How to Spot the Blue Team

Red Team Infrastructure Security

R.A.H. Lahaye
rick.lahaye@os3.nl

March 27, 2018

Supervisors:
M. Bergman and M. Smeets

Abstract

Red and Blue Teaming is a competitive training simulation between the Blue Team and the Red Team. The goal of the Red Team is to infiltrate an infrastructure, stay undetected and steal data. Whereas the Blue Team's goal is to secure this infrastructure by preventing, detecting, and securing from such intrusions.

Knowing if a Red Team's operation has been spotted by the Blue Team or not, could make the difference between losing the target and securing the target. If the Red Team knew their operation was spotted, it could adapt its techniques. This lead to the following research question: *"How to use a Red Team's infrastructure to detect a Blue Team's analysis?"*

A logging and monitored Red Team infrastructure can be used to detect a Blue Team's analysis by using anomaly-based detection. A Proof of Concept was built using the Elastic Stack and Python scripts that is able to detect anomalies in the Command and Control (C2) communication. As the Elastic Stack brought challenges with it in alerting and detection during multiple Red Team operations, a Python script was created. This Python script is able to spot the Blue Team in a fast and dynamic way by searching for anomalies based on the operational details given as input.

1 Introduction

Red and Blue Teaming is a competitive training simulation between two teams; a Red Team and a Blue Team. The Red Team is the attacking team also known as the bad guys, and the Blue Team is the defending team known as the good guys. The goal of the Red Team is to infiltrate an infrastructure, stay undetected and steal data. Whereas the Blue Team's goal is to secure this infrastructure by preventing, detecting, and securing from such intrusions.

Figure 1 demonstrates what such operations by the Red Team entail.

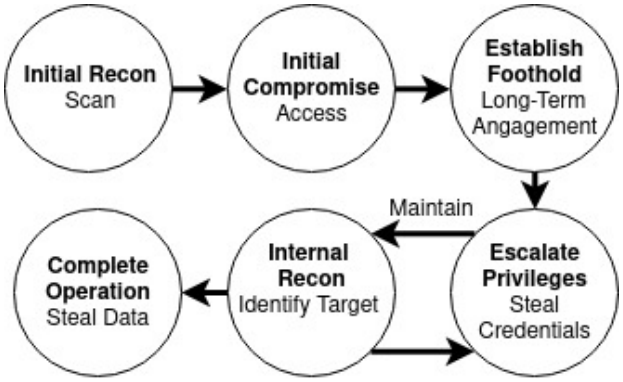

Figure 1: Red Team Operation

A Red Team operation starts with initial recon and compromising a system as entry point of access in the infrastructure. When infiltrated, internal recon is performed as more access is available from within the infrastructure than from the outside. This internal recon can lead up to months of stealing data and credentials, resulting in long-term engagement. Staying in an infiltrated infrastructure for a long period of time, most likely results in being detected by the Blue Team. Knowledge by the Red Team of it being spotted could lead to adaptation of the used methods to become undetected again and ensure keeping its long-term engagement. Hence, detecting a Blue Team's analysis and adaptation to avoid detection is

a continuous process.

The goal of the project is to find a way to detect Blue Team's actions so that the Red Team can stay undetected and achieve its long-term engagement. This resulted in the following research question: *"How to use a Red Team's infrastructure to detect a Blue Team's analysis?"*

Some previous research has been done on how Blue Teams and Red Teams operate[1]. Red Teams use particular attack techniques with an infrastructure designed for this purpose[1][8], where Blue Teams use defending and detecting techniques such as logging, monitoring, and system hardening[4][27]. However, no research has been done on detecting Blue Team's analysis of a Red Team's operation. Knowledge of Red Team operations and infrastructures results in knowledge on how to distinguish between legit event and anomalies of a potential Blue Team's analysis. Therefore, first an overview is given on how a Red Team's operation and infrastructure looks like, next an analysis of Remote Access Tool software that is used by Red Teamers, and as last a Proof of Concept that is able to detect a Blue Team's analysis.

2 Red Team Infrastructure

A basic Red Team infrastructure has been demonstrated in Figure 2. Not all Red Team infrastructures consist of the same structure, although they do often share similarities like a target, a Trusted Third-Party Domain, redirectors, and a Command and Control server. Redirectors and Trusted Third-Party Domains are explained in Section 2.1 and 2.4

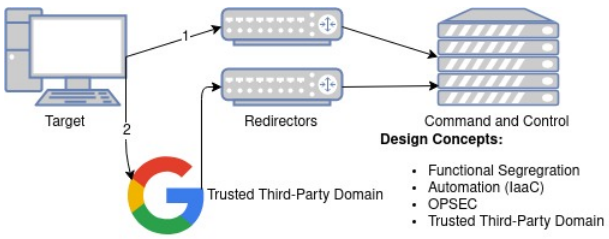

Figure 2: Red Team Basic Infrastructure

As shown in Figure 2, most infrastructures are based on the following design concepts[8]:

- Functional Segregation
- Automation
- Operational Security (OPSEC)
- Trusted Third-Party Domain

[1]Mostly writings on personal blogs.

