UNIVERSITY OF TARTU
Institute of Computer Science
Cyber Security Curriculum

Guillaume Brodar

Analysis of Exploit-kit Incidents and Campaigns Through a Graph Database Framework

Master's Thesis (30 ECTS)

Supervisor(s): Toomas Lepik
Raimundas Matulevicius

Tartu 2017

Analysis of Exploit-kit Incidents and Campaigns Through a Graph Database Framework

Abstract:

In today's threat landscape, the delivery of malware via browser exploit-kits poses specific challenges to the forensics analyst and from a defensive perspective in general. Web browsers offer a large surface of attack through their own implementation, the plugins they offer and the operating systems that they rely on.

However, when looking at network traffic captures, they also leave specific traces that the analyst can identify but those still wouldn't be enough to clearly determine if an infection was successful or not. Isolating these traces currently requires a lot of manual work and a complete analysis will also have to rely on third-party information in order to give a clear picture and understanding of the incident.

A great deal of automation can be achieved here by using public APIs such as VirusTotal, whois databases, IP blacklists, etc during the analysis and a first part of our work is dedicated to that.

From our perspective, we also see that the use of graph databases can be of a great help when putting together information from different sources that hold relationships with one another and a second part of our work will be to demonstrate that a graph database approach can be used to analyze single incidents as well as the delivery infrastructure of specific exploit-kits or malware campaigns that are spread by specific actors.

We will then show that this approach reveals patterns and clusters from which decisions can be made from a defensive perspective.

Keywords:

Exploit-kit, graph databases, malware analysis, incident response, network forensics.

CERCS: P170 Computer science, numerical analysis, systems, control

Ründetarkvara intsidentide ja levitamisviiside analüüs graafilise modelleerimise abil

Lühikokkuvõte:

Tänapäeval peaaegu igas seadmes kasutatav veebilehitseja on soodsaim keskkond kurivara levitamiseks, kuna võimaldab ründe-programmidel ekspluateerida hulgaliselt erinevaid vahendeid alates veeblihitseja üldisest ülesehitust ja pluginatest kuni sellega seotud operatsioonisüsteemi eripäradeni. Seega on veebilehitsejates kasututavad ründetarkvara komplektid üks levinumaid kurivara esinemisvorme ning seetõttu ka märkimisväärseks probleemiks nii digitaalse ekspertiisi spetsialistidele kui kurivara tõrje valdkonnas üldiselt.

Kuigi salvestatud võrgupakettidest võib leida indikaatoreid, mis võimaldavad analüütikul tuvastada kurivara olemuse, pole need tihti piisavad süsteemi kompromiteerituse ulatuse määramiseks. Taoliste indikaatorite üksipulgi läbitöötamine on aeganõudev protsess ning intsidendi analüüs ja terviklik ülevaade sõltuvad seljuures peamiselt kolmandate osapoolte kaudu saadud infost.

Analüüsi käigus saab protsesse küll osaliselt automatiseerida kasutades avalikult kättesaadavaid programme nagu VirusTotal, WHOIS ja erinevad (IP, domeeni või veebilehe põhised) mustad nimekirjad. Seda osa protsessist tutvustab ka antud töö esimene pool.

Graafiline modelleerimine aitab kaasa erinevatest allikatest pärineva info ja seoste koondamisel ühtsesse ja kergemini hoomatavasse vaatesse. Töö teine osa keskendubki viisidele, kuidas kasutada graafilise modelleerimise võimalusi nii üksikute intsidentide analüüsiks kui konkreetse ründetarkvara erinevate levitamisviiside kuvamiseks täpsema tervikpildi saamise eesmärgil.

Töö viimane osa demonstreerib moodusi, kuidas antud lähenemine aitab välja tuua seosed ja seaduspärad, mille põhjal on võimalik teha pahavara tuvastamise ja tõrjumisega seonduvaid otsuseid ja järeldusi.

Võtmesõnad:

graafiline modelleerimine, visualiseerimine, kurivara analüüs, pahavara analüüs, intsidendi haldus, arvutivõrkude analüüs

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Table of Contents

List of Figures

List of Tables

1 Introduction

When looking at the threat landscape that general users and enterprises face, the threat of client-side attacks is definitely one of the major areas of focus. Client-side attacks are relying on some sort of social-engineering as a supporting mechanism to direct the end-user to run unintended code.

Historically, the spread of malware has started via the exchange of storage media. The development of the Internet has seen this spread to migrate to email and file-sharing services like BitTorrent. A lot of effort has been put by anti-virus vendors and security platform providers to secure those distribution channels and there is today a good understanding of these infection vectors. The direct delivery of malicious binaries via email is a strategy that is not as effective as it used to be and the delivery mechanisms of malware have now moved to the web and actively target the web browser.

The web browser presents an interesting surface of attack as it not only exposes itself to a malicious server, it also exposes the plug-ins that are used to display rich-media like Adobe Flash, Acrobat PDF, Java, etc and also provides information about the operating system. From an attacker's perspective, these provide avenues that can all be exploited.

Web-based attacks, also known as drive-by downloads, are attacks that are known to be spread either via phishing by directing the user to an unsuspecting link or in the worst-case scenario via a malvertising campaign where the malware distribution channel is piggy-backing on a legitimate online advertising network where the web browser is targeted either directly by malicious banners or from a remote location through hidden iframes or redirection to a malicious Content Delivery Network (CDN) as shown by A. J. Sood and R. Enbody in [1].

In [2], C. Grier, L. Ballard *et al.*, have extensively exposed the emergence of a criminal industry with a business-model centred around the distribution of malware via drive-by downloads where exploit-kits play a central role.

This work brings a conclusion to my cursus in the Master's of Science in Cybersecurity and Digital Forensics at the Technical University of Tallinn during which I developed a specific interest for the topics of malware analysis as well as network and system forensics. While practising my analysis skills on network traffic captures (PCAP files), I have realized that the extraction of information contained in the PCAPs regularly follows the same steps but also that the extracted information often needs to be cross-checked with external sources. It appears quite quickly that a great deal of time can be saved by scripting those tasks and by making use of publicly available APIs to enrich the data that we are extracting from those PCAPs.

On another note, I have also witnessed the general development of use-cases for graph databases in the realm of cybersecurity in things as diverse as fraud detection, impact and root-cause analysis. After exploring the details of the Neo4j graph database covered in [3] by I. Robinson, J. Weber *et al.*, I have assessed that adopting a graph-based approach could be specifically valuable when studying web-based attacks that rely on exploit-kits. This work is thus an exploration in merging a data analysis technique with a recurring task in malware analysis and forensics.

For this, a Neo4j 3.1.1 instance on Ubuntu 16.04 with the default visualization engine that comes with it, will be used together with Bro IDS 2.4.1 as a way to parse and log the events and artifacts that a PCAP file contains and a couple of python 2.7 scripts have been written in order to automate the extraction of artefacts from the Bro logs, enrich the extracted data

through external APIs and create corresponding entries in the Cypher Query Language (CQL) that Neo4j uses.

The set of PCAPs that is used here is taken from the malware-traffic-analysis.net website where they are analysed and reported case by case. Other sets from Contagio (contagiodump.blogspot.com) and from Broadanalysis (www.broadanalysis.com) have also been used in order to validate that the analysis method would show consistent results across different samples.

The hypothesis that has guided this work is that exploit-kit incidents leave specific traces in traffic captures because of the finger-printing mechanisms that they are using to target the browser. The use of a graph database as a supporting back end should make the detection of these incidents easier and it should be possible to derive specific search patterns that not only highlight the use of exploit-kits but also provide actionable threat intelligence in a wider context than isolated incidents.

The first part of this thesis will cover how exploit-kits can be viewed from a forensics perspective and builds a case for the use of a graph-based approach from a visual perspective as well as for isolating specific patterns in a series of events. The second part will present the dataset used in this work and will present the build-up of our system and its data-model. The third part will show in-situ what information can be extracted from our analysis method when working with isolated incidents and whole campaigns alike.

All source code for the scripts we have written as well as the database exports of the graphs we are presenting is available at: https://github.com/gbrodar/pcap4j.

It is important to note that the code that is presented here is not production-grade and should only be viewed as a proof-of-concept.

This work has been done under the co-supervision of Toomas Lepik from the Tallinn University of Technology and, of Raimundas Matulevicius from the University of Tartu. I would like here to thank both of them and the institutions that they are representing.

2 The specifics of exploit-kit based incidents – environment, challenges and response

2.1 Exploit-kits: definition and mode of operation

As defined by the ENISA in [4], exploit-kits "are an automated means for the deployment of malware and hold a key role in infection vectors: they check available vulnerabilities in the targeted environment and install the appropriate malware that exploits the detected vulnerabilities", as such they are specifically targeting web browsers and their associated plug-ins.

What exploit-kits essentially provide to the attacker is a fingerprinting service and a delivery mechanism: the fingerprinting service essentially identifies the client system and decides if and how a payload delivery should happen. The client-side can be fingerprinted based on its source IP address, Operating System, Browser version and plug-ins installed and when a compatible exploit is found, the victim gets injected a malicious payload of various types (banking trojans, ransomware, RATs…).

A generic depiction of a browser attack takes the following 3-steps and involves the following elements shown in Figure 1.

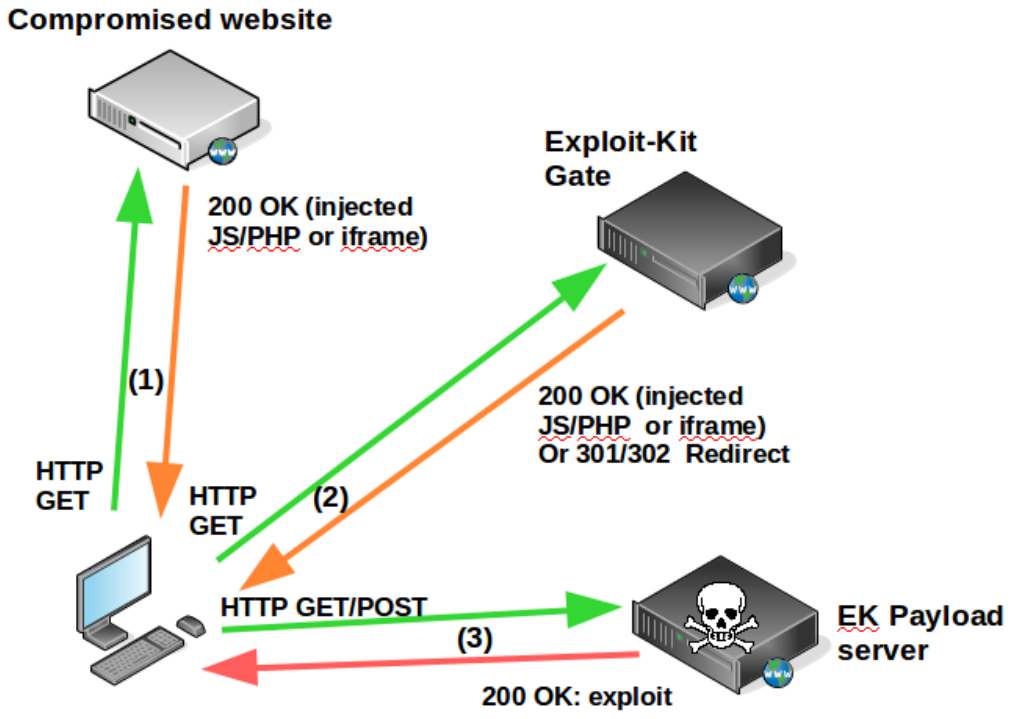


Figure 1: Generic exploit-kit behaviour and exploit delivery mechanism

As such, the events occurring in the first and second steps described in Figure 1 are not malicious. However, they are essential parts of the infection cycle and can not be taken away from the exploitation phase that happens in the step 3.

While exploit-kit gates and payload servers are hosted on infrastructure that is directly leased by the attackers, they make use of compromised legitimate websites to redirect the client towards those malicious servers. In [5], the CERT-UK is identifying "compromised Content

Management Systems (Drupal, Joomla…) with poor access controls, insecure forum software" and malvertising campaigns as the most usual places where attackers are able to inject malicious scripts.

The same technique can also be used in a targeted context either by giving an incentive to a user to click on a specific link via phishing or by compromising a website that the targeted user visits on a regular basis and would not suspect it to have a malicious component.

Redirection to the exploit-kit gate relies either on obfuscated iframes that evade known detection techniques or by "302 cushioning" as described in [5].

Once the client has been redirected, the fingerprinting and exploit delivery process takes part as shown in Figure 2.

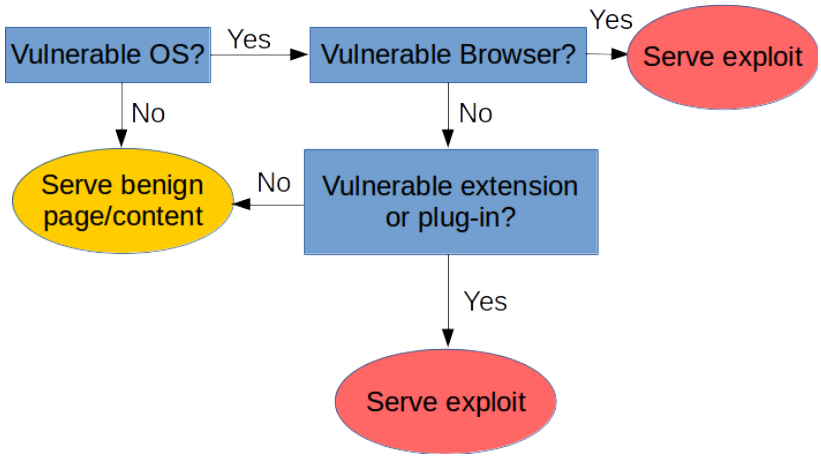


Figure 2: Exploit-kit fingerprinting workflow

In [6], V. Kotov and F. Massacci have found that exploit-kits are using the following techniques in general: user-agent detection, IP blacklisting, user-agent validation, exploit selection, exploit obfuscation and executable delivery. The detection and validation phase can be independent one from the other: the functions of identifying the client are then achieved by an exploit-kit gate ("EK-gate") that then redirects to another location where the exploits and payloads are located, this scheme adds another layer of flexibility and modularity when deploying an "Exploit-kit as a Service".

The research done in [6] also show that in average the exploit-kits they studied were carrying a mean number of 11.1 exploits per kit with the majority of the exploits being already 1 or 2 years old and they target browsers (Internet Explorer, Firefox, Opera) as well as their plugins (Java, Adobe Flash, Adobe Acrobat…) or underlying operating system (Microsoft Windows). While this might read like only commodity exploits targeting poorly-patched systems, industry research is showing on a regular basis that zero-days are also known to be deployed, sometimes even in targeted contexts, as from Symantec in [7] and [8] or FireEye in [9].

However, the research in [6] is from 2013 and the number of exploits carried as well as the type of vulnerabilities targeted and their staging possibly needs an update given the rapid changes in the threat actors motivations and technical sophistication.

2.2 Incident response and forensics perspective

From an incident response and digital forensics perspectives, while their objectives differ, a complete understanding of the course of events and of the infection scenario is needed in order to provide a valid response to an incident.

In incident response mode, the objectives are to close every identified way by which the same incident could reoccur but also define from there responses that look at strengthening the security posture of the IT infrastructure. From a forensics perspective: detailed collection of artefacts, a time-line of the events and a clear description of the components involved in the system compromise is expected.

Very often, the entry point for a security analyst or incident responder will be to be provided a PCAP of an actual incident and his investigation will start with this item. PCAPs provide a wealth of raw but actionable information such as:

- Timestamps
- Source and destination IP addresses
- Source and destination ports
- Protocols used
- DNS requests and name resolution information
- Application-level timestamps, headers and metadata
- Raw traffic

For a trained analyst, this level of information could be enough to perform a throughout analysis work but a great deal of triage can be done to reduce the manual workload, just by leveraging on open-source intelligence from VirusTotal, Hybrid-Analysis.com, Malwr.com and others, in order to make an informed decision.

Provided that the packet capture covers the complete time-range in which the infection cycle occurs, the expected outcome of the investigation should: confirm that the infection was successful, provide a time-line and Indicators of Compromise (IOC) as well as corrective procedures if possible.

In the case of an exploit-kit based incident, it will be needed to know:

- IP address and domain name of the compromised website (entry point)
- IP addresses and domain names of the exploit-kit gate and delivery servers (reconnaissance and exploitation)
- Type and hashes of the binaries delivered and their maliciousness

The idea of this work is that their mutual relationship is an interesting marker to look for and that being able to detect it reliably could speed up the analysis process. Enriching the collected dataset with information available on open-source channels will provide an extra level of contextual information.

The artefacts that are left on the system can be quickly checked against open-source databases like VirusTotal with a quick hash search and provide items for a further action plan.

Further levels of granularity can be easily added later by from the output of other analysis tools be that in static analysis, sandbox execution, passive DNS history, memory dumps… or any other source of information that is deemed relevant to the analyst.

2.3 Identifying the challenges of exploit-kit incident analysis

The redirection mechanism that leads to the browser compromise leaves a signature in the network traffic that we can translate in a specific pattern with the information we can extract from the collected PCAPs.

The HTTP protocol [10] provides us with a client-server model that can be easily modelled in terms of nodes and relationships and mapped back to the HTTP requests and responses that are going between the client and the server.

However, when strictly looking at the logs that are extracted from a PCAP, only one-to-one source-destination relationships can be extracted and it takes a tremendous amount of manual work to complete a full forensics process.

The forensics analysis process of a PCAP generally follows the diagram shown in Figure 3.

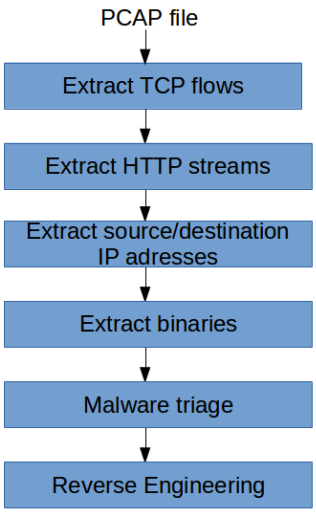


Figure 3: Network incident response workflow

Each of these steps depicted in Figure 3 will provide information that will further qualify the incident and its severity. The process of malware triage is critical in terms of incident response as its outcome can involve resources that are outside of a malware analysis scope and potentially involving critical business processes, emergency procedures, law enforcement agencies, etc.

Reverse engineering is a specific task that isn't always part of the incident response process as it is a largely manual task that requires time and expertise. In all the other steps of analysis, a great deal of work can be automated and fed into third-party APIs to provide further information on the content extracted from the PCAP.

This analysis process is geared towards the extraction and cross-referencing of information between two or more log files from different systems, which requires extra time, and can often be tedious without automation. In this work, it will be crucial to identify the redirections that the exploit-kit will send back to the client and link them back to a browser exploitation phase.

The specific problem of isolating the relationships between the different hosts and the files that they drop on the system applies to the case where an isolated incident is considered. From a broader angle, associating the hosts involved with higher level information will prove

to be a problem of aggregating data with different properties (domain names, whois registry information, country of origin...).

2.4 The case for visualization in cybersecurity

The research to bring visualization solutions in non-cyber defense systems has been a field of research for quite some time but that still seems to be developing slowly in the field of cybersecurity.

In 2009, D. Best, A. Endert *et al.,* in [11] identified 7 key challenges to bring visualization solutions further. Among them are scalability and elasticity issues that are being addressed today from various perspectives but that are outside of the scope of this research. They would nonetheless be critical aspects to take into account if our solution was to be transposed in a production environment. Two of the main challenges exposed were also the quality of the data and the possibility to link different sources together. This aspect will be of prime importance in the work that we are developing further in this research.

In their own field research for IBM from 2011, J. Rasmussen, K. Ehrlich *et al.,* in designing the NIMBLE project [12] found out along the same lines that a visual representation of IDS alerts in a visual form to a panel of trained cybersecurity analysts was not significantly improve their analysis performance but also acknowledged some interesting comments from their panel:

"In summary:

- Analysts appreciated the system's clustering of events and machines.
- "Strays" or anomalies lost in large data sets stand out in the diagrams.
- Some visual cues were considered insufficiently expressive, such as the use of subtle line thickness differences to represent event volume"

In that respect, adding a visual layer to our PCAP analysis should be specifically useful when analysing large amount of data because not only would we be able to extract raw data from the database itself but also be able to spot outstanding events such as clusters and outliers as the topology of the relationships they have one with the other.

