connection timeout	Edit Details
http://81.206.15.117:12756/.i	2024-09-06	2024-09-06	1	● unreachable	inactive	Connection refused	Edit Details
http://43.249.172.195:888/112	2023-07-22	2024-09-05	2895	● unreachable	active	Connection refused	Edit Details
http://43.249.172.195:888/112s	2023-07-25	2024-09-05	2865	● malicious	active	Hash control	Edit Details
http://103.219.60.221:60146/linu	2023-09-27	2024-09-05	293	● malicious	inactive	Hash control	Edit Details
http://sos.vivi.sg/ns1.jpg	2023-10-26	2024-09-05	2047	● malicious	active	Manual check	Edit Details

Fig. 2. Main page of the web interface containing the list of evaluated URLs and their associated information

An analyst periodically checks the web interface. When an unclassified URL appears, he or she looks at its details and assign the final class (optionally with a comment). The URL detail page shows all the data obtained during the previously attempted automatic classification, as well as all the SSH sessions the URL was observed in. Sometimes this is enough to decide about the purpose of the URL (together with the analyst's experience), sometimes more investigation using other systems or data sources is needed.

On most days, there are just a few URLs that need manual classification, sometimes there are none, so it does not cost much of the analyst's precious time.

VI. Data utilization

As mentioned above, when a URL is classified as malicious, it is submitted to URLhaus if it is not there yet. This ensures that the information is quickly shared with the broader cybersecurity community.

All malicious URLs are also stored in our instance of MISP which is used as a source of indicators to be searched in network traffic.

For detection, we utilize our extensive flow-monitoring infrastructure. This includes a set of dedicated probes (using *ipfixprobe*[8] software), one on each of our border links, which are able to monitor all traffic on 100 Gbps links without sampling and support extraction of information from application-layer protocols. In this case, we utilize data from HTTP headers[9]. These L7-extended flow data are then processed by various modules of the NEMEA[10] system [1]. For the detection of accesses to malicious URLs, we implemented a simple NEMEA module which takes a list of URLs and compares any incoming flow record with HTTP data against it.

The list is made by combining two sources. The first one is the MISP instance containing malicious URLs from the URL Evaluator. Only the currently active URLs are taken (the "IDS"

[7] https://bazaar.abuse.ch/

[8] https://github.com/CESNET/ipfixprobe

[9] Note that, according to our experience as well as data in URLhaus, most URLs used to spread malware are plain HTTP. We would not be able to observe URLs in HTTPS traffic, but the use of HTTPS for malware distribution is still very rare.

[10] https://nemea.liberouter.org/

TABLE I
NUMBER OF URLS BY CLASSIFICATION RESULT

Label	Count	% of total
Unreachable	3403	87,2 %
Malicious	472	12,1 %
Harmless	28	0,7 %

TABLE II
NUMBER OF URLS BY REASON FOR THE CLASS ASSIGNMENT

Classification reason	Count	% of total
Connection refused or timeout (→ unreachable)	3361	86,1 %
Error response (→ unreachable)	41	1,1 %
Blacklist check (→ malicious)	255	6,5 %
Hash control (→ malicious/harmless)	171	4,4 %
Manual check (→ any)	75	1,9 %

flag is used in MISP to store this information, it mirrors the *active* flag from the URL Evaluator). The second source is a copy of the latest version of the URLHaus blacklist (again, only active URLs are used). Thus we also include a large number of URLs from the community, which we do not see on the honeypots.

When the module detects a successful connection to one of the URLs on the combined list, it generates an alert describing the event and sends it into CESNET's alert-sharing system, Warden. There is already a complex automation machinery that ensures that each such alert is reported to the person or team responsible for the security of the network from which the problematic connection originates (e.g. a CERT team of a university), so they can investigate it. A detailed explanation of this automatic reporting is, however, out of the scope of this paper.

If used in another environment (not an ISP-level network), accesses to malicious URLs might not only be detected, but also blocked by some kind of Intrusion Prevention System.

VII. RESULTS

The system has been developed and deployed iteratively at CESNET since the end of 2023. This section presents some statistics from a three month period (June to August 2024) in which all described components and functions were already fully operational.

During this time frame, 3 903 unique URLs were analyzed. The distribution of classification results is shown Table I. The results show that most URLs are already unreachable at the time of first evaluation, which is usually shortly after they are first observed at the honeypots (the delay depends on the source honeynet project, for the CESNET one, it is in the order of minutes, for other ones it can be up to a day since they only provide daily dumps). So, it seems most of the malicious URLs get offline very quickly, some of them might even never have worked (or maybe there are some filters, e.g. geolocation-based, which do not allow connections from our server while allowing it from elsewhere – this is a topic for future research). Still, there are hundreds of active malicious URLs found. The 28 harmless URLs confirm our assumption that not every URL observed in honeypot logs should be automatically considered harmful.

Table II shows the distribution of reasons for the class assignment. Only 75 URLs had to be checked manually (less than one per day on average).

Out of the 3903 URLs observed, only 110 of them are still active (reachable) at the end of the three-month period. The average lifespan of URLs, i.e. the time between the first observation of an URL and the last time it was active, is 6.3 days. Most URLs become inactive in less then a day, but a few ones are active for several months.

During this three-month period, 70 malicious URLs were successfully submitted to URLhaus as new contributions.

Regarding the detection subsystem – most alerts about a device accessing a malicious URL are caused by honeypots in our network, which are repeatedly attacked and used to download malware. We have implemented a whitelist to suppress such alerts. Nevertheless, the system has also detected several real incidents where poorly secured devices were compromised by different types of malware. Without this detection system, some incidents would probably remain undetected or detected much later.

VIII. CONCLUSION

The URL Evaluator and related tools demonstrate how a set of relatively simple programs and scripts can be combined into an effective pipeline for automating cybersecurity operations – from data gathering, evaluation and filtering, to the detection of incidents within the network. Manual intervention is only required occasionally when there is insufficient information for reliable automatic evaluation, and, of course, when a malware infection is detected and must be addressed.

The described system has been successfully deployed at CESNET and already helped to detect several malware infections within member networks. It has also contributed many new URLs to the community blacklist URLHaus.

ACKNOWLEDGEMENT

This work was partially supported by the SOCCER project, which is funded under Grant Agreement No. 101128073 and supported by the European Cybersecurity Competence Centre.

REFERENCES

[1] Tomas Cejka, Vaclav Bartos, Marek Svepes, Zdenek Rosa, and Hana Kubatova. NEMEA: A framework for network traffic analysis. In *2016 12th International Conference on Network and Service Management (CNSM)*, pages 195–201, New York, NY, USA, 10 2016. IEEE.

[2] Pavel Kacha, Michal Kostenec, and Andrea Kropacova. Warden 3: Internet Threat Sharing Platform. *International Journal of Computers*, 10, 2016.

[3] Cynthia Wagner, Alexandre Dulaunoy, Gérard Wagener, and Andras Iklody. MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform. In *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, WISCS '16, pages 49–56, New York, NY, USA, 2016. ACM.

[4] Adli Wahid. The APNIC Community Honeynet Project (blog post), 2019. Avaiable at: https://blog.apnic.net/2019/09/17/the-apnic-community-honeynet-project/.