2.1 Functional Segregation

Functional segregation is the concept of segregating each system based on its function. This results in resilience and agility against the Blue Team by not requiring the disposal of the whole infrastructure when spotted by the Blue Team. Instead, only the particular spotted system can be disposed and recreated. Using this concept, Red Team infrastructures often make use of redirectors to achieve more resilience and concealment. Redirectors are the front-end/edge servers of the Red Team infrastructure that forward traffic from the target to the Command and Control servers. These redirectors provide additional security as they have their own Internet Protocol(IP) address and therefore prevent direct contact with the target[30]. Redirectors are basically relay servers that filter and forward traffic by using IPtables, proxies, rewrite modules, or other methods. Redirectors can be used for the following functions[8]:

- Short/long-term C2 (e.g. HTTP(S), DNS, NTP)
- Sectionerving payloads (e.g. HTTP(S), FTP)
- Phishing sites and emails (e.g. HTTP(S), SMTP)

2.2 Automation

When a Blue Team's analysis has been spotted, one could decide to create and use a new Red Team infrastructure unknown to the Blue Team. The old infrastructure could then be automatically disposed. Continuous Delivery (CD) or Infrastructure as a Code (IaaC) is typically used to allow fast creation and disposal of a Red Team Infrastructure when it has been spotted[8]. The advantage of using CD/IaaC is that systems can be treated as disposable and replaceable objects, instead of systems that are complex and indispensable[23].

2.3 Operational Security

Operational Security (OPSEC) is used to protect the data of a Red Team's infrastructure. OPSEC is the process of protecting information that can be used against the Red Team[26]. Red Team infrastructures contain sensitive information regarding an operation and therefore security controls are desired. A Red Team infrastructure can have zero, one, or multiple

security controls implemented. These security controls can be categorized as follows:

- Preventive Security Controls
- Detective Security Controls
- Responsive Security Controls

Some of the security controls have been explained in the following sections. What security is implemented is a trade-off by the Red Team between the risk of implementation and the usability/result of it[33].

2.3.1 Preventive Security

Preventive security controls can be used to limit and prevent access to the Red Team infrastructure. Preventive security has the highest risk of implementation due to its impact; too much preventive security results in high levels of complexity where a Remote Access Tool's callback could be blocked, too less security results in higher risk of detection and the potential of being hacked by another party[32].

Preventive Security Controls:

- Access Control/Firewall
- Concealment
- System Hardening

Access Control and Firewalls limit access to a system. Limiting access can be done based on multiple characteristics like authentication (username and password authentication, or another type of challenge), IP address, ports, signature on a particular offset in datagram, or particular flags in headers[31][37].

The Command and Control servers should only be accessible by team-members and redirectors. Therefore, any untrusted port and Internet Protocol (IP) addresses should be filtered out, and authentication should be present for both the team-members and the redirectors. Access to redirectors could be limited by particular IPs. These IPs can be company or country ranges, or static IPs used by the Command and Control. Though, limiting access by IP gives challenges that are described in Section 4.3

Concealment can be used for systems and network traffic. The goal of concealment is to have activity classified as legitimate traffic so it does not raise awareness. Domain fronting can be used to communicate with the redirectors as described in Section 2.4. Also encryption is recommended towards and from the redirectors[34].

Hardening, also known as system hardening, can be used to harden a system and make it more prone to attacks. System hardening includes software updates and patching. Common hardening guides can be used[2].

2.3.2 Detective Security

Detective security controls can be used to detect infiltrations of a Red Team infrastructure when preventive security controls have been bypassed.

Detective Security Controls:

- Monitoring and Logging
- Intrusion Detection System

Monitoring and Logging gives an insight on what activity takes place in the Red Team infrastructure. It can be used as an indicator that a Red Team operation has been detected, or when the Red Team infrastructure has been compromised[18].

Intrusion Detection Systems (IDS) can be used to detect counterattacks and anomalies in the Red Team Infrastructure. Using an IDS could show false positives due to the use of exploits and payloads by the Red Team operation[7]. Note that an IDS cannot be used to detect a Blue Team's analysis as discussed in Section 6.

2.3.3 Responsive Security

Lastly, responsive security controls are desired. Responsive security can be used as a response after the detective security controls detected an anomaly. This includes countermeasures or alerting the Red Team.

Responsive Security Controls:

- Alerting
- Disposing

Alerting can be used when the preventive security measures have been bypassed and the detective security measures have detected this. Alerting the Red Team ensures that countermeasures can be taken immediately.

Disposing an infrastructure can be used when a Red Team infrastructure has been compromised, or a Red Team operation has been spotted. After disposing, a new infrastructure can be created with fresh domains and IP addresses[32].

2.4 Trusted Third-Party Domain

Domain fronting is a technique where a trusted third-party provider is used for rerouting traffic towards the Red Team's redirectors, and by this hiding the true endpoint/destination. The idea is to use different domain names at different layers of communication[2]. Often popular services like Amazon CloudFont, Google App Engine, and Microsoft Azure are used for domain fronting. These services are effective as they use a Content Delivery Network (CDN) which is a network consisting of distributed servers used to deliver content to typically clients from the same geographical location. These CDNs decrypt the request with the trusted domain, and then forward the traffic to the destination host domain that is located in the header[28][17].

For example, using domain fronting with HTTPS as communication protocol, traffic will look like the following: DNS query and TLS Server Name will be the trusted provider, and the HTTP packet inside the TLS session[3] will contain the untrusted domain as destination host in the header[17].

Other Trusted Third-Parties can be used as well for Command and Control traffic like file sharing sharing, text paste services, social media, and other services that allow messaging through data.

3 Red Team Software Analysis

An analysis was done of Cobalt Strike[4] and Powershell Empire[5]. Cobalt Strike and Powershell Empire are Remote Access Tools (RAT)/Post-Exploitation Frameworks used by Red Teamers for Red Team operations. These 2 Remote Access Tools, a free one and a commercial one, were chosen because they present the most complete and used frameworks available[29]. Other Remote Access Tools work in similar way.

Remote Access Tools are used as backdoor to remotely control and access a compromised system. Most of these RATs include a Command and Control server and a payload. The payload is the piece of software that contains the malicious code that needs to be run on the target system, that then calls back to the Command and Control server for instructions[20].