In the specific case of volume analysis, attention is naturally brought to volume effects but a visual analysis of the graph can reveal structural weaknesses or outliers that can be exploited from a defensive perspective.

2.5 Current work in the field of merging cybersecurity and graph databases

The very nature of cybersecurity makes the use of graph databases specifically well-suited as in practice, cybersecurity requires to correlate informations from different sources into a common context and to establish the relationships that lead to an infection path or system compromise.

The MITRE Organization is working in that direction with the CyGraph project [13] that aims at "enhancing the security posture,maintaining situational awareness in the face of cyberattacks, and focusing on protection of mission-critical assets.". CyGraph is built on a REST infratructure and aggregates data related to the network infrastructure, security posture, cyber threats and mission dependencies. MITRE has built a variety of use-cases for CyGraph

that cover a large range of tasks: from evaluating attack graphs and network exposure to specific threats as well as mission impact analysis.

CyGraph is also based on Neo4j and provides a graphical overlay to the built-in Neo4j browser and leverages their efforts in developing STIX as a standardized language and format to share cyber threat intelligence.

The Japanese CERT (JP-CERT) has exposed during the FiRST conference in March 2016 [14] that they have been using a setup involving Neo4j, Django, vis.js and a bunch of python scripts to consolidate and automate the collection of IOCs when tracking the Winnti and Cloudy Omega APT campaigns that targeted Japanese organizations back in November 2014 according to Symantec in [15]. The motivations of JP-CERT to develop such tool were primarily based on the need to correlate IOCs from different incidents and to produce overall picture of the attacks.

At Troopers17, Marion Marschalek and Raphael Vinot have exposed a similar graph-based approach using Neo4j but in the field of static malware analysis [16] where a Neo4j back-end is used to map function calls, strings and other IOCs obtained from radare2 to estimate if a code is malicious or not. While their approach is definitely geared towards reverse engineering and the static analysis of malicious binaries, we found a similar mindset and direction in their work as for what we are undertaking here.

2.6 Problem statement and contribution

For the malware forensics analyst, the analysis of PCAPs at a purely technical level doesn't provide a full picture and understanding of an incident: it provides a significant amount on information that needs to be further analysed and enriched by other sources. The output of analysis tools like Wireshark only provide a linear view of incidents and the search for specific patterns or behaviours is left to the analyst. The full understanding of the incident is only possible by cross-referencing items and elements of a diverse nature and putting them back into the right context.

From a broader perspective, traditional relational SQL databases do not seem well-suited for the analysis of relationships and patterns between different entities. Their use often results in costly UNION statements that are often hard to write if at all possible given the heterogeneous nature of the data involved in that cross-analysis and there is a need for a better solution.

From a visual analysis perspective, there are few tools available "out of the box" that can provide the analyst with a relational view of a network traffic capture. Solutions are available from the industry but it is generally left to the analyst to build an overlay on top of the tools he or she is using for the investigation. This is specifically true when considering opensource solutions over commercial ones.

In short, the challenges that a malware or network forensics analyst will face when working on an incident will be:

- To automate the PCAP analysis process for a quick turn-around
- Enrich the data extracted from the PCAP with third-party information to provide more context
- Quickly sort out the outstanding elements of an incident and translate them into actionable IOCs.
- Consolidate the analysis data in a way that allows searching and referencing

After a thorough analysis of those challenges and evaluation of the technical solutions that are currently available, this contribution will be articulated around the use of Neo4j as a graph database to support the analysis tasks. Besides the installation, configuration and setup of Neo4j, it also provides:

- A way to breakdown the PCAPs in smaller work units (IP addresses, hostnames, decoys…) for further analysis.
- Demonstrate that we can leverage third-party REST APIs like VirusTotal to enrich our analysis and create a context around it.
- Assess the benefits of using a graph database as supporting back-end when looking at single incidents or multiple incidents of the same kind.
- Provide future directions in terms of how graph databases can be leveraged in a cybersecurity environment.

3 Experiment data-set and system design

3.1 Dataset used for our analysis

The network traffic captures that are used for this thesis work are freely available on www.malware-traffic-analysis.net and provide a wide variety of samples that are both known as malicious and that have been thoroughly analysed. Every sample acquired has a corresponding blog entry that details the delivery mechanism: compromised domain names and IP addresses, exploit-kit involved, malicious binary delivered…

PCAPs are acquired in bulk from the main page of the website and will need further triage: malware-traffic-analysis.net does not provide a search engine or any specific classification for the data-set that it is providing. The identification of the exploit-kits used and the malware they deliver is done by leveraging open-source analysis platforms such as VirusTotal and Hybrid-analysis.com and is complemented by the manual analysis of the author.

The PCAPs from the acquired dataset cover a period from 18th June 2013 to 15th March 2017 for a total of 524 usable PCAPs.

The filenames for the PCAPs that are used are not consistent although they seem to respect a general syntax which is as follows: ***<campign_name>-<exploit-kit_used>-<malware_delivered>.pcap*** and the archive shows that the captures are for the following campaigns:

- PseudoDarkLeech
- EITest
- AfraidGate

In a series of analyses, [17], [18] and [19], Brad Duncan from Palo Alto Networks has exposed all 3 campaigns. They all show specificities that our analysis method should show in greater detail.

PseudoDarkleech is described in [17] as a threat campaign targeting specifically Apache servers and WordPress instances and that is still active today. It has been known to deliver various types of malware including ransomware via all major types of exploit-kits available such as Angler, Netrino, Nuclear and Rig.

EITest as detailed in [18] is known to be based on the Rig exploit-kits and its variants. From his study, EITest seems to have now shifted from a gate delivery mechanism to a direct delivery via a weaponized exploit but its infrastructure should still rely on Rig.

AfraidGate is a relatively new campaign that has first been detected in February 2016 as pointed out in [19]. AfraidGate is known to use a gate system and also to have shifted from the Neutrino exploit-kit pack in February 2016 to NuclearEK in March 2016, to Angler since April 2016. AfraidGate is known to spread various types of ransomware (Locky, CryptXXX…) and banking trojans.

The number of attributed captures in the analysed dataset is shown in Table 1.

Table 1: Attributed exploit-kit PCAPs per campaign

Campaign	Exploit-kit	Number of PCAPs
Pseudo DarkLeech	Angler	12
	Neutrino	15
	RIG	41
ElTest	Angler	4
	Neutrino	6
	RIG	40
	Sundown	1
AfraidGate	Neutrino	5
	RIG	13

These exploit-kits will be studied later on a case-by-case basis and together as a campaign under the scope of a graph database.

A breakdown of the exploit-kits independently of the campaign they have been linked with a is shown in Figure 4.

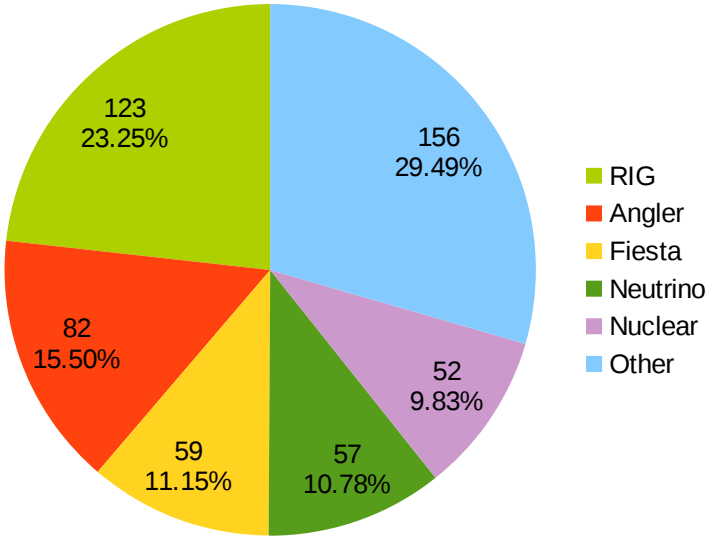


Figure 4: Composition of the PCAP dataset per exploit-kit type

3.2 System overview

Aside from having Bro [20] for the PCAP triage system, Neo4j will be used as a graph database for the backend database analysis. The system takes a PCAP file as an input and will process it to extract a log of all the TCP and UDP connections, an HTTP connection log, a file log keeping track of the associated TCP connection and a dump of all the extracted binaries.

A set of Python scripts will then check the logs and transpose them in the Cypher syntax that is used in Neo4j. VirusTotal is also used as a third-party API for quick triage and all the files decoyed from the PCAP are hash checked against the VirusTotal database. Any system based on a REST API can be used to enrich the analysis data in the same way.

The overview of the system designed for this work is depicted in Figure 5.

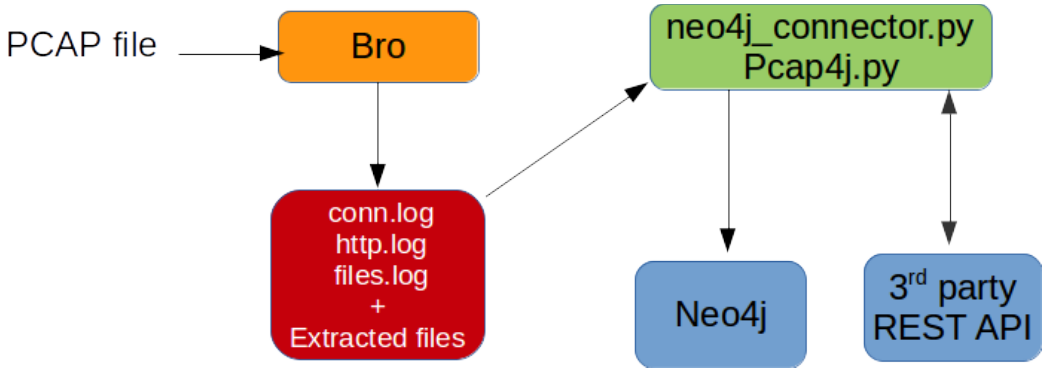


Figure 5: System overview

3.3 Feature isolation in incident detection

From the description shown in Figure 1 on how exploit-kit based incidents happen, the exploit-kit gate should be the point of focus of our analysis as it connects both to the compromised website (source of infection) and to the exploit-delivery infrastructure although in some cases the exploit and payload can be delivered on the host where the gate function is processed.

The features that our analysis will be focus on are: HTTP Requests, HTTP Response with a specific classifier for 302/301 responses and REDIRECTs, as well as the binaries that are dropped on the system.

The pcap4j.py script will launch an instance of bro with the command:

```
bro -r /tmp/" + PCAP_FILE + " -C frameworks/files/extract-all-files frameworks/files/hash-all-files.bro protocols/http/http-header-logs.bro.
```

This command will generate the log files http.log, files.log and conn.log that will be further carved out by our script.

The script that parses the Bro logs (pcap4j.py) and its functions are explained in Table 2. Its general purpose is to feed a PCAP into Bro and extract from the logs the IP addresses and hostnames involved in HTTP transactions. The script will also prepare 3 lists that will be used to create the graph based on the HTTP requests, responses and redirections from the hosts that have been previously extracted. HTTP requests are responses are automatically matched by Bro and are part of the same line of logs. The extraction of server headers is needed when a 301 or 302 response code is seen and this option must be specifically set in the HTTP protocol extraction options of Bro.

The output of the pcap4j.py script is later being used by the script neo4jconnector.py which specifically looks at creating the nodes and relationships in the Neo4j database backend.

Table 2: Functions description for the log carving script (pcap4j.py)

Function name	Description
bro_prep(PCAP_FILE)	Launches bro to extract the PCAP file provided as an argument and relocates the output to /tmp.
host_list()	Extracts unique IP addresses from conn.log.new and evaluates if they are public or private.
add_hostnames(host_list)	From a given list of hosts, extract the hostname from the http.log for a matching IP address.
http_tracker()	From the http.log.new file, split up the log in 3 lists of dictionaries: for http requests, http responses and redirections.
decoy_tracker()	From the files extracted by bro in /tmp/extracted_files/ calculate the SHA1 hash of the decoy, reattribute its source IP and filters out the application file types only.
vt_check_files(decoy_list)	From the list of decoys provided by decoy_tracker(), check VirusTotal if the SHA1 hash of the decoy file is known as malicious.

3.4 Data-model for the graph database backend

Using a graph database presupposes that we structure a data-model that will describe the different entities (nodes) taking part in the generated graph, the relationships (edges) that nodes will have one to the other and the properties that both nodes and edges can have.

In this case, the analysis seeks to understand the relationships that the hosts found in the PCAP have with one another at different levels:

- At the protocol level: the system will map the HTTP requests and responses seen in the PCAP analysis. The hosts will then have relationships with one another based on the characteristics of the HTTP protocol.
- At the network level: the system will map the relationships that these hosts have with the Autonomous System (AS-BGP) where their IP address belongs. By attributing an AS number with a host that is involved either as an exploit-kit or as an exploit delivery server, the analysis seeks to highlight what is the underlying infrastructure of the campaigns.
- At the country level: the system will find out where is physically located this infrastructure. The physical location of systems that are engaged in malicious activity is an important information when relating back law enforcement partners.
- From an infection cycle perspective: the system will able to extract the binaries and other file artefacts from the PCAP, and these artefacts will be attributed to the host dropping them.
- The system will finally leverage open-source threat intelligence to estimate the maliciousness of the artefacts dropped on the system during the finger-printing or infection phases.

The Table 3 sums up the different nodes that the system will create for the graph analysis and the relationships they have with one another.

Table 3: Inter-nodes relationships of the data-model

	Host	File	Payload	AS	Country
Host	HTTP REQUEST, HTTP RESPONSE REDIRECT REFER	DROPS	-	BELONGS_TO	-
File	DROPS	-	IS_A	-	-
Payload	-	-	-	-	-
AS	-	-	-	-	LOCATED_IN
Country	-	-	-	-	-

From a visual perspective, this given set of node-relationship dependencies is depicted in Figure 6.

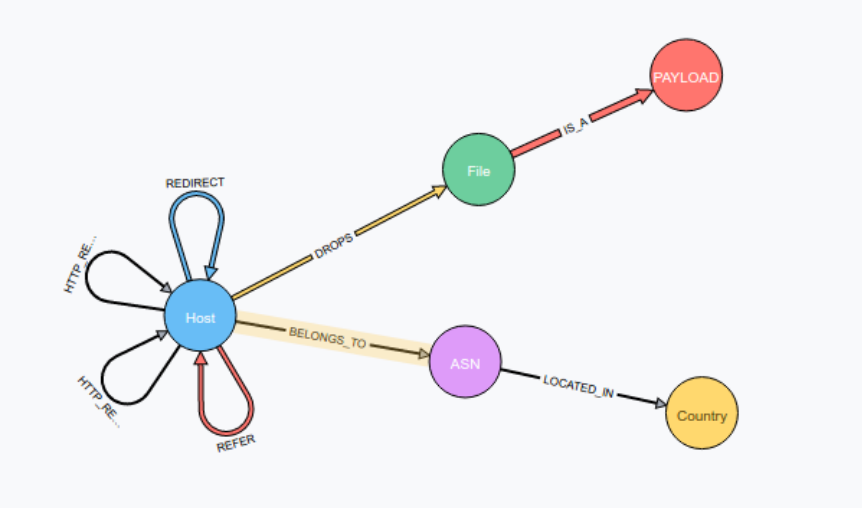


Figure 6: Graphical representation of the data-model

When creating a data-model for a graph database, not only should the relationships between the different nodes be defined, the nodes that will be part of the graph should also be allocated properties. The Table 4 shows the properties that been chosen in the first place to defined our nodes. Those properties can later be extended for specific node types which allows to have first a simple model that can later be made more complex if additional properties can be gathered from external sources or further internal analysis.

Table 4: Node properties of the data-model

Node type	Properties
Host	IP address, type of IP address (public or private), hostname (if available)
File	File type, filename (as extracted by Bro), SHA-1 hash value
Payload	Signature
ASN	Number
Country	Name

The neo4j_connector.py script is primarily used as an interfaces with the Neo4j database and adding nodes and edges as well as updating node properties is done from this script.

Table 5: Functions description for the database population script (neo4jconnector.py)

Function	Description
add_nodes(node_list)	Creates nodes from a list of hosts returned in host_list().
add_hostnames(hostnames_list)	Matches a hostname based on an IP address of a node and add that property.
build_http_graph(requests_list,response_list,refer_list)	Build the HTTP_REQUEST, HTTP_RESPONSE and REFERER edges based on the http.log.new log file.
add_downloaded_files(file_list)	Add files that have been decoyed in /tmp/extracted_files/.
add_malicious_file_markers(checklist)	Adds a PAYLOAD node and a relation with the file of corresponding SHA1 identified Malicious on VirusTotal.
add_asn_owner(host_list)	Performs a WHOIS lookup for each IP address supplied in the host_list and returns the AS Owner, Handle, Country of Origin.

The build_http_graph function takes a specific care at creating only one instance of the HTTP_REQUEST or HTTP_RESPONSE relationship between two hosts via the MERGE query in order not to overcrowd our graph: working at the individual request-response level provides a too high granularity for our analysis and does not yield added value and the representation of a single instance is preferred here. On the other hand, a specific care is taken to extract all the 301 and 302 Moved responses and show their individual redirection paths. This is done for each of them with a CREATE query rather than a MERGE query in this case. The creation of REFER relationships is directly obtained for the parsing of the client request headers seen in the Bro logs.

A complete developer documentation of the Neo4j database is available at [21].

4 Incidents, infrastructure and campaign analysis

4.1 Introduction to Cypher Query Language and single incident analysis

Cypher is the query language that comes with Neo4j. The queries emulate an ASCII-art style of syntax as nodes are represented into round brackets (), edges are built with – and relationships are in square brackets [].

As such, the query `MATCH (n) RETURN n` will then return all the nodes in the graph.

A relation between two nodes in Cypher is then written *(a)-[r]-(b)* in it simplest form.

The following example is a PCAP taken from the dataset 2015-12-18-Rig-EK-traffic.pcap – f447f0ba11a54e5e62d7ddfdd76d7fd2dad8541d.

Once acquired, the graph for the PCAP can be viewed via the integrated Neo4j browser with the Cypher query: `MATCH p=()--() RETURN p`, which will return all the nodes of any type that have a relation of any type with another node of any type and is shown in Figure 7.

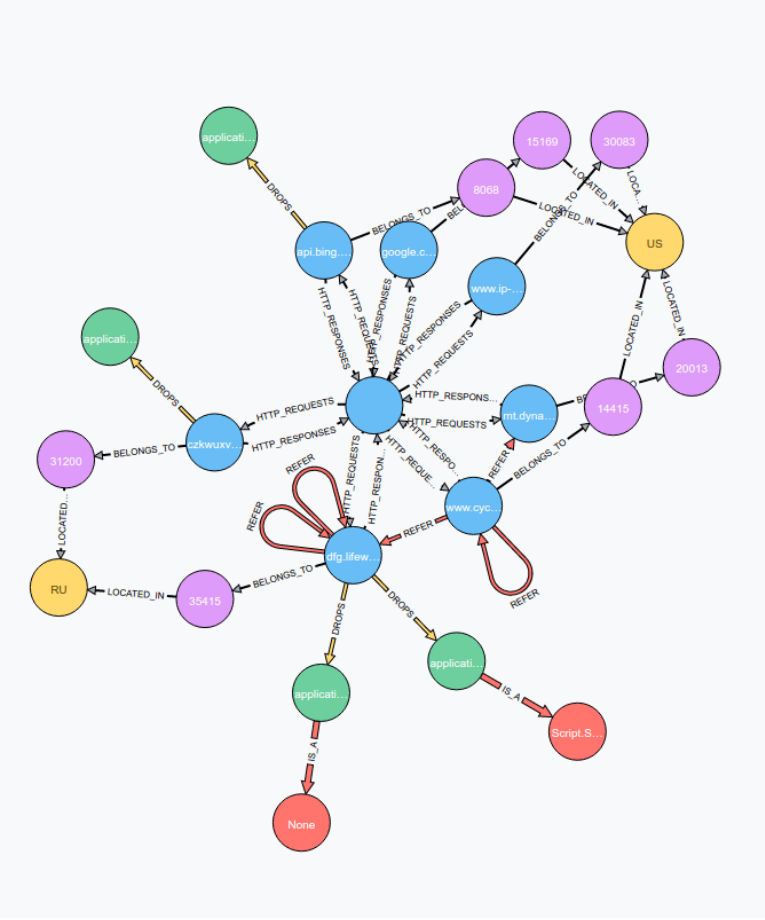


Figure 7: Graph of a Rig Exploit-kit incident

This query is only relevant to analyse single events as it gives the highest possible level of granularity and should be avoided on large sets of unfiltered data.

A quick visual analysis shows the client in the middle of the cluster and the various HTTP requests that have been made from it and their subsequent responses.

Finding out the pattern for an exploit-kit behaviour which is manifested by a REFER to either itself for direct payload delivery or to another server is done with the query ***MATCH p=(n:Host)-[r:REFER]-() RETURN p*** and returns the graph shown in Figure 8.

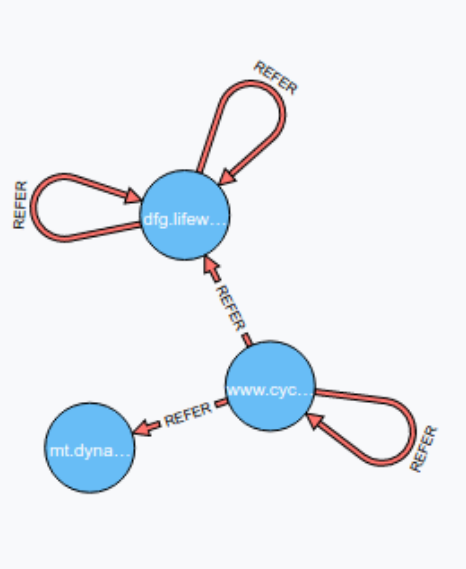


Figure 8: Exploit-kit browser fingerprinting mechanism graph

The list of nodes taking part in this graph is returned with the query ***MATCH p=(n:Host)-[r:REFER]-() RETURN DISTINCT n*** and provides the following output when copy-pasted from the browser:


```
{"hostname":"www.cyclocamping.com","address":"96.31.44.126","type":"PUBLIC","ek_type":"Rig"}
{"hostname":"dfg.lifewavedenmark.com","address":"46.30.43.213","type":"PUBLIC","ek_type":"Rig"}
{"hostname":"mt.dynamicwords.us","address":"192.185.21.183","type":"PUBLIC","ek_type":"Rig"}
```


This analysis can be further expanded to include all the hosts that are showing such behaviour and from which malicious files have been downloaded with the query: ***MATCH (n:Host)-[r:REFER]-() MATCH p=(n)--(m:File)--(o:PAYLOAD) RETURN p*** and returns the graph shown in Figure 9.

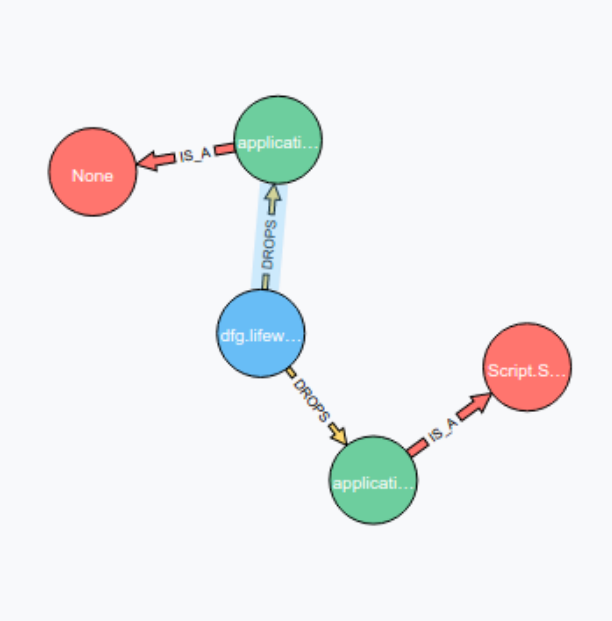


Figure 9: Graph extract of an incident involving the delivery of multiple exploits

The list of hosts, files and identified payloads is returned with the following query: ***MATCH (n:Host)-[r:REFER]-() MATCH p=(n)--(m:File)--(o:PAYLOAD) RETURN DISTINCT n*** which will return us the node that shows an exploit-kit behaviour with a known payload drop:

{"hostname":"dfg.lifewavedenmark.com","address":"46.30.43.213","type":"PUBLIC","ek_type":"Rig"}

The list of files that have been dropped and identified via VirusTotal as known malicious payloads is returned via the query: ***MATCH (n:Host)-[r:REFER]-() MATCH p=(n)--(m:File)--(o:PAYLOAD) RETURN DISTINCT m*** and this information can be directly reused for further analysis:

{"sha1":"217830fc4a7392d7ab5bd7c8195086676df832f7","filename":"extract-1450459886.035647-HTTP-FJIp2o4Uvg9GE4zXC9","type":"application/x-shockwave-flash"}
{"sha1":"d4ac740da4294f05d2a7496739bbdd3fe56a20bc","filename":"extract-1450459891.662535-HTTP-FRMENv2CnotXi85pe8","type":"application/octet-stream"}

When looking at the files that have been dropped and identified as malicious payloads, the query: ***MATCH (n:Host)-[r:REFER]-() MATCH p=(n)--(m:File)--(o:PAYLOAD) RETURN DISTINCT o*** returns us the signatures that have been hit from F-Secure, AhnLabs and G-Data after a hash search on VirusTotal or if any positives at all are yield in this way:

{"ahnlabs":"SWF/Exploit","positives":"30","gdata":"Script.SWF.C289","fsecure":"Script.SWF.C289"}
{"ahnlabs":"None","positives":"1","fsecure":"None","gdata":"None"}

From a visual perspective, the analyst will be focusing its attention on the artefacts that are directly linked with a known infection vectors shown here above.

A first set of incident response measures from this visual analysis and database queries will be to isolate the client that is involved in this PCAP (192.168.122.213) and also to block access to the hostname dfg.lifewavedenmark.com, IP address 46.30.43.213 and finally blacklist or seek further analyses for the files with the hashes 217830fc4a7392d7ab5bd7c8195086676df832f7 and d4ac740da4294f05d2a7496739bbdd3fe56a20bc.

However, this analysis also shows that not all the files that have been decoyed have been marked as malicious. This might be either because they are genuinely non-malicious files or because VirusTotal doesn't provide a hit for an identified threat when searching for its SHA-1 signature.

To have another look at all the files decoyed on the system and where they were served from can be done with the following query: ***MATCH p=()--(m:File) OPTIONAL MATCH (m)--(o:PAYLOAD) RETURN p*** and the graph retuned shown in Figure 10 points at additional sources where files have been received from.

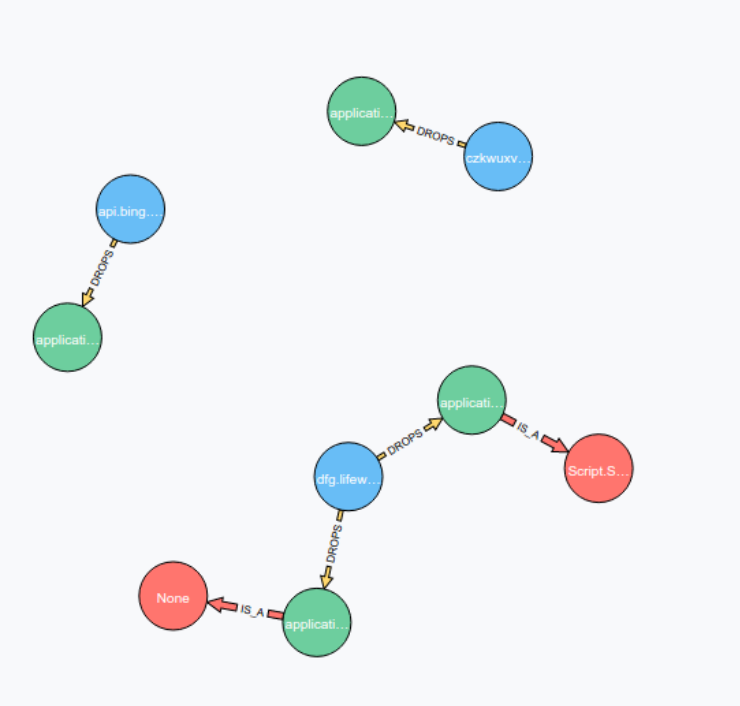


Figure 10: Graph extract of an incident showing both malicious and non-malicious decoys

Querying for all the hosts that shown in Figure 10 with: ***MATCH p=(n:Host)--(m:File) OPTIONAL MATCH (m)--(o:PAYLOAD) RETURN DISTINCT n*** will return the following domain names that can be investigated with further attention.


```
{"hostname":"dfg.lifewavedenmark.com","address":"46.30.43.213","type":"PUBLIC","ek_type":"Rig"}
```

```
{"hostname":"czkwuxvndxrjsprm.org","address":"37.194.118.106","type":"PUBLIC","ek_type":""}
```

```
{"hostname":"api.bing.com","address":"13.107.5.80","type":"PUBLIC","ek_type":""}
```


The second hostname returned looks suspicious and should be further investigated.

4.2 Visual analysis clustering

The visual analysis of this incident also provides an interesting visual clue as to what nodes are directly linked to the infection of the client and what nodes can be discarded from the analysis effort. The nodes of specific interest are depicted in the cluster 1 shown in Figure 11.

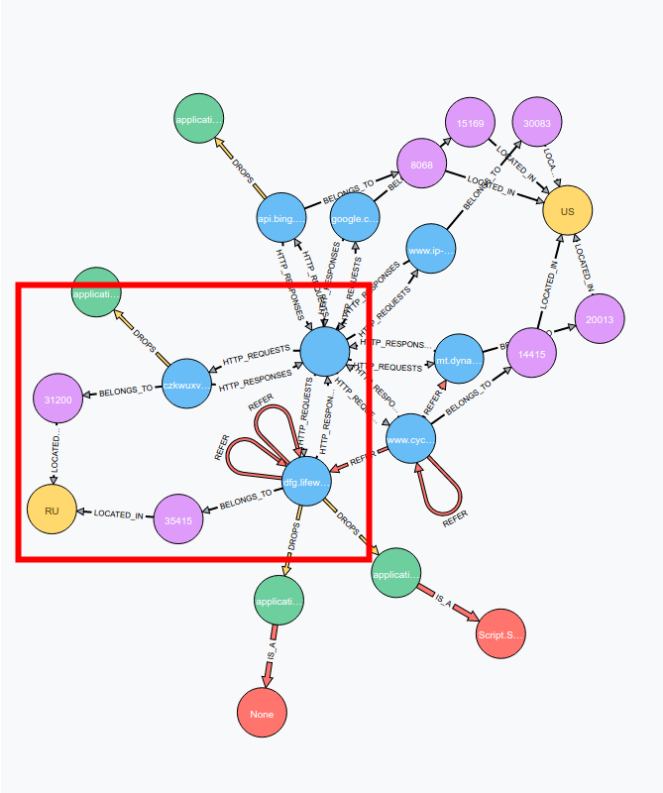


Figure 11: Visual clustering on a Rig Exploit-kit incident

This cluster shows:

- A host that is known to show an exploit-kit behavior with multiple REFERs to itself.
- This same host drops two files that are identified as malicious.
- A non-malicious binary dropped from a host with a suspicious domain name.
- Both hosts belonging to Autonomous Systems that are located in the same country

This visual analysis presents a set of information that is easily identifiable and reusable in a different security contexts; from local incident response to law enforcement.

4.3 Topographical comparison with Angler and Fiesta exploit-kits

As a comparison, the analysis of an incident that involves the Angler exploit-kit and that is also taken from the dataset: 2015-12-03-Angler-EK-delivers-CryptoWall-traffic.pcap – 207523e183bd818f7d682dfdf38427d4c081be0f is performed.

The visual analysis of this PCAP yields the graph shown in Figure 12.

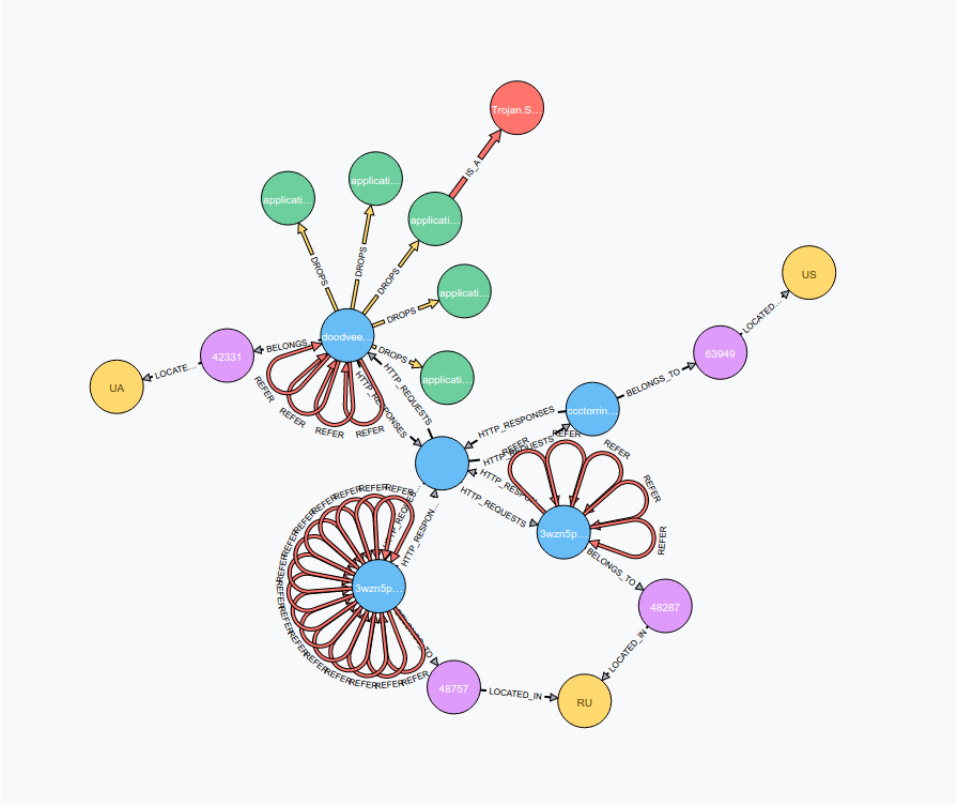


Figure 12: Graph of an Angler Exploit-kit incident

Figure 12 shows that exploit-kit behaviours are seen on 3 different hosts with one of them known to drop a malicious binary.

The hosts that are showing an exploit-kit behaviour can be isolated with the following query: ***MATCH p=(n:Host)-[r:REFER]-() RETURN p*** and shown on Figure 13.

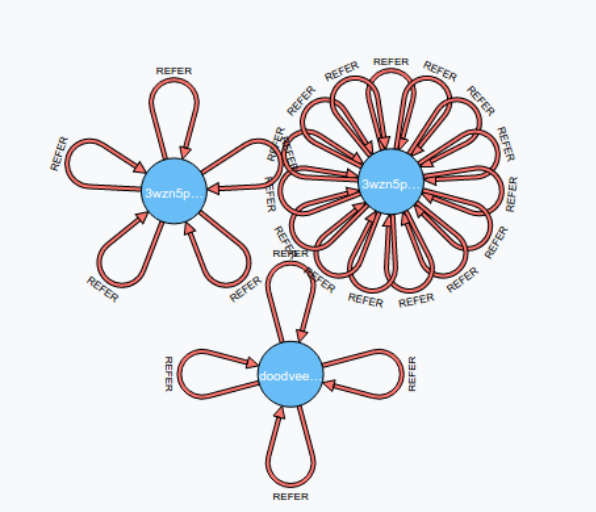


Figure 13: REFERs observed in an Angler exploit-kit incident

The analysis can be narrowed down to the hosts that are showing an exploit-kit behaviour and dropping a known malicious binary with `MATCH p=(n:Host)--(m:File) OPTIONAL MATCH p1=(m)--(o:PAYLOAD) RETURN p,p1` and the resulting graph is depicted on Figure 14.

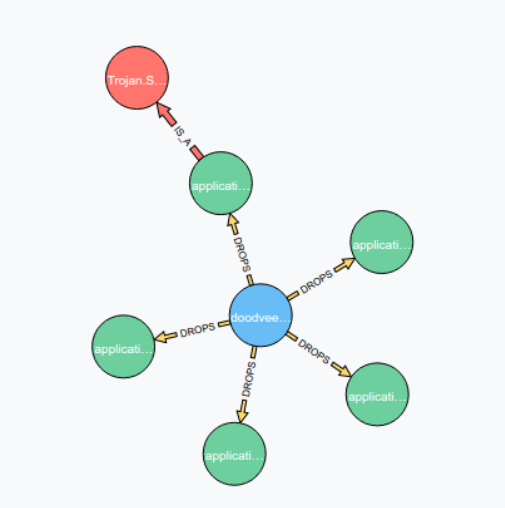


Figure 14: Exploit delivery in an Angler exploit-kit incident

At this stage, further investigations can be made on the other binaries that are dropped from the host doodveeantoviar.kyderby.org at the IP address 178.20.159.71.

While this analysis shows that the initial compromised website is ccctorrington.com, further work needs to be done to understand completely the infection vector.

With further analyses on other PCAPs where Angler has been identified, it appears that the Angler exploit-kit often presents this sequence of 4 consecutive REFER with which the client's web browser gets fingerprinted.

As a last example, an incident involving the Fiesta exploit-kit with the same visual method with the PCAP 2015-04-02-Fiesta-EK-traffic.pcap – 4c8dc04e901ed681616bb7ac895e-b6b95f5d96b6 returns the graph shown in Figure 15. While being an older exploit-kit that was nearly phased out around 2015, it can not be ruled out that this exploit-kit is not being re-purposed or redeveloped, and studying legacy traces can also give a feel for what to expect with newer versions of it.

This time again, as shown on Figure 15, the exploit-kit gate can be identified as the host showing multiple REFERs that are used to fingerprint the client browser and with associated binary artefacts being dropped subsequently.

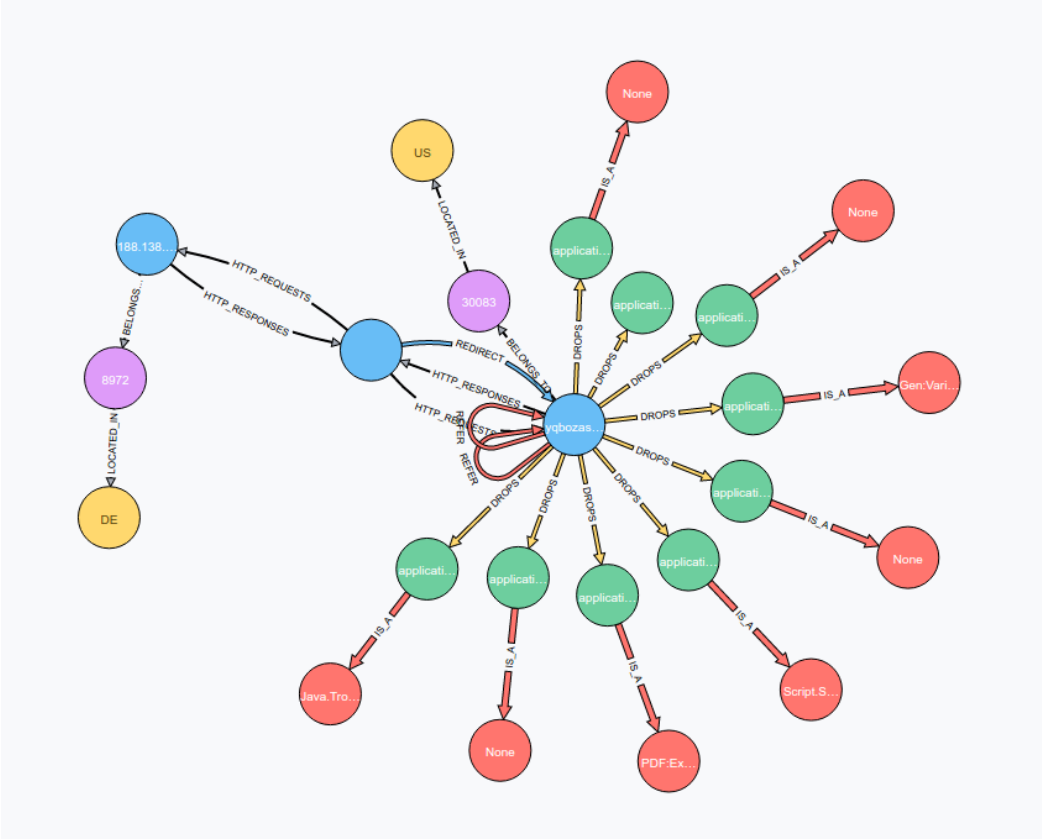


Figure 15: Graph of a Fiesta exploit-kit incident

In the specific case of Fiesta, a large number of detection for various kinds of exploits such as Java exploits, Flash exploits, PDF exploits and various kinds of payloads, is taking place. It seems that the Fiesta exploit-kit is trying blindly to infect our client by throwing at it all the possible exploits it has been loaded rather than targeting one specific type of vulnerability as Angler and Rig do with Flash.