An analysis of a payload that communicates over HTTP(S) show that it has the following properties (other protocols would have similar properties):

- Payload has a unique cryptographic hash
- Is focused at successful communication with Command and Control Server
- Uses a protocol for communication (e.g. HTTP(S) or DNS)
- Uses DNS Domain Lookups
- Has a profile that can be customized

Each used payload is unique, resulting in having a unique cryptographic hash. Hashes are used to identify data by mapping data of arbitrary size to data of fixed size. Security appliances use known hashes to detect known malware. Using a unique payload, and therefore unknown to security appliances, could prevent the payload from being detected and blocked.

The main purpose of a payload is to have successful communication with the Command and Control server to send data to, and receive instructions from. This communication makes use of a protocol to send data over that has to appear legitimate and be functional. Examples of well-used protocols are HTTP(S) and DNS.

DNS domain lookups are used to lookup the IP address of the domain to send the data to.

A profile allows the payload to be customized. A profile is used to tell the payload how (what protocol and how it must look like), when (date or time), and whom (domain or IP) to communicate with.

An example of customizable options for a payload that uses HTTP(S) are the following: domain/IP and port to callback to, TLS certificates to be used, time or data to callback at, and what headers fields to set and use like communication URLs and user-agents. Options are similar when other protocols are used.

4 Detecting Blue Team's Analysis

A Blue Team analysis can in theory be detected by monitoring the Red Team infrastructure for Com-

[2]From an OSI model perspective[3].
[3]Hidden from inspection due to end-to-end TLS encryption.
[4]Commercial Post-Exploitation Framework
[5]Free Post-Exploitation Framework

mand and Control (C2) communication anomalies. Anomalies can be triggered by Internet scan, or someone with inside knowledge of the payload used by the Red Team operation.

To differentiate between random Internet scans and the Blue Team's analysis, anomalies should only be triggered by events of which can be concluded that its source had inside knowledge of the payload used, and not events of which its source could be automated and randomly generated. Therefore, the Red Team must use unique communication characteristics that cannot be attributed to automated Internet scans, or that are used by other Red Team operations, to make differentiation possible between inside knowledge and Internet scans. Using these unique communication characteristics limit false positives, although preventing false positives at all is not possible due to the unknown factors of how random Internet scans are or will be operating.

An anomaly triggered by a Blue Team's analysis can be defined as the following:

> *An event that is not legit Command and Control communication, but has similarities to legit Command and Control communication that can only be known when the Red Team's payload has been analyzed.*

By using the findings from the analysis done in Section 3, a Blue Team's analysis can be detected by monitoring the infrastructure for the following anomaly characteristics:

- Communication Paths[6]
- User-Agent[6]
- Geo Location
- DNS Domain Lookup
- Virustotal Lookup

Depending on the payload, anomaly characteristics can change. The more characteristics are monitored, the higher the chance to detect a Blue Team's analysis. Although 100% detection ratio of a Blue Team's analysis is not possible due to the unknown factors on how the Blue Team is operating.

4.1 Communication Paths

Communication paths are used by the payload to communicate with the Command and Control server. The actual messaging between the payload and the Command and Control server is being put in a file that gets identified by using the communication path. Depending on the configuration of the payload, one or multiple communication paths can be used. An example of a payload's communication paths are the following:

- /legit/communication/uri/to/filter/with/get.php
- /legit/communication/uri/to/filter/with/news.php
- /legit/communication/uri/to/filter/with/login/process.php

When an operation has been spotted by the Blue Team, the Blue Team can analyze communication paths that has similarities to the paths used by the payload like:

- /legit/communication/uri/to/filter
- /legit/communication/uri/to/filter/index.html
- /legit/communication/uri/to/filter/with/login

It is recommended to use long and unique paths for the payload to prevent false positives generated by random Internet scans, and to increase the chance that one or multiple sub-paths get analyzed by the Blue Team.

An anomaly would be any communication path being accessed that is not used by the payload, but has similarities to it that can only be known when an analysis has been done.

4.2 User-Agent

The user-agent is part of a standard HTTP Header and is used when accessing any of the previous communication paths described in Section 4.1. This field is set by browsers and other web applications to let the server know what client/agent is used for accessing the file located at the communication path.

Most payloads can be configured to use a particular user-agent. For example: "Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko". If the communication paths are accessed by other clients than the payload, and the user-agent has not been set accordingly, the user-agent will be different than the one used by the payload. An example of a different user-agent is the following: "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36".

An anomaly would be access to a Command and Control communication path with a user-agent different

[6]Only when HTTP(S) is used, though with other protocols other header fields could be used instead.

from the one used by the payload. Keep in mind the client sets the user-agent, and therefore can be spoofed.

4.3 Geo Location

When any of the HTTP(S) communication paths are accessed, the server knows the source IP address to route the traffic back to. This source IP can be used to generate a potential geographical location. If a Red Team operation takes place in a particular country, it is expected to only have IPs communicating with the communication paths from or around this particular location.

For example, when a Red Team operation takes place in The Netherlands, it is unexpected to have IPs from Russia accessing the communication paths. However, false positives can occur with traveling employees, the use of Virtual Private Networks (VPNs), wrong mapping of IP to location by the Geo Location provider used, or the fact that IP addresses and ranges can change ownership and therefore its location. An anomaly would be any communication to communication paths from an unexpected location.

One could also look at IP addresses. An anomaly would be any communication from an IP address that was not first used for Command and Control communication. Though this could give some false positives as well due to traveling employees and the use of multiple uplinks (e.g. when loadbalancing, mobile, WAN, or satellite is used).