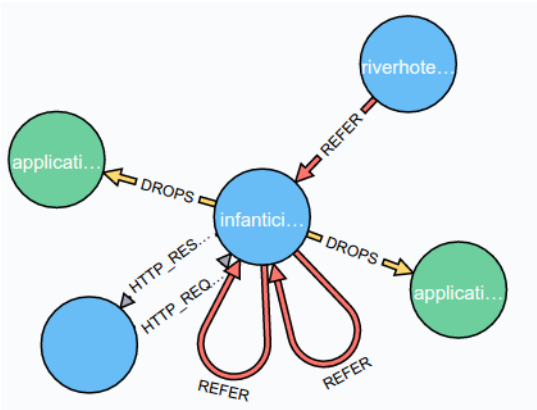
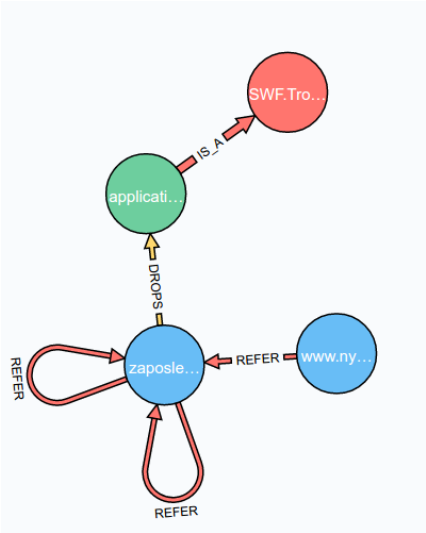
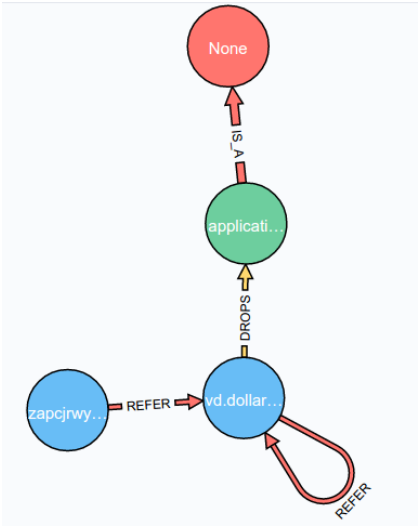
4.4 Generic search pattern for exploit-kit detection

After analyzing those separate incidents, a generic detection pattern can be articulated around the following description "find all the complete paths where a public host refers to itself or another public host that drops a binary". This basically translates into the following query: ***MATCH (n:Host)--(h:Host)-[t:DROPS]-(f:file) WHERE n.type = 'PUBLIC'***. This query can later be extended to include the files that have been matched against a VirusTotal hash search with ***MATCH (n:Host)--(h:Host)-[t:DROPS]-[f:file]--(o:PAYLOAD) WHERE n.type = 'PUBLIC'***.

Those queries are fairly generic and while they will filter out the legitimate traffic out from the potentially malicious events, they will also return a number of results that might be too large to be truly useful from a visual perspective. The WHERE clause can be used to narrow down the results to a set of matching properties such has the IP address, hostnames, domain names, file type or signature name.

The generic detection patterns for exploit-kit detection are shown in Table 6.

Table 6: Generic detection queries, patterns and associated severity

Query	Visualization example (isolated incident)	Severity
`MATCH p=(n:Host)--(h:Host)-[t:DROPS]-(f:file) WHERE n.type = 'PUBLIC' RETURN p`		**Suspicious.** Further investigation is needed. The path shown is similar to an exploit-kit infection but the dropped file is either not present in VirusTotal or has 0 hits on a hash search.
`MATCH p=(n:Host)--(h:Host)-[t:DROPS]-(f:file)--(o:PAYLOAD) WHERE n.type = 'PUBLIC' RETURN p`		**Malicious.** The dropped payload is referenced on VirusTotal and has a signature from the F-Secure corpus.
`MATCH p=(n:Host)--(h:Host)-[t:DROPS]-(f:file)--(o:PAYLOAD) WHERE n.type = 'PUBLIC' AND o.positives <10 RETURN p`		**Highly suspicious.** The dropped payload has a low detection rate on VirusTotal (less than 10 engines) and it might or not have a signature in the F-Secure corpus.

4.5 Analysis against non-malicious traffic and potential for false positives

At this stage, it can be assumed that a host showing multiple REFERs to itself and dropping one or more binaries is probably an exploit-kit. This pattern is present in all the cases that were discussed so far but, as often in cybersecurity, threats emerge from the malicious use of legitimate features and this detection method should also be tested against non-malicious traffic in order to see if there is a potential for false-positives.

In order to test the validity of the created model, different harmless websites (Facebook, Twitter, Netvibes, SoundCloud, SoFoot and YouTube) are visited and briefly browsed.

A general overview of the graph created for this traffic capture with the query `MATCH p=(n:Host)--()` returns the following graph shown in Figure 16.

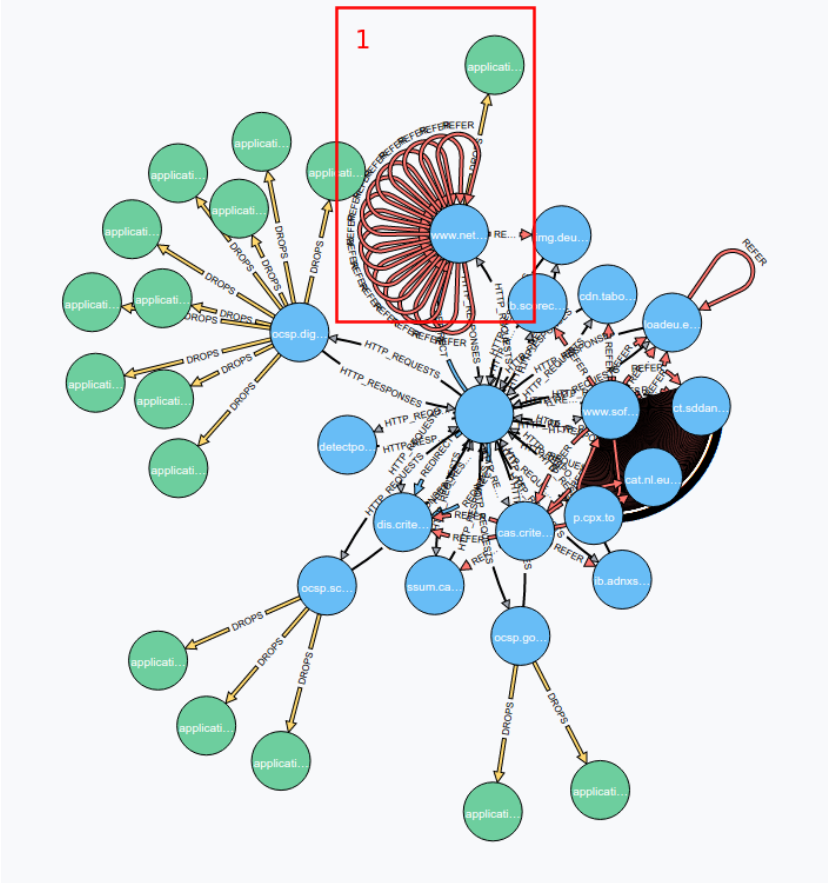


Figure 16: Graph of a casual browsing on non-malicious websites

Most of the files dropped are coming from OCSP sub-domains that provide responses known as OCSP stapling, prior to the establishment of an HTTPS connection to a CDN service as described in [22] by J. Liang, J. Jiang, *et al*. These OCSP responses should be filtered out as they ultimately clutter the analysis with artifacts that are known to be benign. In the cluster 1 shown in Figure 16, the host www.netvibes.com shows multiple REFERs and finally drops a binary. This behaviour is identical to those seen when analysing traffic captures that have shown an exploit-kit infection. A further manual analysis of the decoyed file reveals that we have a Web Open Fonts Format (WOFF) file and there are known occurrences of this file format being

format being exploited such as in CVE-2010-1028 [23] and CVE-2014-9668 [24].

This case is typically showing the type of situation that needs further analysis: on the one hand, the domain netvibes.com can be considered trusted and the hash search on VirusTotal did not return a positive result; on the other the type of file that is decoyed has already been involved in a couple of CVEs and although these are quite old CVEs, there still is a chance that the vulnerabilities could reoccur or that the systems are not patched against those for some reason. There is, in this case, sufficient trust in the netvibes.com domain name but this appreciation would be totally left to the analyst for cases that are not so obvious.

In order to minimize the chance for a false positive, the analysis must involve additional data: a hash search on VirusTotal is only a crude tool that has obvious limitations. The hash searched might be simply missing because it has never been submitted for full analysis or because we have a new variant of an existing malware. Extending the tool to resubmit for full analysis these kind of decoy would easily help us confirm if the file is malicious or not. An extension of the data-model that would include artifacts obtained from the use of static and dynamic analysis tools such as PE header structure, modified Windows registry keys, mutexes created, etc, would expose additional elements that can be seen as clear indicators of compromise (IOCs).

4.6 Actionable IOCs after single incident analysis

When analysing single instances of incidents, the analyst is able to isolate the set of Indicators of Compromise that is shown in Table 7 and to respond accordingly with a possible range of actions.

Table 7: Actionable Indicator of Compromise and possible responses

Indicator of compromise (IOC)	Response
Internal host IP address.	Host quarantine. Forensics examination.
External host IP addresses.	IP address blacklisting (per address or per block).Internal systems log analysis (to determine if multiple hosts could have been compromised).
External host domain names.	Domain and sub-domain blacklisting. Internal systems log analysis (to determine if multiple hosts could have been compromised).
Dropped files.	Resubmission for full engine analysis on external services (VirusTotal). Static analysis. Dynamic sandbox analysis. Reverse Engineering.
Dropped files names and hashes.	Further intelligence gathering from external services (VirusTotal, Google Search, Hybrid-Analysis.com …). Manual anti-virus scanning on internal hosts.

Forensics, incident response and defensive processes are specific specific to every organization and it is beyond the scope of this work to discuss them. It is possible that the IOCs presented here are not enough to justify the undertaking of those processes but they form a basis upon which further decision can be taken.

5 Infrastructure analysis

In this part are analysed the 3 main exploit-kits that have been used during the Pseudo DarkLeech campaign with the intent to understand if they share a common infrastructure or expose structural weaknesses that can be exploited from a defensive perspective at an ISP level for example.

5.1 Angler EK infrastructure analysis

In this case, the analysis is done on the 83 PCAPs from our dataset that are known to have an Angler-related incident and our aim will be to study the underlying infrastructure of this exploit-kit.

Neo4j returns 677 infection paths and we can display them with `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` and shown in Figure 17 where 4 clusters can be isoalted.

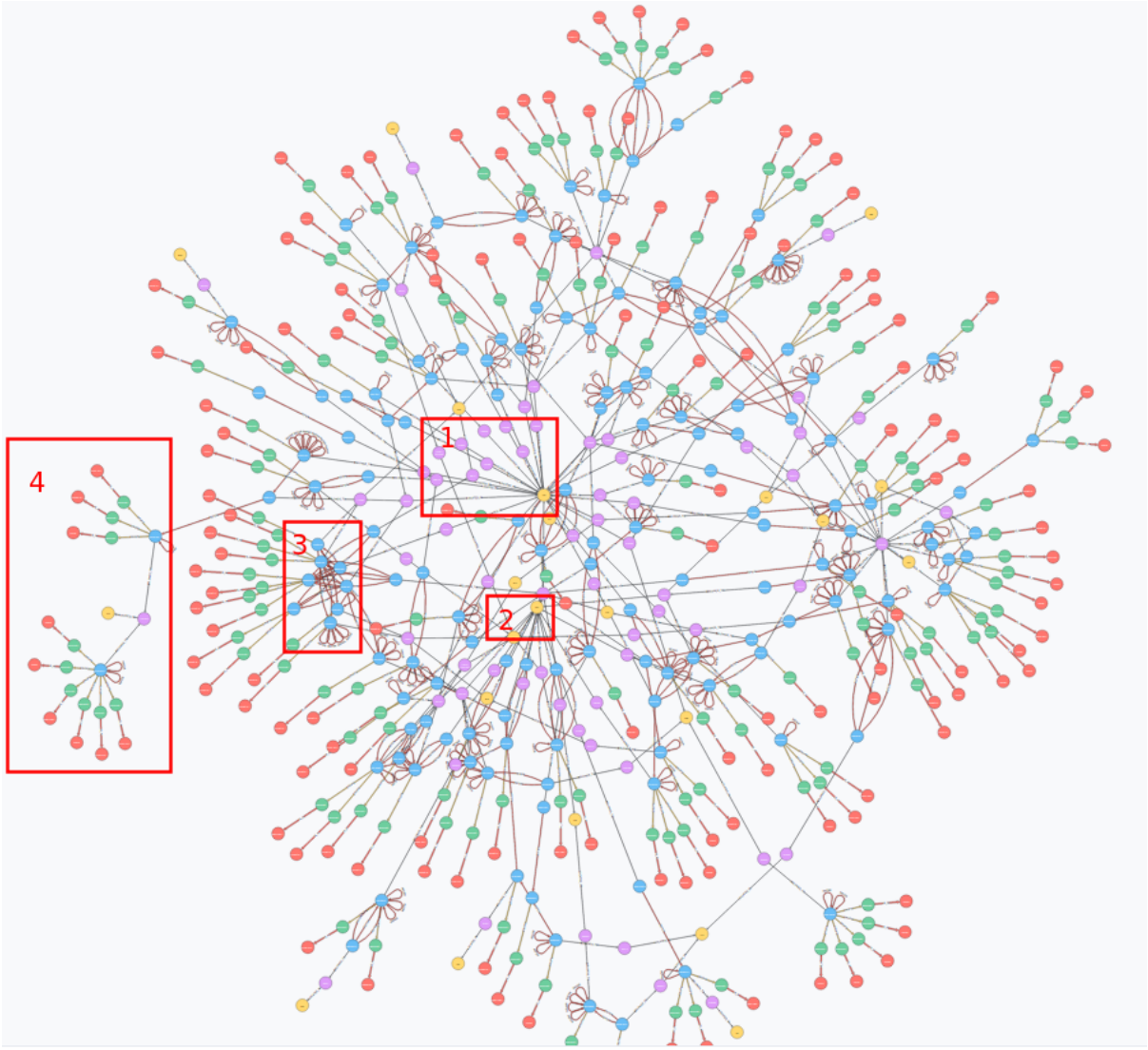
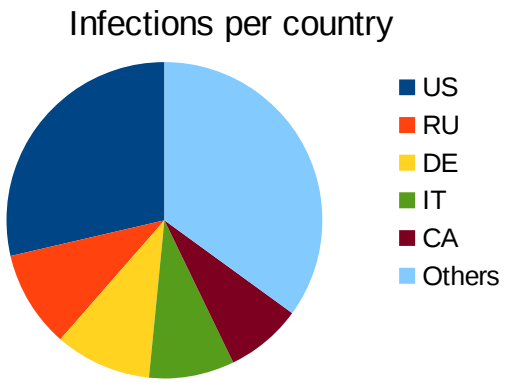


Figure 17: Clusters observed in the analysis of the Angler exploit-kit infrastructure

Cluster 1 shows a part of the infrastructure that is centred on the US. Cluster 2 is showing the part of the infrastructure that is based in Russia. Cluster 3 shows hosts that are inter-related through their malware delivery branches. Cluster 4 is showing that Thailand specifically stands out as an infrastructure used for malware delivery platform.

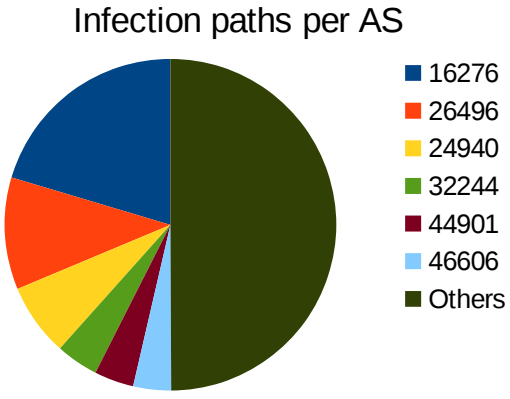
The number of infection paths per country is returned with: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT c.name, count(c) ORDER BY count(c) DESC*** and get the corresponding statistics shown in Figure 18.



Country	Infection Paths
US	194
RU	67
DE	67
IT	59
CA	53
Others	237

Figure 18: Angler exploit-kit infrastructure statistics per country of origin

A similar search but looking at the number of Infection paths per AS with: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths ORDER BY count(a) DESC*** will return the statistics that are shown in Figure 19.



ASN	InfectionPaths
16276	138
26496	74
24940	48
32244	28
44901	26
46606	25
Others	338

Figure 19: Angler exploit-kit infrastructure statistics par AS of origin

A specific focus on the AS16276 that is responsible for the majority of the Infection Paths from a visual perspective is shown on the graph of Figure 20 via the query:

```
MATCH p=(c:Country)-[]-(a:ASN{number:'16276'})-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p, c.name as Country, h.address as Hostname, f.sha1 as File
```

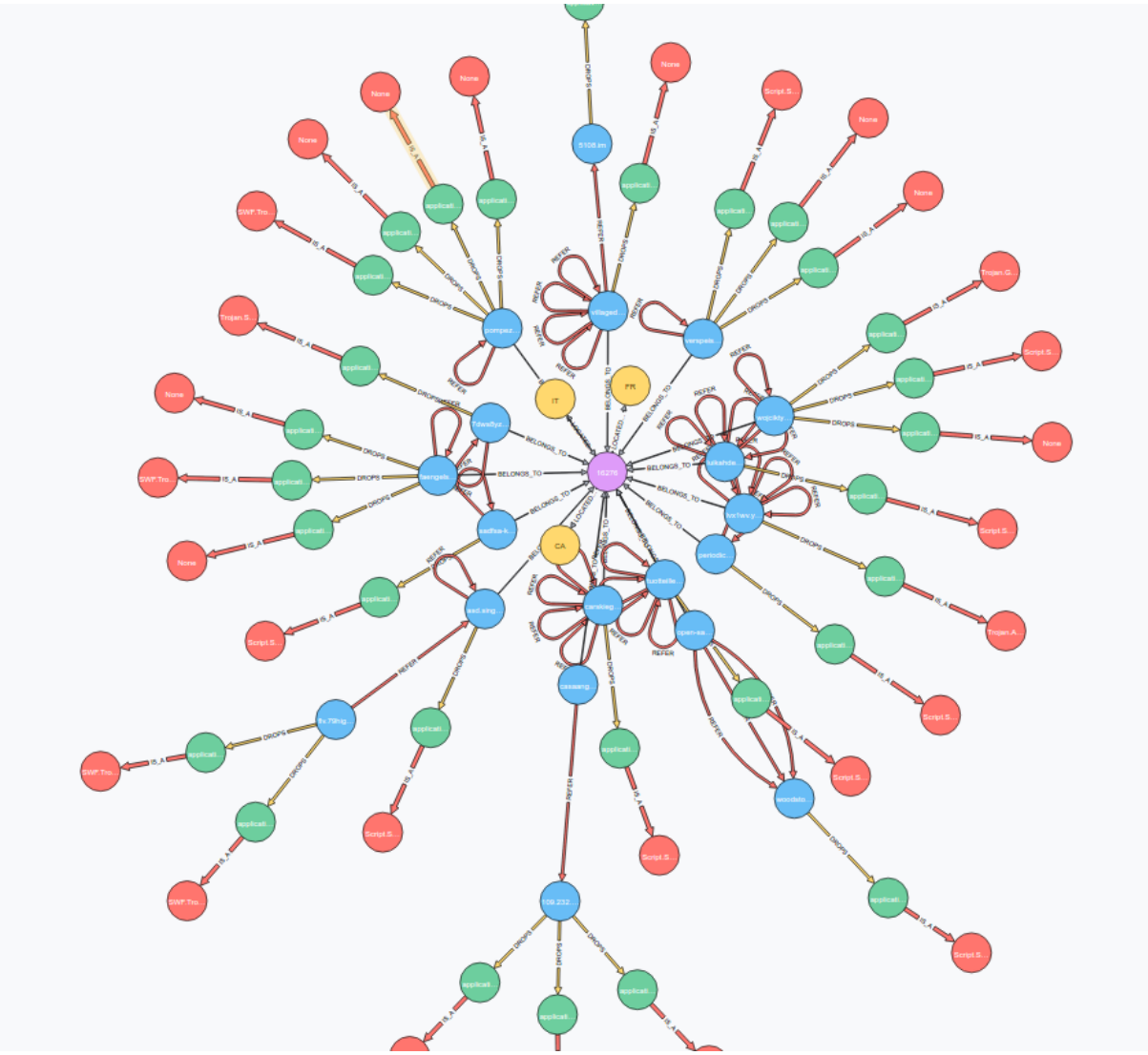


Figure 20: Focus on AS16276 infection paths in the Angler Exploit-kit infrastructure

The AS16276 supports the infrastructure of malicious payload delivery based in France, Italy and Canada and belongs to OVH which is a major hosting provider based in France and active on the European market. This information is valuable in the sense that it points back to a root-cause at least in terms of system ownership.