4.4 DNS Domain Lookup

DNS domain lookups are used by the payload, or more precisely the target's operating system, to lookup the IP address to send traffic to. An example of a domain name for a Red Team operation is the following: "rt-1.very.legit.domain.tours.prac.os3.nl". In the case of the Red Team operation being spotted by the Blue Team, the Blue Team could do an analysis (query) of any of the sub-domains for example: "legit.domain.tours.prac.os3.nl". An anomaly would be a lookup that does not match the domain, but match a particular prefix of that domain. An example of a prefix for the domain used by the operation is: "*.domain.tours.prac.os3.nl".

An anomaly shows that someone is resolving a subdomain that should not have been known, and therefore had inside knowledge of the full domain used. To decrease the chance of false positives by Internet scans, one should use unique and long sub domains that can not be generated or guessed. Though this could raise suspicion by the Blue Team in the first place.

4.5 Virustotal Lookup

Virustotal.com is a well-known and well-used website that uses multiple vendors to scan files for malware. One could upload a file, or search for one by using URLs, IP addresses, domains, or cryptographic hashes[35]. When a Red Team operation have been spotted, and the Blue Team has their hands on the used payload, it could have been uploaded to Virustotal for a malware scan. Therefore, one could check if the cryptographic hash of the payload is known to Virustotal or not[7]. When Virustotal knows the payload's hash, it does not necessarily mean that the Blue Team sent the file for analysis, as it could have been automated as well. Some security appliances and software upload incoming files automatically to scan for malware.

4.6 Post-Detecting Blue Team's Analysis

After a Blue Team's analysis has been detected, the Red Team can change its methods to become undetected again. The Red Team can move to another target system in the infiltrated infrastructure, change its payload's communication properties that are described in Section 3, or dispose and create a new Red Team infrastructure that is unknown to the Blue Team. The old Red Team infrastructure can be left up and running as a decoy, or be destroyed.

5 Proof of Concept

A Proof of Concept was created that had to match the following requirements:

- Able to detect a Blue Team's analysis of a Red Team's operation
- Usable for multiple Red Team operations
- Should not be triggered by random Internet scans

[7] Other websites can be searched for as well.

5.1 Infrastructure

Figure 3 shows the created infrastructure to detect a Blue Team's analysis.

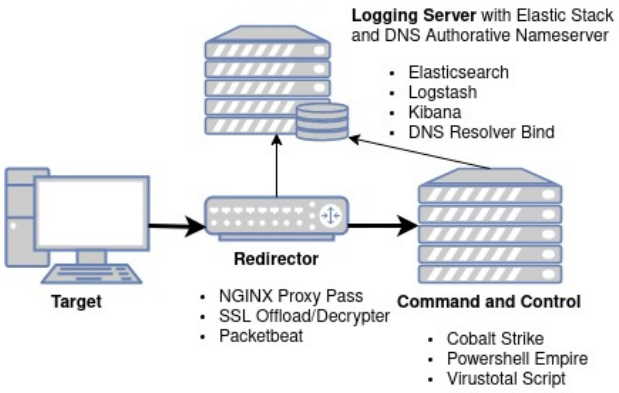

Figure 3: Red Team Logging Infrastructure

5.1.1 Logging Server

The logging server is running Elastic Stack version 6.1.1 with Bind version 9.10 as an authorative nameserver.

Elastic Stack is a suite consisting out of Elasticsearch, Logstash, and Kibana. The company Elastic describes the stack as the following:

> *"Elasticsearch is a search and analytics engine. Logstash is a serverside data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch"*[16]

Elasticsearch acts as the database to hold all logging. Logstash is used to filter and sort all data before it gets send to Elasticsearch and saved. The configuration that was used is the default configuration recommended by the Elastic Stack and Product Documentation[11].

Bind is an open source authorative DNS nameserver that is able to resolve DNS queries[6]. The logging server is running Bind as authoritative nameserver for the domain "tours.prac.os3.nl". The zone contains records for the redirectors at the domain "rt-1.very.legit.domain.tours.prac.os3.nl". Logstash is used to parse the query logging of Bind and then sends it to Elasticsearch after filtering. This implementation ensures that DNS domain lookup anomalies can be spotted as described in Section 4.4. The configuration file of Logstash can be seen in Appendix Figure 4, and the log output in Elasticsearch in Appendix Figure 5.

5.1.2 Redirector

The redirector is running NGINX version 1.10.3 as a reverse proxy and Packetbeat version 6.1.2.

NGINX is a web server that supports caching, load-balancing, proxying and more[25]. Running NGINX as a reverse proxy allows requests being passed away to another server without the requester being aware of this. In the Proof of Concept it pass requests to the Command and Control server and back[24]. This ensures that a target system never directly connects to the Command and Control server during an operation. In the Proof of Concept NGINX also decrypts SSL/TLS traffic before passing it to the Command and Control server. The NGINX configuration file can be seen in Appendix Figure 6.

Packetbeat is a network packet analyzer that is able to send data over TLS to Logstash or Elasticsearch. In the Proof of Concept, data is sent to Logstash as it provides the option to do additional filtering and altering of the data. Logstash sends the source IP from packets through a Geographical IP Lookup service providing the location of the IP address[13], and scans data for user-agents[15]. This implementation ensures that communication path, user-agent, and Geo location anomalies can be spotted as described in Section 4.1, 4.2, and 4.3. The Logstash configuration file used can be seen in Appendix Figure 7, and the log output in Elasticsearch in Appendix Figure 8 and 9.

5.1.3 Command and Control Server

The Command and Control server is running the Remote Access Tools Cobalt Strike and Powershell Empire. These tools also generate the used payloads for an operation, and are therefore stored on the filesystem. These payloads are monitored by a Virustotal Script.

Virustotal Script is a Python script running on the Command and Control server that generates a hash of the used payloads, and then sends it to Logstash on the logging server. Logstash uses the Virustotal API to gather information whether or not the hash has been scanned by Virustotal[36]. This implementation ensures that Virustotal anomalies can be spotted as described in Section 4.5. The Python source code can

be seen in Appendix Figure 12, the Logstash configuration file in Appendix Figure 10, and the log output in Elasticsearch in Appendix Figure 11.

5.2 Detection

With use of the created infrastructure, all logging is sent to Logstash first, and then to Elasticsearch. Using Kibana, Elasticsearch's Front End Visualization Tool, these logs can be searched, viewed, and visualized. Detecting a Blue Team's analysis can be done by only displaying logs that match particular criteria that represent an anomaly. Matching these particular criteria can be done by using filters. Filters are used to display logs that match a particular condition by using operators. The following operators are available: is, is not, is one of, is not one of, is between, is not between, exists, does not exist[12].

5.2.1 Communication Paths

The following filters are used to detect communication path anomalies:

Field	Operator	Value
Path	is not one of	Payload's $path^1$, $path^2$, $path^x$
Path	is	Payload's path prefix

Table 1: Filters for Communication Path Anomalies

5.2.2 User-Agent

The following filters are used to detect user-agent anomalies:

Field	Operator	Value
Path	is one of	Payload's $path^1$, $path^2$, $path^x$
User-agent	is not	Payload's user-agent

Table 2: Filters for User-Agent Anomalies

5.2.3 Geo Location

The following filters are used to detect geo location anomalies:

Field	Operator	Value
Path	is one of	Payload's $path^1$, $path^2$, $path^x$
Country_name	is not	Country of the operation

Table 3: Filters for Geo Location Anomalies

5.2.4 DNS Domain Lookup

The following filters are used to detect DNS domain lookup anomalies:

Field	Operator	Value
Query_resolver	is not	Domain used by operation
Query_resolver	is	Domain prefix used by operation

Table 4: Filters for DNS Domain Lookup Anomalies

5.2.5 Virustotal Lookup

The following filter is used to detect Virustotal (VT) anomalies:

Field	Operator	Value
Response_code	is not	0

Table 5: Filters for Virustotal Lookup Anomalies

A response code of 0 shows that the hash is not known by Virustotal, and therefore has not been scanned.

5.3 Python Query Script

The filters described in Section 5.2 can be used to detect a potential Blue Team's analysis. However, creating these filters requires the operational details of an operation and takes time duo to the fact that no API is available, especially with multiple ongoing Red Team operations. Even though these filters can be created by using the Domain Specific Language (DSL) language, a language used to query Elasticsearch, it still takes a lot of time to rewrite the filters to the DSL language, and requires knowledge of the DSL syntax[14]. Basically, all filters are DSL queries as Kibana rewrites the filters in the background to query Elasticsearch.

When the Red Team detects a Blue Team's analysis, the Red Team changes their methods to become undetected again. This results in a change of operational details which then results in the filters not being able to detect anomalies anymore. New filters have to be created to detect anomalies again. Getting the right information with the right filters can be complex with multiple ongoing Red Team operations. Furthermore, Elasticsearch or Kibana also dont have free and working alerting options.

Therefore, a Python Query script was created that is able to query Elasticsearch to search for anomalies without the need of creating filters first or having knowledge of the DSL search language. By not having to create these filters, anomalies can be searched for in a fast manner based on the operational details supplied as input. The script makes use of the high level Elasticsearch Python library named "elasticsearch-dsl" to query Elasticsearch. This library can query Elasticsearch using plain text or the DSL language[19]. The script requires the operational details of the operation as input, and rewrites these details to queries in the DSL language. It then outputs the found anomalies and its related logs. As the script is written in Python, it can easily be adapted to output to syslog, e-mail, or an API for social media messaging for alerting possibilities.

What input operational details can be supplied can be seen in Appendix Figure 13, and the output of the Python Query script in Appendix Figure 14[8].

5.4 Experiments

The following experiments were done to prove the detectability of the Python Query Script:

1. Access communication paths that have the same prefix/URI (and are therefore similiar) but are different than the ones used by the payload
2. Access communication paths with a different user-agent than the one used by the payload
3. Access communication paths from a different location other than the location of the C2 communication
4. DNS Domain Lookup of a sub-domain that has the same prefix (and are therefore similiar) but is different than the one used by the payload
5. Upload the payload to Virustotal

[8]The version on Github also shows what events triggered the anomaly[22]

For each experiment, the Red Team operation had the following characteristics:

- Communications paths used:
  - /legit/communication/uri/to/filter/with/get.php
  - /legit/communication/uri/to/filter/with/news.php
  - /legit/communication/uri/to/filter/with/login/process.php
- User-agent is Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
- Location maps to The Netherlands
- Domain is rt-1.very.legit.domain.tours.prac.os3.nl
- Hash of unique payload is unknown to Virustotal

In experiment 1, a communication path was accessed that has similarities to the one used by the payload; "/legit/communication/". Accessing this path resulted in an anomaly detected with the Python script.

In experiment 2, a communication path was accessed with a user-agent different than the one used by the payload; "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2486.0 Safari/537.36 Edge/13.10586". Using this user-agent resulted in an anomaly detected with the Python script.

In experiment 3, a communication path was accessed from a different location that the location expected from the operation; "Russia". Accessing the communication path from this location resulted in an anomaly detected with the Python script.

In experiment 4, a DNS domain lookup was done that has similarities to the one used by the payload; "legit.domain.tours.prac.os3.nl". This DNS domain lookup resulted in an anomaly detected with the Python script.

In experiment 5, the unique payload used by the Red Team operation was uploaded to Virustotal. This upload resulted in an anomaly detected with the Python script.