5.1.1 US-based infrastructure

A specific look at the infection paths that are occurring through a US-based AS is shown in Figure 21 via the query `MATCH p=(c:Country {name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p, a.number as ASN, h.address as Hostname, f.sha1 as File`

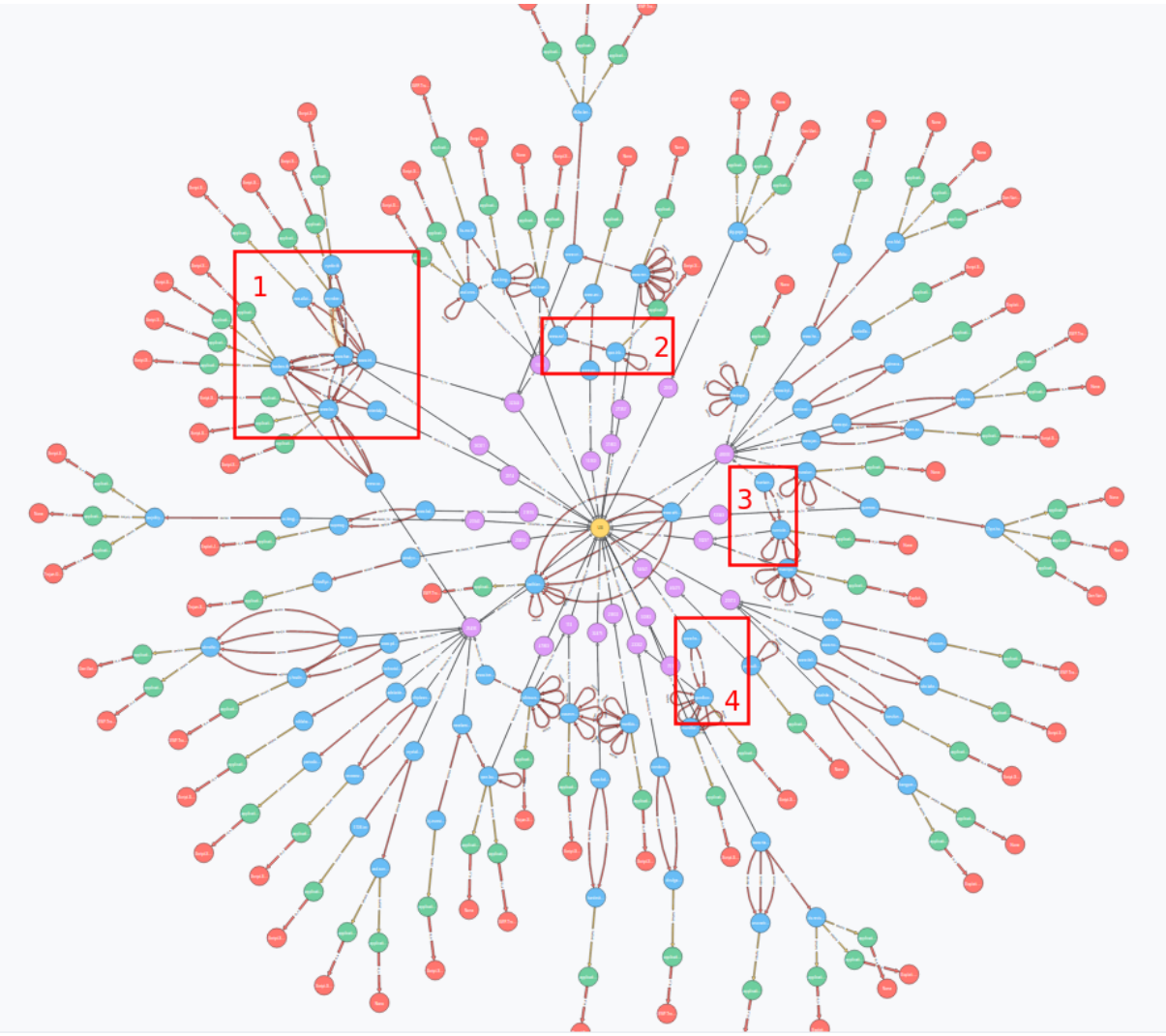
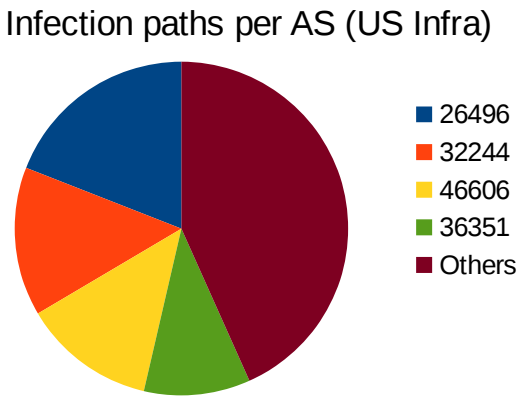


Figure 21: Clusters observed in the US-based infrastructure of the Angler Exploit-kit

The Figure 21 exposes 4 separate clusters where it is clear that the malware delivery platform is shared across different AS numbers.

The query looking for Infection Paths from the US: ***MATCH p=(c:Country{name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]-(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths ORDER BY count(a) DESC***, returns the statistics shown in Figure 22.

The spread between the different AS numbers does not really show a predominant vector of infection. A set of 4 AS numbers is responsible for the majority of the infection paths detected but they are also showing distributions that are fairly close one to another.



ASN	InfectionPaths
26496	37
32244	28
46606	25
36351	20
Others	84

Figure 22: Infection Path statistics for the US-based infrastructure of Angler

5.1.2 Russian infrastructure

A similar visual analysis on the Russian infrastructure of Angler with: ***MATCH p=(c:Country {name:'RU'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p, a.number as ASN, h.address as Hostname, f.sha1 as File*** returns the graph shown in Figure 23.

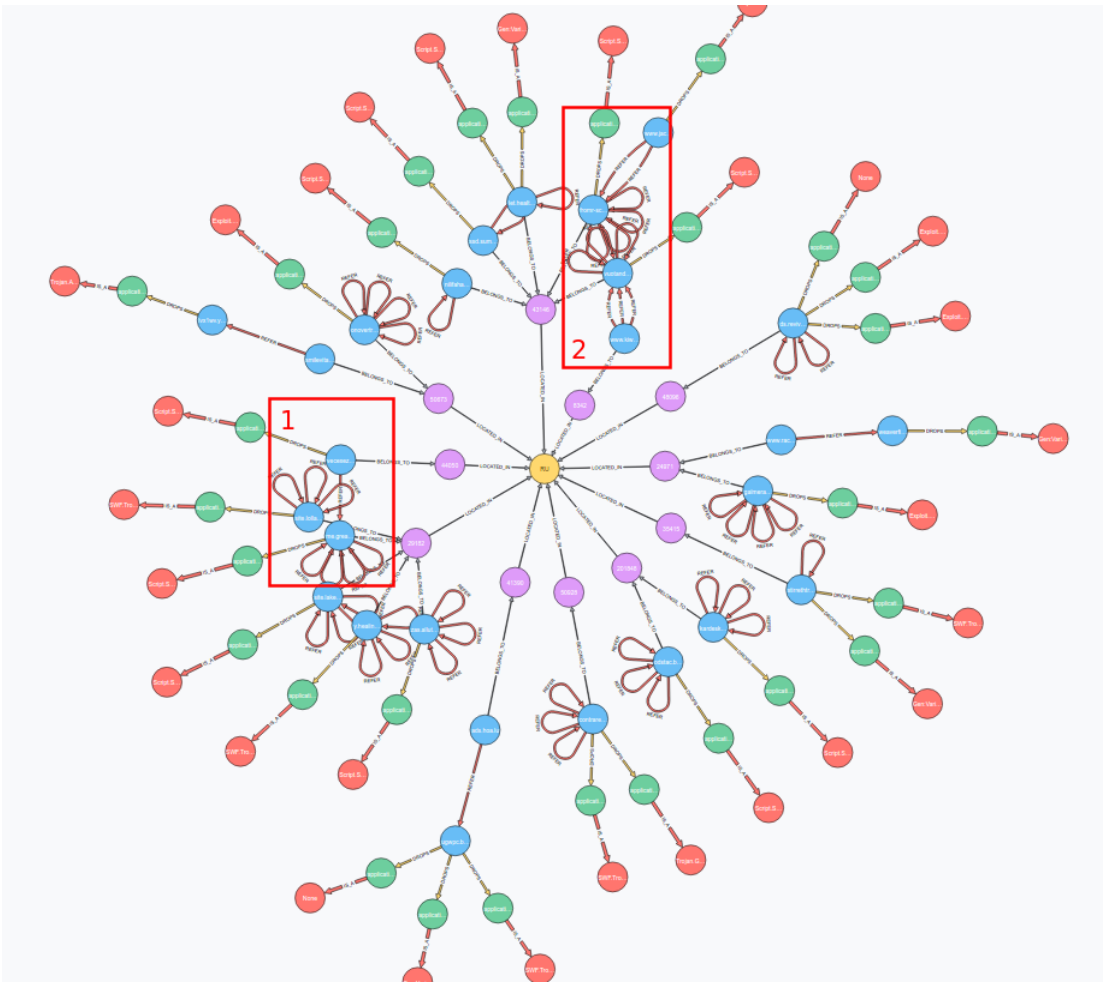
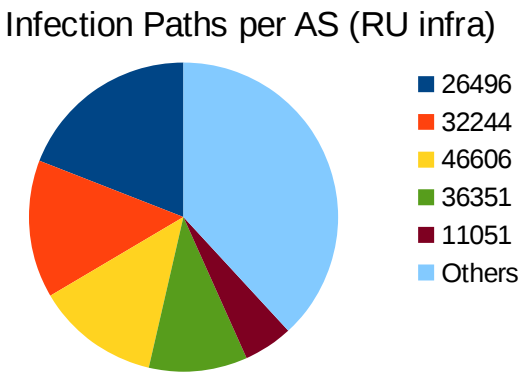


Figure 23: Clusters observed in the Russian infrastructure of Angler

The clusters 1 and 2 shown in Figure 23 are pointing at hosts that are also sharing a malware delivery mechanism across different AS and hosts.

The statistics for the number of infection paths per AS on the Russian infrastructure obtained with the query: ***MATCH p=(c:Country {name:'RU'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths ORDER BY count(a) DESC*** are shown in Figure 24.



ASN	InfectionPaths
26496	37
32244	28
46606	25
36351	20
11051	10
Others	74

Figure 24: Infection paths statistics for the Russian Angler infrastructure

5.1.3 European infrastructure

Looking specifically at the infection paths that are going through the infrastructure of Angler, we get the graph shown in Figure 25 with: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' AND c.name='DE' OR c.name='IT' OR c.name='FR' OR c.name='NL' Return p***.

The figure 25 can be compared with the two previous results shown in Figure 23 for the US-based infrastructure and Figure 24 for the Russian infrastructure. From a topological perspective, the split between fingerprinting and payload delivery across different hosts that is seen in the US, is mixed with hosts that carry both functions as in the Russian case. It would be interesting to put this analysis in further perspective to understand if the characteristics that are seen in the US and Russia would be the mark of different actors that are also targeting Europe or if the observations we are making there are totally independent.

Also, it would add value to understand how these patterns evolve over time when taken separately from a geographical perspective and also as a whole. However, this work does not go in that direction but should be considered as an extension for future research.

The patterns seen with Angler will serve as a baseline only from a topological reference for the study of Rig and Neutrino and later chapters.

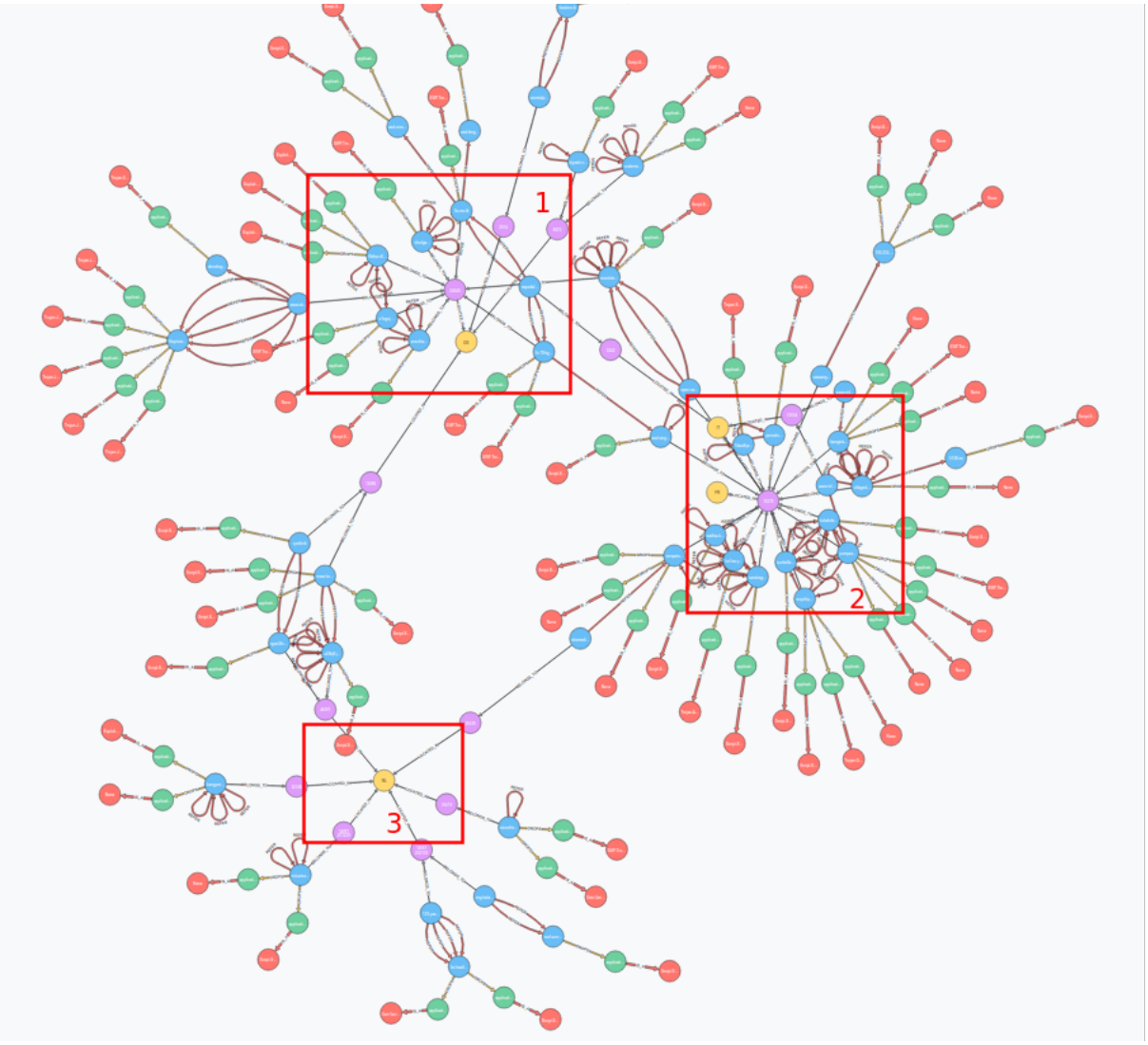
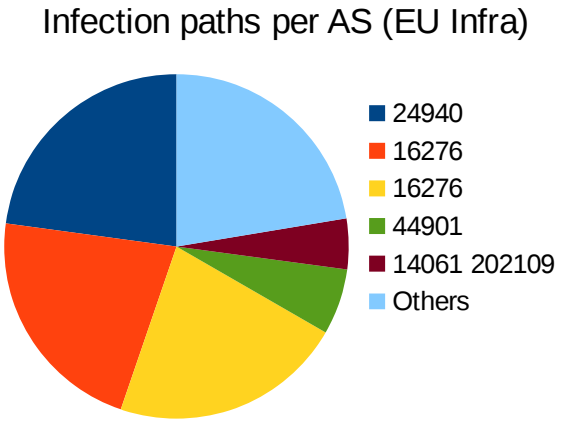


Figure 25: Clusters observed in the European infrastructure of Angler

The statistics for the European infrastructure are returned with the query: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' AND c.name='DE' OR c.name='IT' OR c.name='FR' OR c.name='NL' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths, c.name as COUNTRY ORDER BY count(a) DESC*** and are shown in Figure 26.

If there is a predominance of German hosts in those infection paths, the French and Italian numbers are not very far behind and this distribution is completed by Dutch hosts and others. It is to be noted that the infection path count for AS 16276 is doubled-up and accounts for both the French and Italian infection paths leading back to OVH.

For an unknown reason, there is a duplication of the AS numbers 14061 and 202109 who both belong to Digital Ocean and this result needs to be further investigated.



ASN	InfectionPaths	COUNTRY
24940	48	DE
16276	46	IT
16276	46	FR
44901	13	NL
14061 202109	10	NL
Others	47	

Figure 26: Infection paths statistics for the European Angler infrastructure

5.1.4 Observations and possible mitigations

The infrastructure used to spread malware via the Angler exploit-kit shows a remarkable diversity in the countries and autonomous systems hosting the servers that are compromised with this exploit-kit. Their location is almost evenly spread between the US, Russia and Europe.

At a geographical level, the autonomous systems involved also show a wide spread and it is not possible to conclude that there is a specific prevalence towards one or a specific group. One of the assumptions that can be drawn from there is that Angler is commodity exploit-kit that seems to appeal to various malicious campaigns and criminal groups.

From a defensive perspective, this situation makes it complicated to mitigate this threat: blacklisting the autonomous systems that are responsible for its spread would also mean that entire address blocks hosting legitimate services would also be cut off from the end-users. The visual analysis helps in pinpointing some specific hosts that show multiple redirection paths: these provide resilience to the malware delivery infrastructure and should be treated with a higher priority when deciding on which to filter out in the first place.

However, seeing that infections paths are aggregating around specific autonomous system owners seems to point at structural weaknesses in their security posture. It would be interesting on further investigate those hosts and domains that are involved in the spread of malware through Angler to understand if they would also belong to specific type of customer or server hosting products from their technical owner. While higher cooperation and better security could be achieved in this way, it would ultimately be left to the other party to implement the security measures and practices that would get rid of these threats.

5.2 Rig exploit-kit infrastructure analysis

For this case, all the 112 PCAPs that have been attributed to a Rig exploit-kit are pushed through our system and a general overview of the Infection Paths resulting from the query:

`MATCH(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` will generate Figure 27.