The results of above experiments can be seen in Figure 6.

Experiment	Detected
1 Communication path anomaly	Yes
2 User agent anomaly	Yes
3 Geo location anomaly	Yes
4 DNS domain lookup anomaly	Yes
5 Virustotal anomaly	Yes

Table 6: Experiments Anomaly Detection

6 Discussion

This research has the following discussion points:

- Elastic Stack
- Intrusion Detection Systems
- Kibana API
- Kibana Alerting
- Python Query Script

6.1 Elastic Stack

Elastic Stack has been used because it provides a free, open-source, and fast solution for saving logging information, while also providing means to search, manipulate and visualize this information[9]. Elastic Stack can also contain logging other than the logging from the Red Team infrastructure, resulting in having everything in one place. Yet, the Elastic Stack in the Proof of Concept showed challenges in flexibility with multiple Red Team operations and change in operational details. Therefore, the Python Query script was created.

6.2 Intrusion Detection Systems

Intrusion Detecton Systems (IDS) would not suffice for detecting a Blue Team's analysis. For example, a signature-based IDS like Snort does not allow rules like; if string contains word, but is not a complete match[5]. Using an IDS to detect the Blue Team would require large amounts of rules to find an anomaly. Even when an anomaly-based IDS is used, for example with Machine Learning, it is not intelligent enough to detect a Blue Teams analysis when multiple Red Team operations are taking place. This is not the case when a more flexible framework is used that allows dynamic rules/filtering like the Elastic Stack.

[9]Similar software like Zabbix or Solr might be sufficient as well.

6.3 Kibana API

There exists a community created API for Kibana that allows importing and exporting Dashboards. A dashboard consist of visualizations that are made of filters. Therefore, creating filters is still a time consuming task that needs to performed manually. One of the reasons why the Python Query script was created is to remove the time intensive task of creating filters. Unfortunately, it is not possible to use this API for creating filters and push them to Kibana, as it requires that the filters already exist[21].

6.4 Alerting

Elasticsearch and Kibana have no alerting possibilities that are freely available as stated in Section 5.3. There exist a paid plugin for Elasticsearch named X-Pack that contains alerting possibilities[10], and a free plugin named ElastAlert by Yelp[38]. ElastAlert deemed not sufficient because of its inconsistent documentation and its broken wildcard rule, and could therefore not be used[9]. Hence, the Python Query script was created that outputs to the console.

6.5 Python Query Script

As stated in Section 5.3, the Python Query script was created to allow searching Elasticsearch for anomalies without the need of creating time-intensive filters first. The script writes queries with the DSL language to detect anomalies based on the operational details that are given as input. Otherwise, filters would have to be created for each operation, and again after an operation changes its operational details. As the script is written in Python, it could easily be adapter to output to syslog, e-mail, or an API for social media messaging instead of the console.

7 Conclusion

A logging and monitored Red Team infrastructure can successfully be used to detect a Blue Team's analysis. This detection is typically based on anomaly detection and requires knowledge of the Red Team's operation and its used tools.

An analysis of the Remote Access Tools Cobalt Strike and Powershell Empire showed that these tools work by communicating between the payload at the infiltrated target, and the Command and Control server. This communication has particular characteristics which are useful for anomaly detection. An anomaly is communication that does not present legit Command and Control communication, yet show similarities to traffic that can only be known when an analysis has taken place.

The Proof of Concept successfully detects anomalies by monitoring the Command and Control communication and Virustotal. Setting up the monitoring for detection is a time extensive task due to the creation of suitable filters for each Red Team's operation. Especially due to the fact that these filters need to be altered when the communication characteristics of the Red Team operation changes.

Therefore, a Python Query script was created that solves these challenges by searching Elasticsearch in a fast and flexible manner, without the need of creating and maintaining these filters. This script provides a more dynamic search tool than Kibana to search for anomalies depending on the operational characteristics supplied as input.

7.1 Future Work

Recommendations for future work are the following:

- Kibana Alerting Plugin
- Python Query Script
- Learning an Operation

7.1.1 Kibana Alerting Plugin

A free alerting plugin can be a useful addition as it does not require any additional tooling or scripts like Python. Such a plugin would allow alerting possibilities based on created filters.

7.1.2 Python Query Script

Another recommendation for future work is further developing the Python Query script. The script is built as Proof of Concept and therefore not deemed production proof. The script could be extended to detect anomalies based on other communication characteristics, or its output could be improved.

7.1.3 Learning an Operation

Lastly, a tool that is able to learn an operation without the need of operational details as input. Learning a Red Team operation, or more precisely the Command and Control communication, could be done by using Machine Learning.

References

[1] *ATT&CK Matrix*. 2018. URL: https://attack.mitre.org/wiki/ATT%5C&CK_Matrix.

[2] Simeon Blatchley. *Hardening The Linux System*. 2016. URL: https://www.sans.org/media/score/checklists/LinuxCheatsheet_2.pdf.

[3] Neil Briscoe. "Understanding the OSI 7-layer model". In: *PC Network Advisor* 120.2 (2000).

[4] Colin Chisholm. *Boiling the Ocean: Security Operations and Log Analysis*. 2016. URL: https://www.sans.org/reading-room/whitepapers/logging/boiling-ocean-security-operations-log-analysis-36867.

[5] O'Reilly Commons. *Snort Cookbook/Rules and Signatures*. URL: http://commons.oreilly.com/wiki/index.php/Snort_Cookbook/Rules_and_Signatures.

[6] Internet Systems Consortium. *Bind*. URL: https://www.isc.org/downloads/bind/.

[7] Roberto Di Pietro and Luigi V Mancini. *Intrusion detection systems*. Vol. 38. Springer Science & Business Media, 2008.

[8] Jeff Dimmock. *Wiki to collect Red Team infrastructure hardening resources*. URL: https://github.com/bluscreenofjeff/Red-Team-Infrastructure-Wiki.

[9] ElastAlert. *Writing Filters for Rules*. URL: http://elastalert.readthedocs.io/en/latest/recipes/writing_filters.html.

[10] Elastic. *Alerting*. URL: https://www.elastic.co/products/x-pack/alerting.

[11] Elastic. *Elastic Stack and Product Documentation*. URL: https://www.elastic.co/guide/index.html.

[12] Elastic. *Filtering by Field*. URL: https://www.elastic.co/guide/en/kibana/current/field-filter.html#field-filter.

[13] Elastic. *Geoip filter plugin*. Version 5.0.3. 2017. URL: https://www.elastic.co/guide/en/logstash/current/plugins-filters-geoip.html.

[14] Elastic. *Query DSL*. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html#query-dsl.

[15] Elastic. *Useragent filter plugin*. Version 3.2.2. 2017. URL: https://www.elastic.co/guide/en/logstash/current/plugins-filters-useragent.html.

[16] Elastic. *What is the ELK Stack?* URL: https://www.elastic.co/elk-stack.

[17] David Fifield et al. "Blocking-resistant communication through domain fronting". In: *Proceedings on Privacy Enhancing Technologies* 2015.2 (2015), pp. 46–64.

[18] Seham Mohamed GadAllah. *The Importance of Logging and Traffic Monitoring for Information Security*. 2003. URL: https://www.sans.org/reading-room/whitepapers/logging/importance-logging-traffic-monitoring-information-security-1379.

[19] Github. *Elasticsearch DSL*. URL: https://github.com/elastic/elasticsearch-dsl-py.

[20] InfoSec Institute. *Remote Access Tool*. 2014. URL: http://resources.infosecinstitute.com/remote-access-tool/.

[21] *Kibana-API is an extension to Kibana that lets you tap in to the dashboard management board from your app and change the visualizations dynamically*. URL: https://github.com/Webiks/kibana-API.

[22] Rick Lahaye. *Python script to detect anomalies in Elasticsearch depending on the given Red Team operational details*. URL: https://github.com/ricklahaye/es-rt-anomalies.

[23] T.A. Limoncelli, C.J. Hogan, and S.R. Chalup. *The Practice of System and Network Administration*. v. 1. Addison Wesley, 2016. ISBN: 9780321919168. URL: https://books.google.nl/books?id=mWy2jwEACAAJ.

[24] NGINX. *NGINX Reverse Proxy*. URL: https://www.nginx.com/resources/admin-guide/reverse-proxy/.

[25] NGINX. *What is NGINX*. URL: https://www.nginx.com/resources/glossary/nginx/.

[26] "Operations Security (OPSEC)". In: (). URL: https://www.dodea.edu/offices/safety/opsec.cfm.

[27] Pete. *Blue Team Fundamentals*. 2017. URL: https://securitybytes.io/blue-team-fundamentals-4ee226368b7b.

[28] Krishna PMV. *Domain Fronting - A technique used to circumvent internet censoring*. 2017. URL: https://medium.com/%5C@pmvk/domain-fronting-a-technique-used-to-circumvent-internet-censoring-10ef1bb3db84.

[29] *Red Teaming Software*. 2017. URL: https://attack.mitre.org/wiki/Software.

[30] Alexander Rymdeko-Harvey. "6 Red Team Infrastructure Tips". In: (2016). URL: https://cybersyndicates.com/2016/11/top-red-team-tips/.

[31] R. Sandhu. "Role-based access control models". In: 29.2 (1996), pp. 38–47. ISSN: 0018-9162.

[32] Marc Smeets. *Red Team Security Controls*. Interview. 2018.

[33] Jun Sun, Punit Ahluwalia, and Kai S. Koong. "The more secure the better? A study of information security readiness". In: *Industrial Management & Data Systems* 111.4 (2011), pp. 570–588. DOI: 10.1108/02635571111133551.

[34] Vectra. *Five ways cybercriminals conceal command-and-control communications*. URL: https://vectra.ai/assets/cybercriminals-conceal-command-and-control.pdf.

[35] Virustotal. *Virustotal*. URL: https://www.virustotal.com/#/home/search.

[36] *Virustotal Lookup filter for Logstash*. URL: https://github.com/coolacid/logstash-filter-virustotal.

[37] Hazen Weber. *Role-Based Access Control: The NIST Solution*. 2003. URL: https://www.sans.org/reading-room/whitepapers/sysadmin/role-based-access-control-nist-solution-1270.

[38] Yelp. *Easy & Flexible Alerting With ElasticSearch*. URL: https://github.com/Yelp/elastalert.

Appendix

Bind

```
input {
  file {
    path => [ "/var/log/named-query.log" ]
    start_position => "beginning"
    type => "dns"
  }
}
filter {
  if [type] == "dns" {
  grok {
    match => { "message" => "%{MONTHDAY:day}-%{MONTH:month}-%{YEAR:year} %{TIME:time}
        queries: info: client %{IP:client_ip}\#%{DATA:client_port} \(%{DATA:query_record}\):
         query: %{DATA:query_record2} %{DATA:dns_type} %{DATA:query_type} %{DATA:query_flags
        } \(%{IP:query_resolver}\)" }
  }
  geoip {
    source => "[client_ip]"
  }
    mutate {
      add_field => { "timestamp" => "%{day}-%{month}-%{year} %{time}" }
    }
    date {
      match => [ "timestamp", "dd-MMM-YYYY HH:mm:ss.SSS" ]
    }
  }
}
output {
  elasticsearch {
    hosts => ["10.0.0.1:9200"]
    manage_template => false
    index => "packetbeat-%{+YYYY.MM.dd}"
  }
}
```