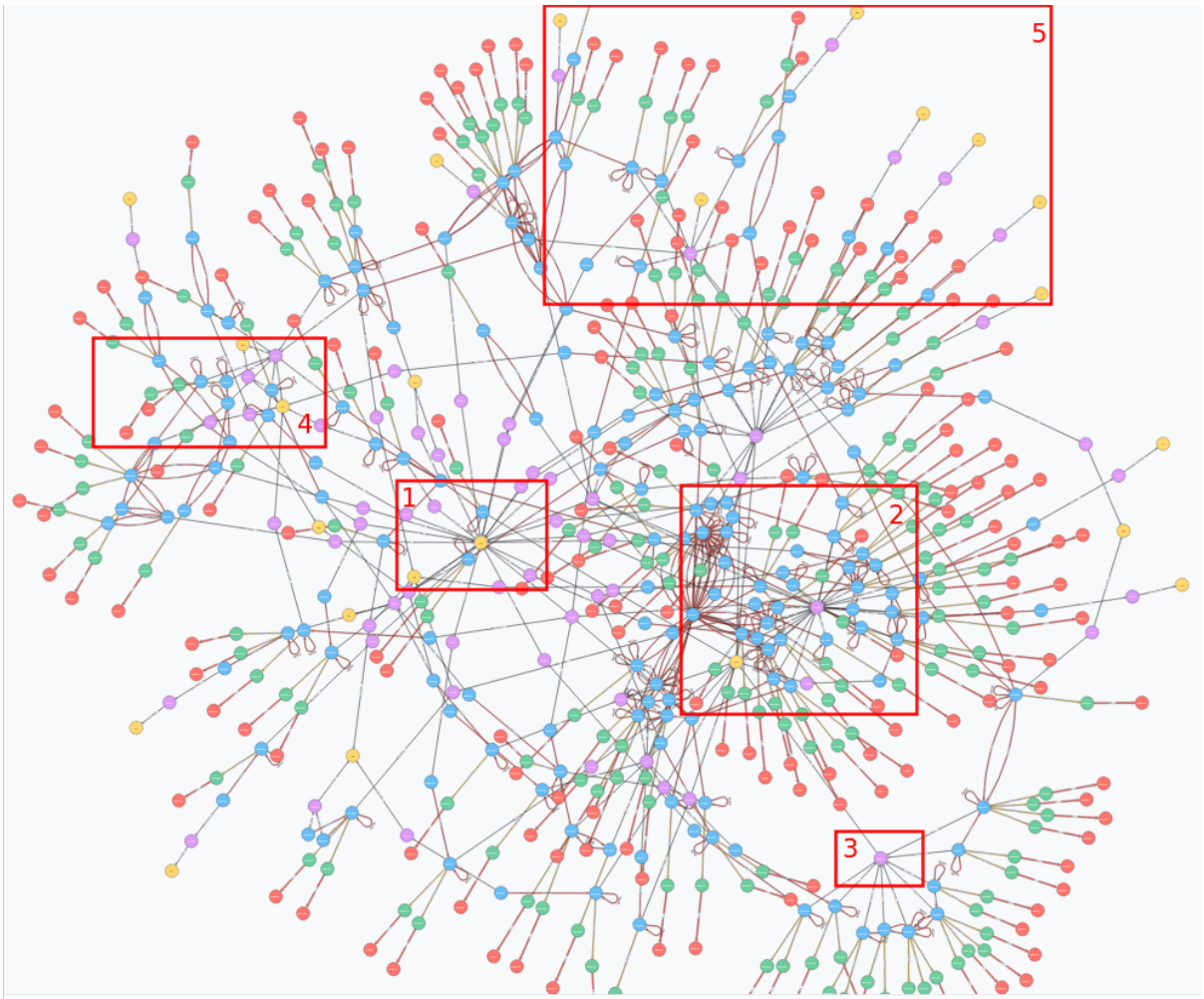
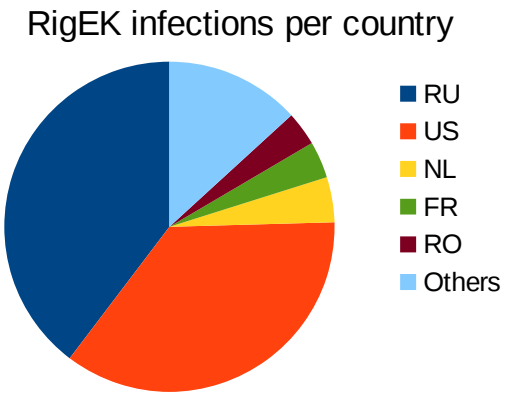


Figure 27: Clusters observed in the RIG infrastructure

In Figure 27: the cluster 1 shows activity centred around the US country node which has relationships in all directions of the graph. The cluster 2 shows activity centred around the RU country node and shows a quantity of host nodes that are redirecting to one another. The cluster 3 shows a lone AS with a large number of compromised hosts and positive detections and a REFER relationship to hosts that are on the side of the infrastructure that is also in Russia. The cluster 4 shows a part of the European infection campaign through loosely related hosts that are prolonged towards the cluster number 5.

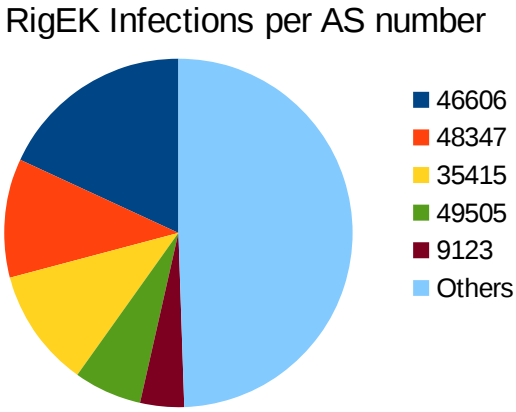
This observation is consistent with the numbers that we have previously extracted from the database but the granularity of the graph makes it hard to use with the built-in browser.

The statistics for infection paths per country are shown in Figure 28 and per as AS in Figure 29.



Country	Infections
RU	252
US	227
NL	28
FR	23
RO	21
Others	84

Figure 28: Rig Infection Paths per Country



ASN	Number of infections
46606	115
48347	70
35415	70
49505	40
9123	26
Others	314

Figure 29: Rig Infection Paths per AS

5.2.1 US-based infrastructure

A focus on the US infrastructure with the query: `MATCH p=(c:Country {name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]->(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` will return the graph shown in Figure 30.

In Figure 30, the cluster 1, we see that we have a first host on the US infrastructure that connects with other hosts that are not related to a US AS number, This seems in line with the general picture that we have from the previous visualization.

It can also be noticed that the browser fingerprinting and payload delivery functions are always split in two, with the fingerprinting host redirecting the browser to the payload delivery stage.

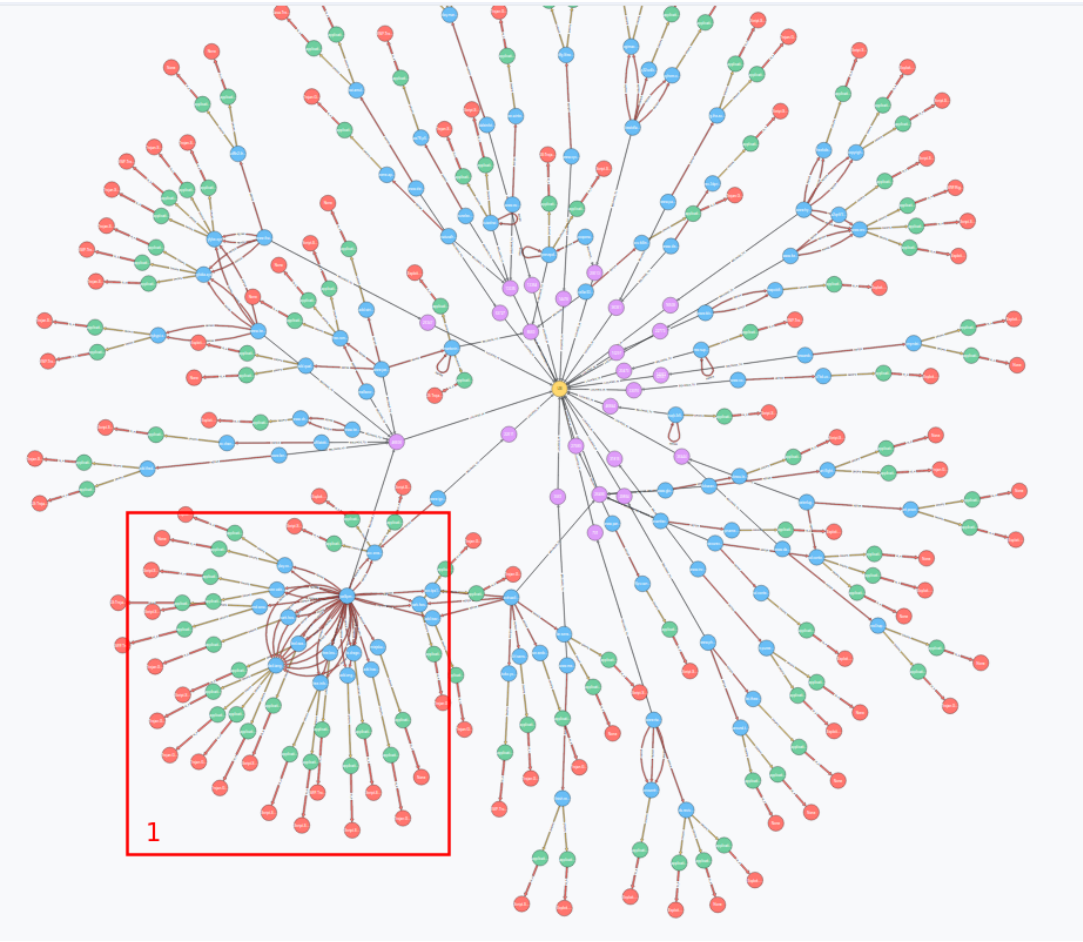
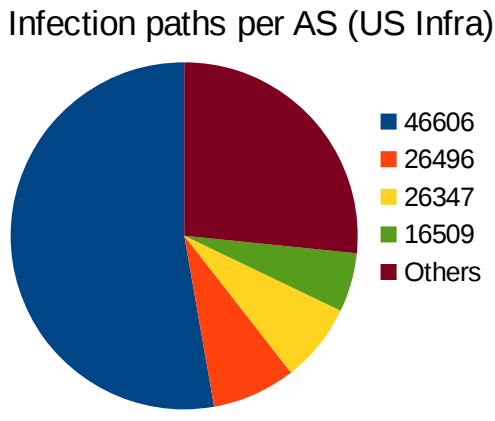


Figure 30: Clusters observed in the US infrastructure of RIG

A specific focus on the US part of the infrastructure of Rig summarized in Infection Paths per AS is shown in Figure 31.



ASN	Number of infection paths
46606	115
26496	17
26347	16
16509	12
Others	58

Figure 31: Infection paths statistics for the US infrastructure of RIG per AS

A focus on the Russian infrastructure of Rig with the query: `MATCH p=(c:Country {name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]->(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` will return the graph shown in Figure 32.

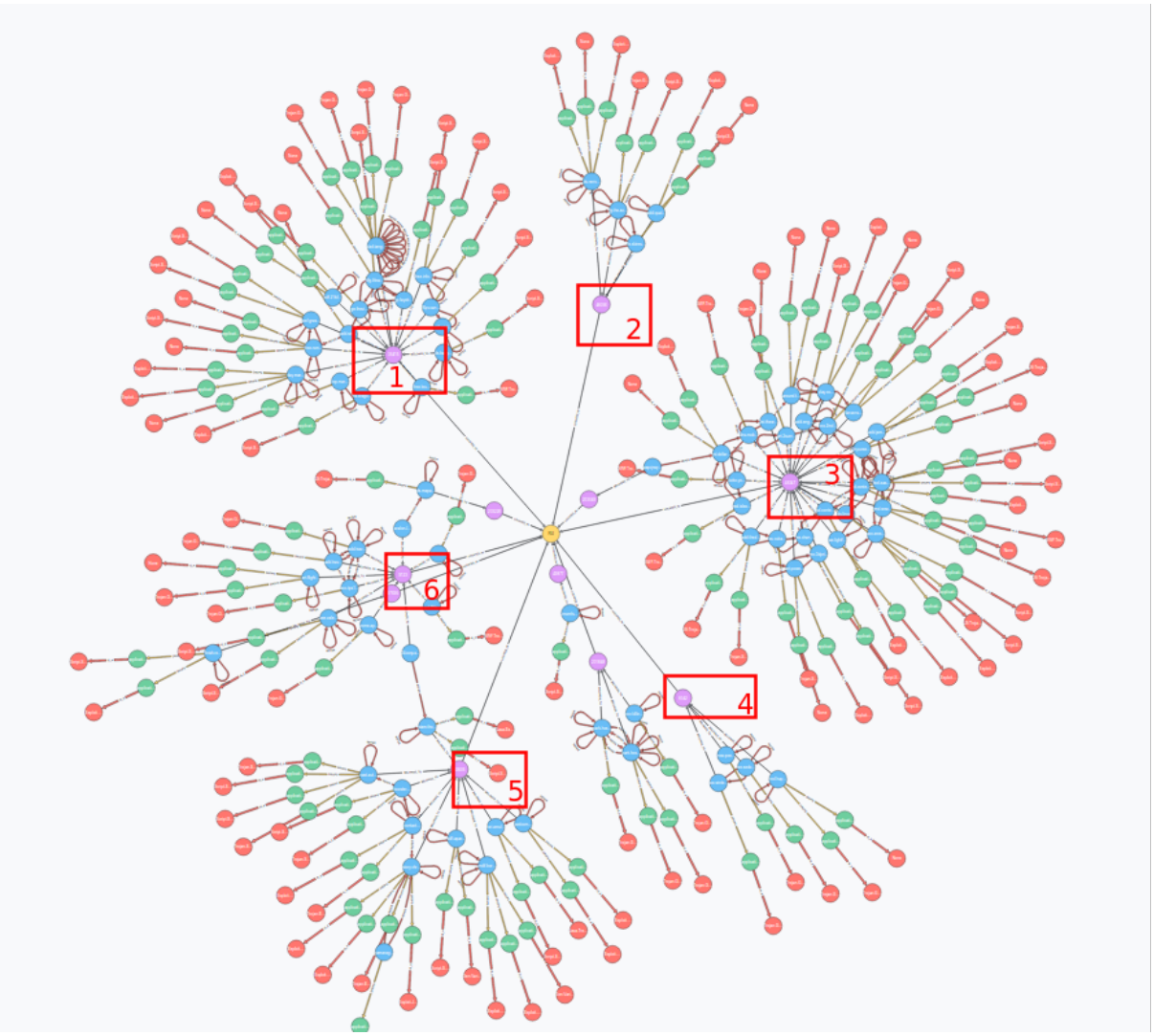
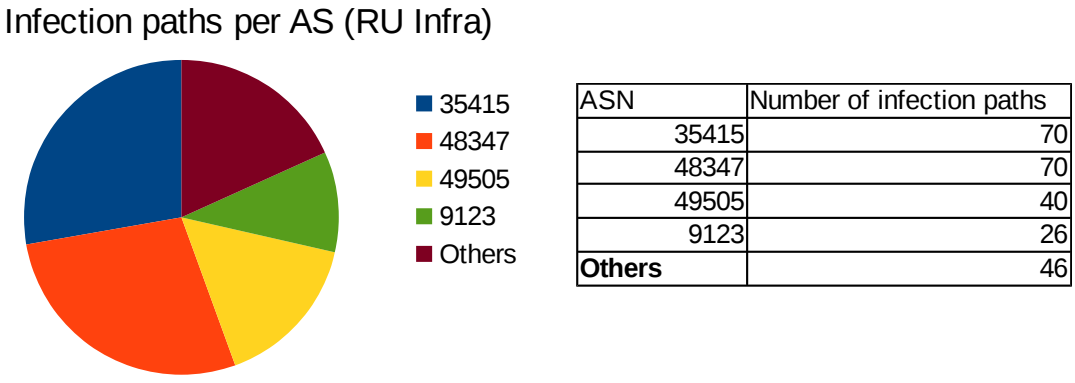


Figure 32: Clusters observed in the Russian infrastructure of RIG



ASN	Number of infection paths
35415	70
48347	70
49505	40
9123	26
Others	46

Figure 33: Infection paths statistics for the Russian infrastructure of Rig per AS

A breakdown of the Infection Path per AS from the Russian infrastructure of Rig is shown in Figure 33.

5.2.2 European infrastructure

A view of the Rig infrastructure based in Europe is obtained via the query: `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]-(f:File)--(o:PAYLOAD) WHERE n.type= 'PUBLIC' AND c.name='RO' OR c.name='NL' OR c.name='FR' RETURN p` and is shown in Figure 34.

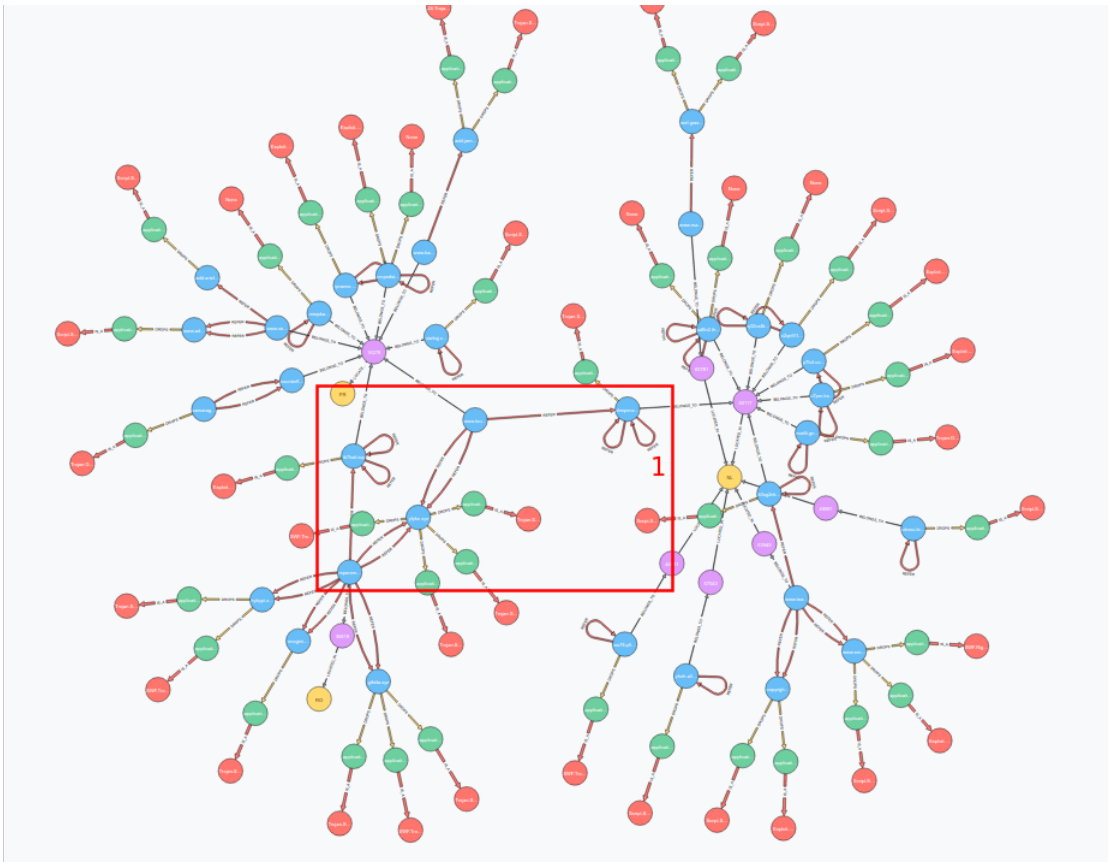
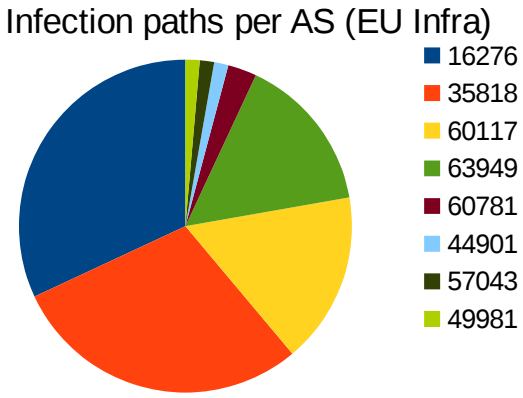


Figure 34: Clusters observed in the European infrastructure of RIG

It is interesting to note here that the French, Romanian and Dutch components are linked together via single hosts that work as gateways. However, when compared with the US and Russian infrastructures where the fingerprinting and payload delivery are split across two different hosts, the European infrastructure mixes the two approaches.

The statistics for the European infrastructure of Rig are obtained with the query: `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' AND c.name='RO' OR c.name='NL' OR c.name='FR' RETURN DISTINCT a.number as ASN, count(a), c.name as COUNTRY ORDER BY count(a) DESC` and shown in Figure 35.

They show that the infection paths are emerging from 3 countries (France, Romania and Netherlands) that are responsible for the large majority of all the recorded infections. These also stem from 4 AS numbers that are responsible for more than 75% of all the recorded infections.



ASN	Number of infection paths	COUNTRY
16276	23	FR
35818	21	RO
60117	12	NL
63949	11	NL
60781	2	NL
44901	1	NL
57043	1	NL
49981	1	NL
	Number of infection paths	COUNTRY
	23	FR
	21	RO
	28	NL

Figure 35: Infection paths statistics for the European infrastructure of Rig

5.2.3 Observations and possible mitigations

The analysis of Rig shows a different picture when compared to Angler: Rig is mostly prevalent in the US and Russia but the repartition of the infection paths for both is completely different. In the US case, one specific AS is responsible for the majority of the infection path whereas in the Russian case, the top four account for the majority of infection paths. The European infrastructure is however largely under-represented when compared to the others.

Once again, this situation makes it difficult to mitigate against that threat but we can easily single out the AS numbers that are responsible for the spread of malware: these can be filtered out at a routing level and contact should be taken with their owners to take corrective measures.

5.3 Analysis of Neutrino Exploit kit infrastructure

In this case, the 61 PCAPs from our dataset that are known to show a Neutrino incident have been put through the system and a visual perspective obtained via the query: `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` is shown in Figure 36.

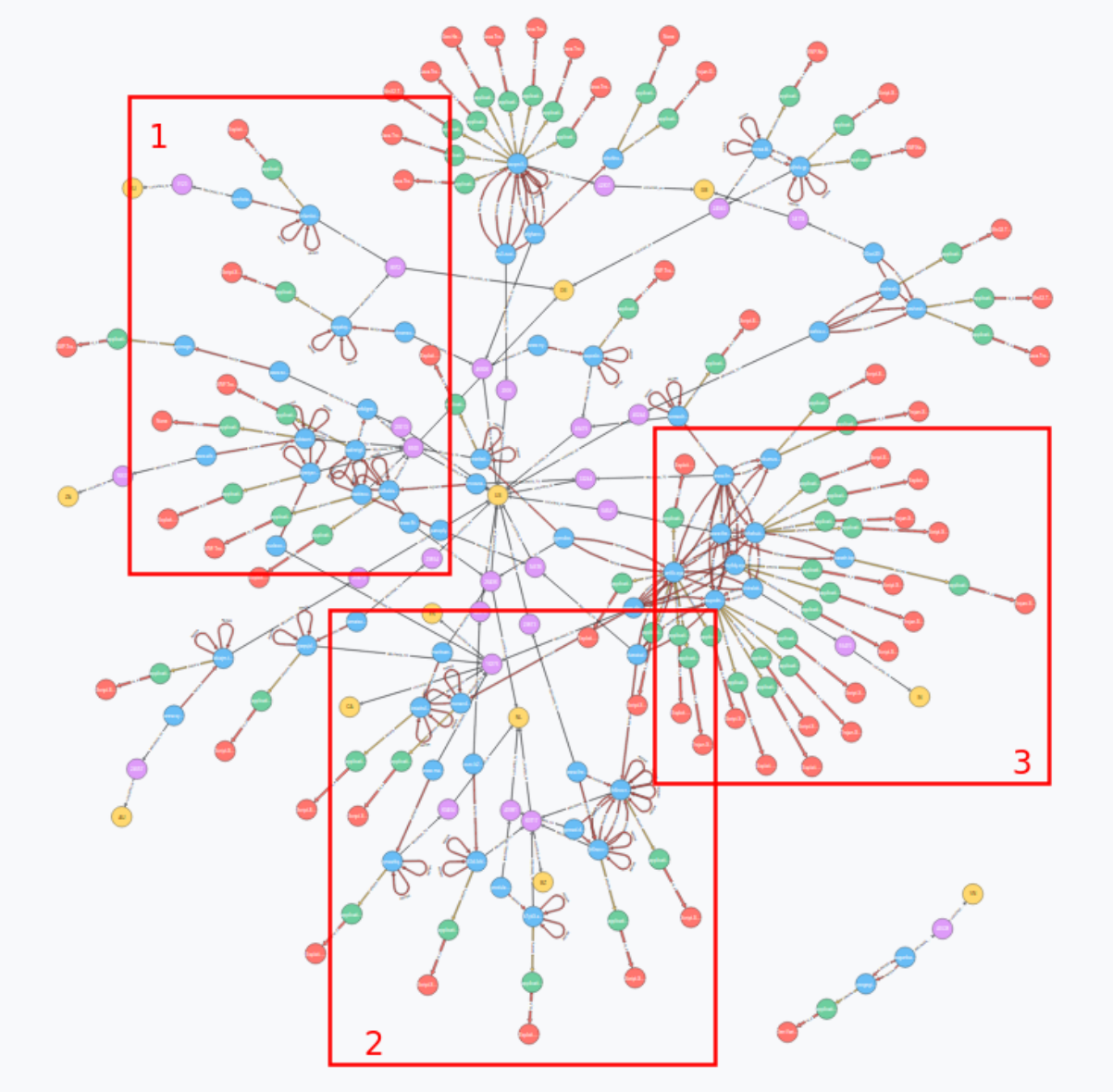
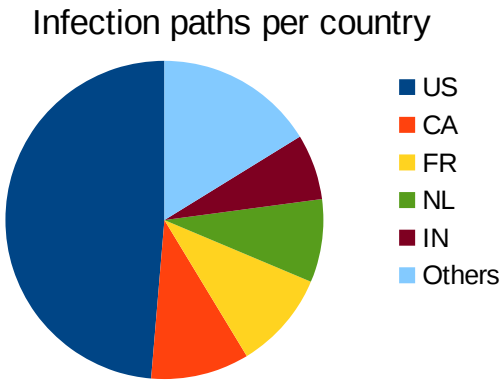


Figure 36: Clusters observed in the Neutrino exploit-kit infrastructure

The general view of the graph in Figure 36 for the Neutrino exploit-kit shows cluster 1 that is centered around the US and the shares some infrastructure with the German part of the infrastructure. Clutser 2 is mainly associated with hosts in Europe and we can also see that they share relationships with hosts from the cluster 3 that is located in the US.

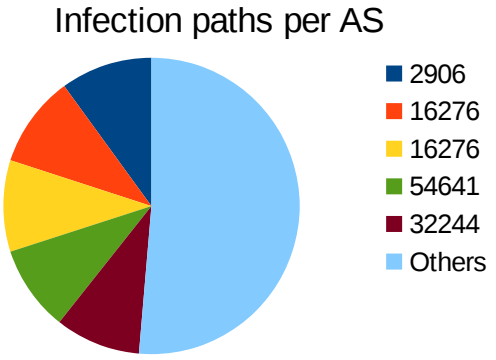
The statistics of the Infection Paths per country obtained with the query: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT c.name, count(c) ORDER BY count(c) DESC*** are shown in Figure 37.



Country	InfectionPaths
US	219
CA	45
FR	45
NL	38
IN	30
Others	73

Figure 37: Infection paths statistics per country for Neutrino exploit-kit

The statistics for Infection Path per AS obtained with the query: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths ORDER BY count(a) DESC*** are shown in Figure 38.



ASN	InfectionPaths
2906	45
16276	45
16276	45
54641	42
32244	42
Others	231

Figure 38: Infection paths statistics per AS for Neutrino exploit-kit

A specific look at the US infrastructure of Neutrino with the query: ***MATCH p=(c:Country {name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]->(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p*** will return the graph shown in Figure 39.

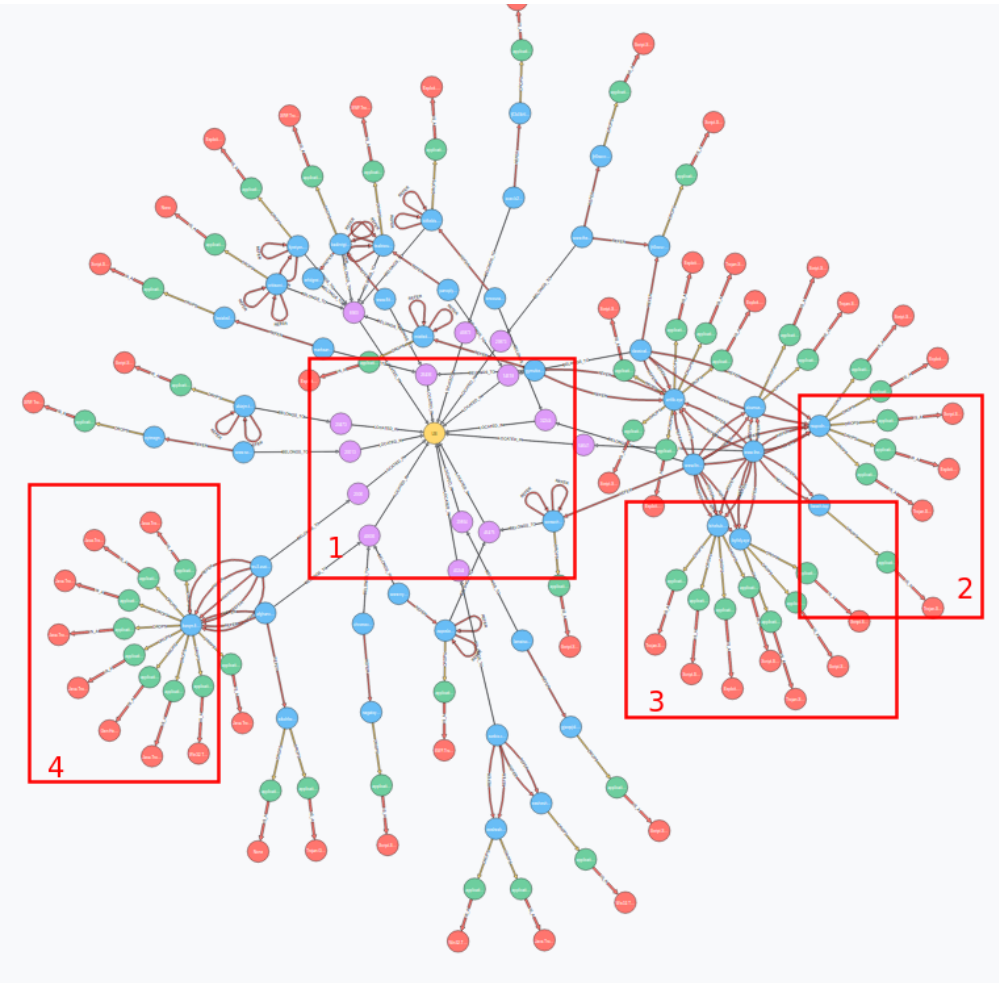
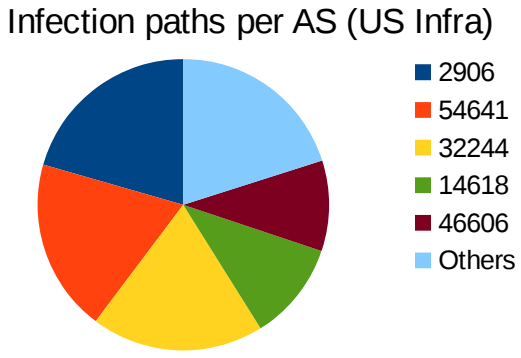


Figure 39: Clusters observed in the US infrastructure of Neutrino

The US part of the Neutrino exploit-kit depicted in Figure 39 shows some interesting clusters with cluster 1 showing a number of AS that are massively used to spread malware. The clusters 2, 3 and 4 are showing infection paths were the malware delivery is happening outside of the US. The statistics for the nfection Paths per AS for the US infrastructure with the query: ***MATCH p=(c:Country{name:'US'})-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN DISTINCT a.number as ASN, count(a) as InfectionPaths ORDER BY count(a) DESC*** are shown in Figure 40.



ASN	InfectionPaths
2906	45
54641	42
32244	42
14618	24
46606	22
Others	44

Figure 40: Infection paths statistics for the Neutrino US infrastructure per AS

Outside of the US, we get can a general view of the infrastructure of Neutrino exploit-kit with the query `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' AND c.name<>'US' RETURN p` which shows the graph in Figure 41.

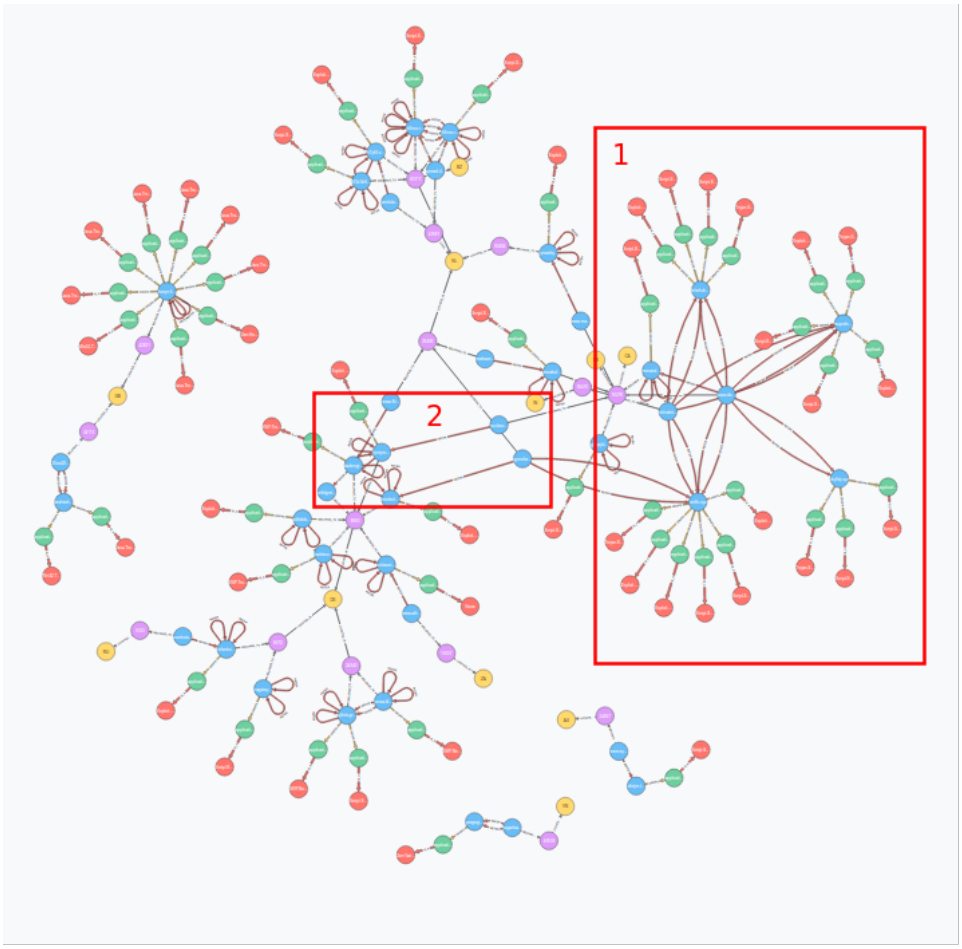


Figure 41: Clusters observed in the non-US based infrastructure of Neutrino

In Figure 41, the cluster 1 shows a part of the European infrastructure that has delivery capabilities outside of the EU as we see that the end host nodes in blue are not connected to an AS in the EU region. The cluster 2 shows hosts spread across different countries sharing a REFER relationship and are sharing their malware delivery capabilities.

5.3.1 Observations and possible mitigations

The study of the infrastructure of Neutrino shows surprisingly that there is not a Russian component to it but that it is focused on European and American hosts instead. The spread by AS is also quite large which makes the mitigation difficult to apply from a single angle.

Corrective actions in this case should be the blacklisting of the domains, sub-domains and IP addresses that are detected here. Once again, contact should be taken with the AS owners to further investigate the root cause of these infection paths.

5.4 PseudoDarkLeech campaign analysis

From our dataset we have a set of PCAPs that have been identified has being from the Pseudo DarkLeech campaign. Pseudo DarkLeech is a campaign that shows the same mode of website infection that is then used to redirect to an exploit-kit that will in turn propagate a specific malware.

The Pseudo DarkLeech campaign is known to us of a mix of the exploit-kits we have analysed before.

From non-geographical perspective, a view of all the hosts involved in the Pseudo DarkLeech campaign is obtained with the query: `Match p=(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' RETURN p` and shown in Figure 42.

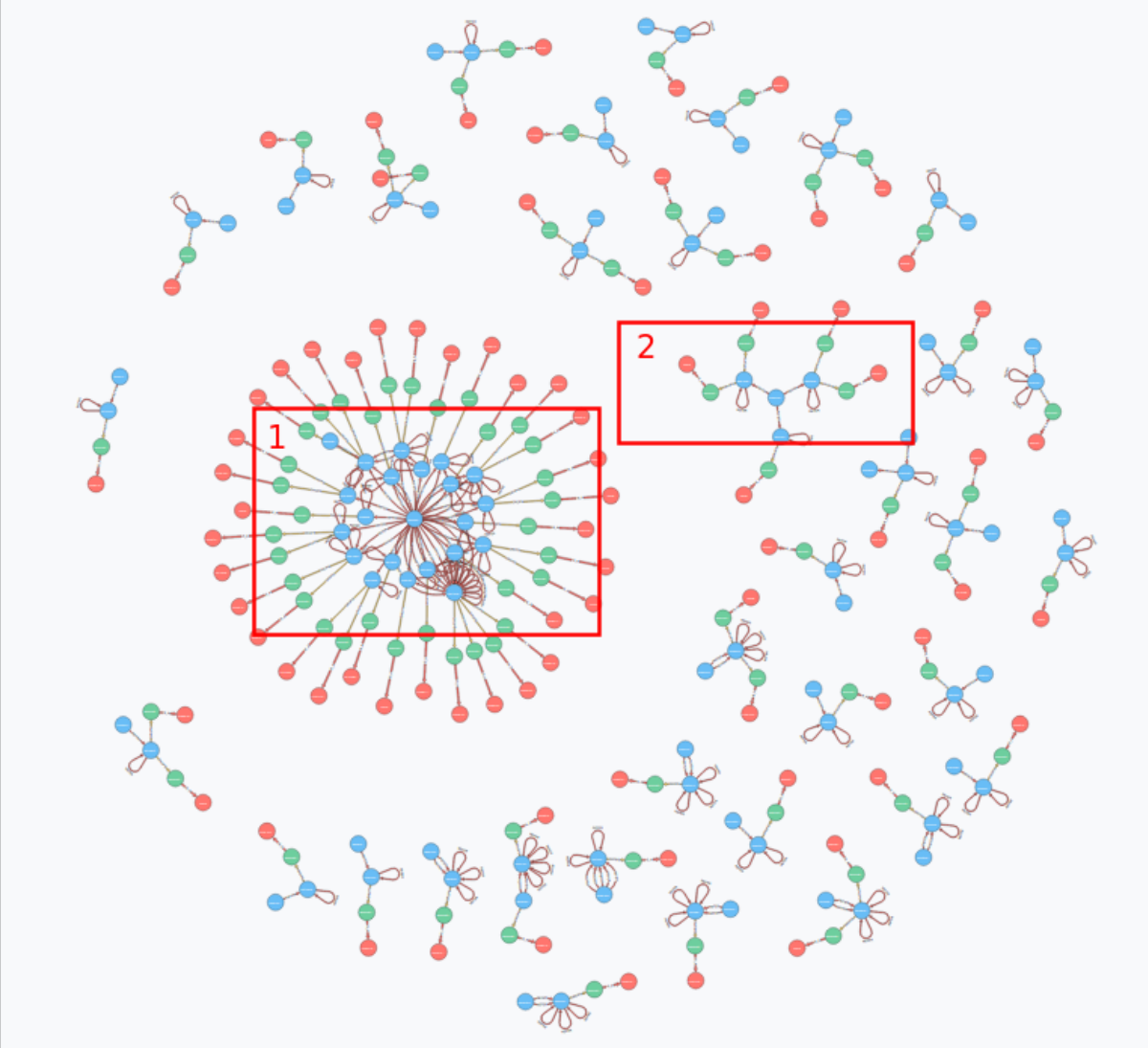


Figure 42: Host clustering observed in the Pseudo DarkLeech campaign

In Figure 42, the cluster 1 shows that the host joellipman.com at the IP address 192.185.225.245 is central to the delivery of malicious payloads. The cluster 2 shows an interesting pattern where the same IP address 192.185.225.245 is also used to spread malware but from the domain name www.joellipman.com.

The Figure 42 also shows a multitude of single paths that do not hold any relationships together. From a higher perspective, the clustering at the AS level is shown with the query: `MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)` with the results shown in Figure 43.

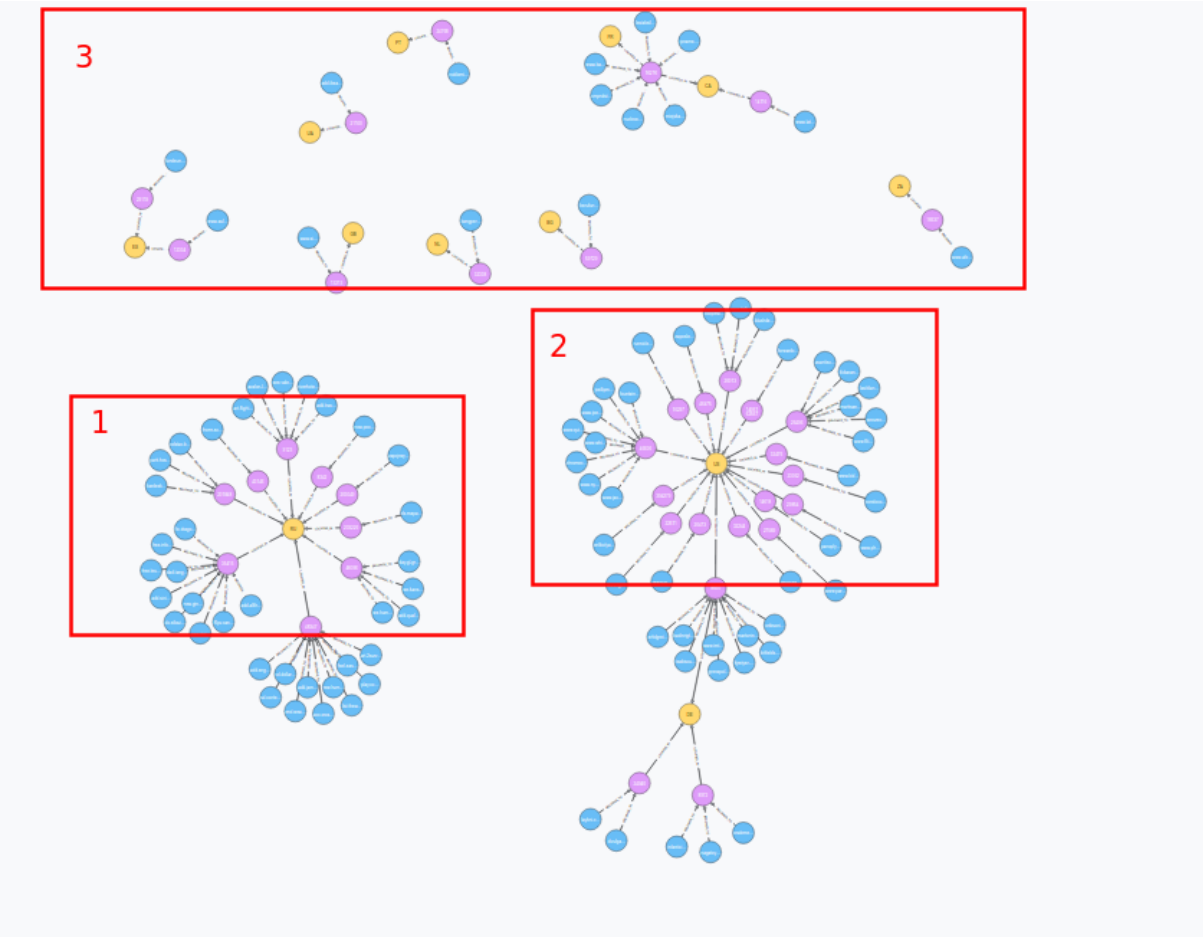


Figure 43: Country clusters observed in the Pseudo DarkLeech campaign

In Figure 43, the cluster 1 shows the part of the exploit-kit infrastructure that resides in Russia. Cluster 2 shows the exploit kit infrastructure that is located in the US in fairly equal proportions as in cluster 1. Cluster 3 is a loosely affiliated group of AS that are situated in Europe.

From the clusters 1 and 2 seen in Figure 43, a further search at the exploit-kit infrastructure that is located both in Russia with the query: ***MATCH p=(n)--(h:Host)-[t:DROPS]→(f:File)--(o:PAYLOAD) MATCH p1=(n)-[u:BELONGS_TO]-(a:ASN)-[v:LOCATED_IN]-(c:Country {name:'RU'}) RETURN p, p1, a.number as asn_number*** will be depicted in Figure 44.

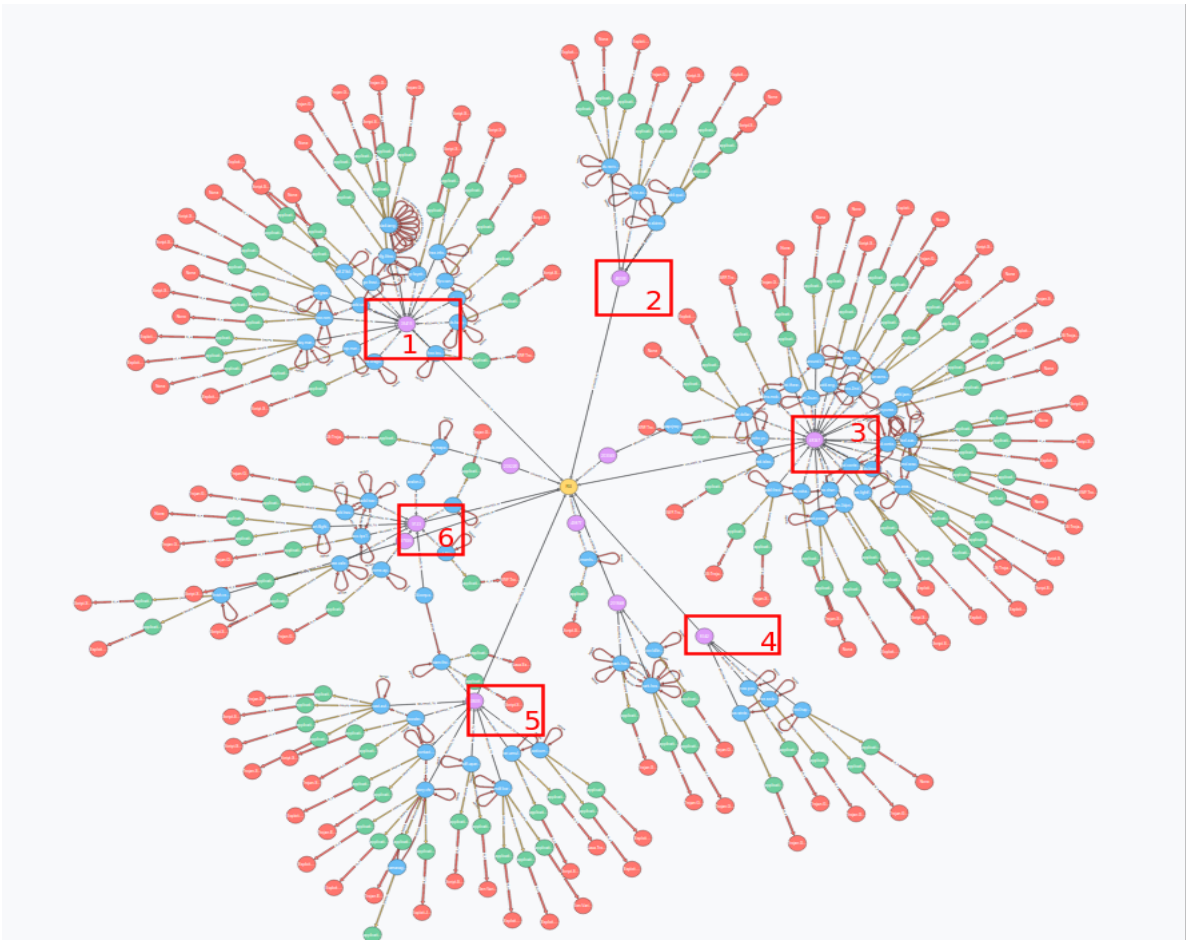


Figure 44: AS clusters in Russia observed in the Pseudo DarkLeech campaign

From the Figure 44, 6 AS numbers with some serious malware launch capability can be isolated and their connected hosts are all showing an exploit kit behaviour. Those AS numbers are: 48347, 35415, 48096, 201848, 43146 and 9123. The number of detected payloads available is 54 with a total of 39 hosts taking part in this campaign.

A similar analysis from the US perspective gives a different picture with the query: ***MATCH p=(c:Country)-[]-(a:ASN)-[BELONGS_TO]-(n)--(h:Host)-[t:DROPS]->(f:File)--(o:PAYLOAD) WHERE n.type='PUBLIC' AND c.name='US' RETURN p*** and its result prsented in Figure 45.

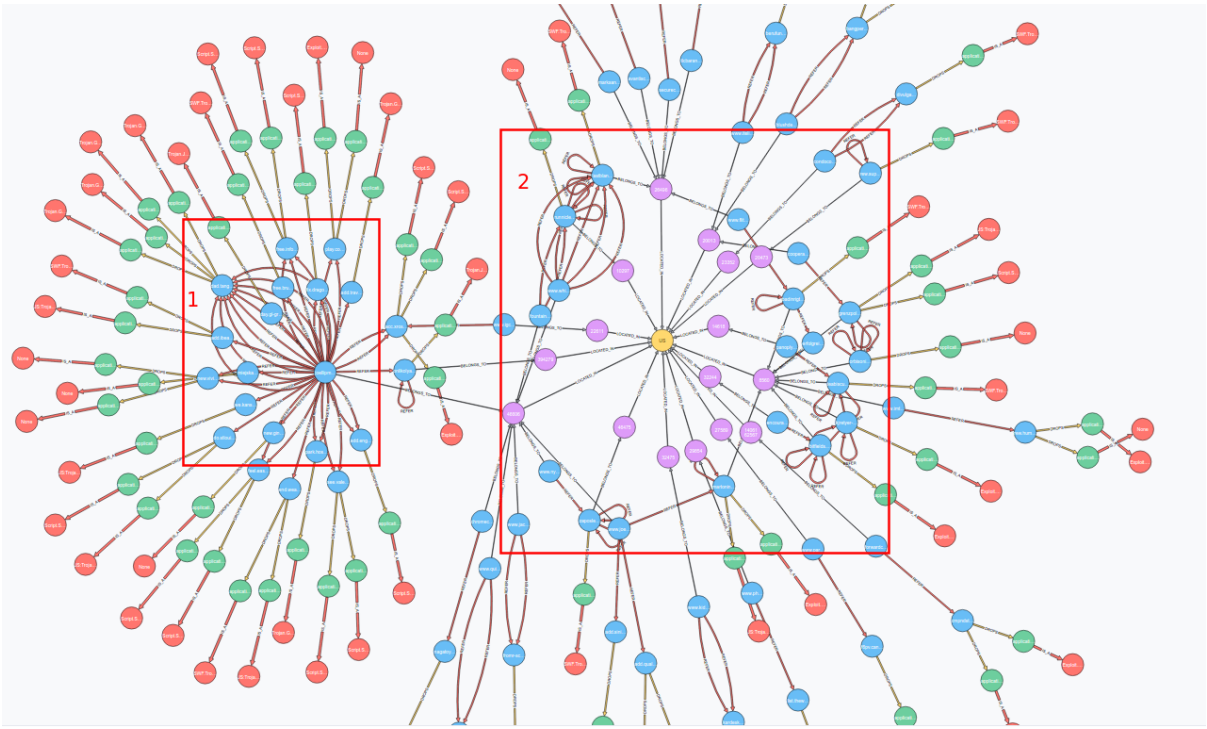


Figure 45: Clusters observed in the US infrastructure of the Pseudo DarkLeech campaign

The US infrastructure clearly shows again a host (joellipman.com) that is responsible for 87 infection paths and shown in the cluster 1. The cluster 2 shows a couple of hosts responsible for known infection paths that are redirecting one to the other and offer infection paths that are spread across different AS numbers.

5.4.1 Observations and possible mitigation

The analysis of a specific campaign differs from the study of the infrastructures that we have previously covered. Campaigns are opportunistic in their nature and target specific weaknesses in known services such as Apache in the specific case of DarkLeech.

In this case, the defensive measures would rather be directed internally rather than externally. A further understanding of the vulnerabilities targeted by the attackers is needed as well as their mode of operation would be beneficial for the local system administrator so that proper maintenance and patching could be done to prevent a local compromise.

6 Conclusions and future research

We have developed here a system that lets us analyse single instances of exploit-kit based incidents, as well as full campaigns that span multiple incidents and scenarios. Automating our acquisition process and enriching our data with third-party APIs clearly speeds up the analysis process either from a visual or a database query perspective and we are definitely seeing that those benefits can be driven further.

We are providing through this system an actionable set of IOCs (SHA-1 hash, IP addresses, domain names, AS numbers, country of origin) that are directly translatable into concrete defensive measures and we understand from this work that the richer the data we can acquire to enrich our existing set, the more granularity we are able to obtain in understanding the structure and relationships between different types of exploit-kits and campaigns.

We have developed here only a simple data-model and set of queries for our research and we are now at a point where we can carry on with the same graph-oriented direction but with more specific goals as our intent here was only to prove that this framework could be valuable and yield conclusions that would be difficult to extract in a usual context.

From an analysis perspective, we have seen that the exploit-kits we have studied present patterns that are specific to their delivery infrastructure. It would be of a great interest to see if and how these patterns evolve if we could further extend our dataset with static analysis IOCs, forensics artefacts, analysis engine traces…

In terms of defensive measures, the possibility to relate new events to a set of known malicious patterns provides an adaptive set of practices: for example, if a file that goes undetected is known to have been downloaded from a host that is related to a malicious infrastructure, preventive measures like host quarantine can be applied right away based on the suspicions raised and further manual corrections can be done from there. Machine-learning being one of the hottest topics in cyber security, we could also bring this approach towards the automatic detection of malicious patterns.

Another direction in which our research can also be used is to survey the liveliness of the infrastructure that we have studied in order to get an actual “live map” of them. The idea would then be to run availability checks on the hosts and domain names that we have extracted and see if exploits and malware are still being delivered from them. This approach could be automated with honeypots that not only emulate the browser personalities that are known to be targeted but also by specific variations in order to survey the offensive capabilities that are available to attackers.

References

[1] A. K. Sood, R. J. Enbody, “Malvertising–exploiting web advertising” in *Computer Fraud & Security*, pp 11-16, Apr. 2011.

[2] C. Grier, L. Ballard *et al.*, “Manufacturing Compromise: The Emergence of Exploit-as-a-Service.” in *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 821-832. ACM, 2012.

[3] I. Robinson, J. Webber *et al.*, (2015, June). *Graph Databases. 2nd Edition* [Online]. Available: https://pdfs.semanticscholar.org/f511/7084ca43e888fb3e17ab0f0e684cced0f8fd.pdf [accessed April 2017].

[4] ENISA, (2016, Jan.). *ENISA Threat Landscape 2015* [Online]. Available: https://www.enisa.europa.eu/publications/etl2015/at_download/fullReport [accessed April 2017].

[5] CERT-UK, (2016, Jan.). *Demystifying the exploit kit - Context Information Security* [Online]. Available: https://www.contextis.com/documents/171/Demystifying_the_Exploit_Kit_-_Context_White_Paper.pdf [accessed March 2017].

[6] V. Kotov, F. Massacci *et al.*, “Anatomy of Exploit Kits – Preliminary Analysis of Exploit Kits as Software Artefacts” in *International Symposium on Engineering Secure Software and Systems*, pp. 181-196. Springer Berlin Heidelberg, 2013.

[7] Symantec Labs. (2012, Dec.). *Internet Explorer Zero-Day Used in Watering Hole Attack: Q&A* [Online]. Available: https://www.symantec.com/connect/blogs/internet-explorer-zero-day-used-watering-hole-attack-qa [accessed March 2017].

[8] Symantec Labs. (2014, Feb.). *New Internet Explorer 10 Zero-Day Discovered in Watering Hole Attack* [Online]. Available: https://www.symantec.com/connect/blogs/new-internet-explorer-10-zero-day-discovered-watering-hole-attack [accessed March 2017].

[9] Y. Lin. (2013, May). *IE Zero Day is Used in DoL Watering Hole Attack* [Online]. Available: https://www.fireeye.com/blog/threat-research/2013/05/ie-zero-day-is-used-in-dol-watering-hole-attack.html [accessed March 2017].

[10] Hypertext Transfer Protocol – HTTP/1.1, RFC 2616, 1999.

[11] D. Best, A. Endert *et al.*, “7 key challenges for visualization in cyber network defense.” in *Proceedings of the Eleventh Workshop on Visualization for Cyber Security*, pp. 33-40. ACM, 2014.

[12] J. Rasmussen, K. Ehrlich, *et al.*, “Nimble cybersecurity incident management through visualization and defensible recommendations.” in *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, pp. 102-113. ACM, 2010.

[13] S. Noel, E. Harley et *al.*, “CyGraph: Graph-Based Analytics and Visualization for Cybersecurity,” *Handbook of Statistics* 35 (2016): 117-167.

[14] H. Soeda. (2016, Feb.). *Analyzing Targeted Attacks through “Hiryu” An IOC Management and Visualization Tool.* [Online]. Available: https://www.first.org/resources/papers/munich2016/soeda-hiryu-the-ioc-management.pdf [accessed April 2017].

[15] Symantec Labs. (2014, Nov.). *Operation CloudyOmega: Ichitaro zero-day and ongoing cyberespionage campaign targeting Japan.* [Online]. Available: https://www.symantec.com/connect/blogs/operation-cloudyomega-ichitaro-zero-day-and-ongoing-cyberespionage-campaign-targeting-japan [accessed April 2017].

[16] M. Marschalek, R. Vinot. (2017, Mar.). *Graph me, I'm famous! Automated static malware analysis and indicator extraction for binaries.* [Online]. Available: https://www.troopers.de/troopers17/talks/774-graph-me-im-famous-automated-static-malware-analysis-and-indicator-extraction-for-binaries/ [accessed April 2017].

[17] B. Duncan. (2016, Mar.). *Campaign Evolution: Darkleech to Pseudo-Darkleech and Beyond.* [Online]. Available: http://researchcenter.paloaltonetworks.com/2016/03/unit42-campaign-evolution-darkleech-to-pseudo-darkleech-and-beyond/ [accessed March 2017].

[18] B. Duncan. (2017, Jan.). *Campaign Evolution: EITest from October through December 2016.* [Online]. Available: http://researchcenter.paloaltonetworks.com/2017/01/unit42-campaign-evolution-eitest-october-december-2016/ [accessed March 2017]

[19] B. Duncan. (2014, Apr.). *Afraidgate: Major Exploit Kit Campaign Swaps Locky Ransomware for CryptXXX.* [Online]. Available: http://researchcenter.paloaltonetworks.com/2016/04/afraidgate-major-exploit-kit-campaign-swaps-locky-ransomware-for-cryptxxx/ [accessed March 2017].

[20] The Bro Network Security Monitor. [Online]. Available: https://www.bro.org/ [accessed March 2017].

[21] Neo4j. (2017, Feb.). *Cypher Query Language reference.* [Online]. Available: https://neo4j.com/developer [accessed March 2017].

[22] J. Liang, J. Jiang *et al.*, "When HTTPS meets CDN: A case of authentication in delegated service." in *Security and privacy (sp), 2014 ieee symposium.* (2014 Dec.) [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6956557 [accessed April 2017].

[23] MITRE Corporation. (2010, Mar.). *CVE-2010-1028* [Online]. Available : https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-1028 [accessed April 2017].

[24] MITRE Corporation. (2015, Feb.). *CVE-2014-9668* [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-9668 [accessed April 2017].

Appendix

I. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Guillaume Brodar**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:
   1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and
   1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

Analysis of exploit-kit incidents and campaigns through a graph database framework

supervised by Toomas Lepik and Raimundas Matulevicius,

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **16.05.2017**