Figure 4: Logstash Configuration for Bind Logging

```
{
  "_index": "packetbeat-2018.02.18",
  "_type": "doc",
  "_id": "NMswq2EBFkNXLp60AncU",
  "_version": 1,
  "_score": null,
  "_source": {
    "query_resolver": "145.100.104.125",
    "year": "2018",
    "query_type": "A",
    "day": "19",
    "query_flags": "-EDC",
    "time": "00:11:45.092",
    "query_record": "rt-1.very.legit.domain.tours.prac.os3.nl",
    "client_port": "48848",
    "timestamp": "19-Feb-2018 00:11:45.092",
    "client_ip": "173.194.170.75",
    "query_record2": "rt-1.very.legit.domain.tours.prac.os3.nl",
    "path": "/var/log/named-query.log",
    "tags": "",
    "message": "19-Feb-2018 00:11:45.092 queries: info: client 173.194.170.75#48848 (rt-1.
        very.legit.domain.tours.prac.os3.nl): query: rt-1.very.legit.domain.tours.prac.os3.
        nl IN A -EDC (145.100.104.125)",
    "@timestamp": "2018-02-18T23:11:45.092Z",
    "geoip": {
      "ip": "173.194.170.75",
      "country_name": "Netherlands",
      "country_code3": "NL",
      "continent_code": "EU",
      "location": {
        "lon": 4.8995,
        "lat": 52.3824
      },
      "latitude": 52.3824,
      "timezone": "Europe/Amsterdam",
      "country_code2": "NL",
      "longitude": 4.8995
    },
    "@version": "1",
    "host": "tours",
    "dns_type": "IN",
    "type": "dns",
    "month": "Feb"
  }
}
```


Figure 5: Elasticsearch Log Output for Bind

NGINX

```
server {
  listen       443 ssl;
  server_name  rt-1.very.legit.domain.tours.prac.os3.nl;
  ssl_certificate /etc/letsencrypt/live/rt-1.very.legit.domain.tours.prac.os3.nl/fullchain.
      pem;
  ssl_certificate_key /etc/letsencrypt/live/rt-1.very.legit.domain.tours.prac.os3.nl/privkey
      .pem;
  location / {
    root    html;
    index  index.html index.htm;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_pass http://145.100.111.131;
  }
}
```

Figure 6: NGINX Revere Proxy Configuration

Packetbeat

```
input {
  beats {
    port => 5044
    host => "10.0.0.1"
    ssl => true
    ssl_certificate => "/etc/pki/tls/certs/logstash-forwarder.crt"
    ssl_key => "/etc/pki/tls/private/logstash-forwarder.key"
  }
}
filter {
  if [type] == "http" {
    mutate { copy => { "[http][request][headers][x-real-ip]" => "client_ip" }
  }
  geoip { source => "[client_ip]" }
  useragent {
    source => "[http][request][headers][user-agent]"
    target => "[access][agent]"
  }
}
output {
  elasticsearch {
    hosts => ["10.0.0.1:9200"]
    manage_template => false
    index => "packetbeat-%{+YYYY.MM.dd}"
  }
}
```

Figure 7: Logstash Configuration for Packetbeat

```
{
  "_index": "packetbeat-2018.01.29",
  "_type": "doc",
  "_id": "XcqgQmEBFkNXLp60qk9z",
  "_version": 1,
  "_score": null,
  "_source": {
    "proc": "",
    "query": "GET /legit/communication/uri/to/filter/width/news.php",
    "beat": {
      "name": "redirect",
      "hostname": "redirect",
      "version": "6.1.2"
    },
    "ip": "145.100.111.130",
    "path": "/legit/communication/uri/to/filter/width/news.php",
    "method": "GET",
    "real_ip": "145.100.102.52",
    "@timestamp": "2018-01-29T15:54:40.294Z",
    "client_proc": "",
    "host": "redirect",
    "port": 80,
    "client_server": "",
    "direction": "in",
    "status": "OK",
    "geoip": {
      "country_code2": "NL",
      "country_code3": "NL",
      "ip": "145.100.102.52",
      "city_name": "Amsterdam",
      "timezone": "Europe/Amsterdam",
      "region_name": "North Holland",
      "postal_code": "1091",
      "country_name": "Netherlands",
      "continent_code": "EU",
      "latitude": 52.35,
      "longitude": 4.9167,
      "region_code": "NH",
      "location": {
        "lon": 4.9167,
        "lat": 52.35
      }
    },
    "client_port": 50850,
    "tags": ""
    "server": "",
    "responsetime": 7,
    "client_ip": "145.100.102.52",
    "type": "http",
    "@version": "1",
```


Figure 8: Elasticsearch Log Output Part (1/2)

```
    "http": {
      "response": {
        "headers": {
          "server": "Microsoft-IIS/7.5",
          "pragma": "no-cache",
          "date": "Mon, 29 Jan 2018 15:54:40 GMT",
          "content-type": "text/html; charset=utf-8",
          "content-length": 1241,
          "cache-control": "no-cache, no-store, must-revalidate",
          "expires": "0"
        },
        "phrase": "OK",
        "code": 200
      },
      "request": {
        "params": "",
        "headers": {
          "cookie": "session=bbXiBbbYS2lzfHCBbZaS+KYGeo0=",
          "host": "145.100.111.130",
          "connection": "close",
          "content-length": 0,
          "x-real-ip": "145.100.102.52",
          "user-agent": "Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like
              Gecko"
        }
      }
    },
    "bytes_out": 1468,
    "bytes_in": 263,
    "access": {
      "agent": {
        "major": "11",
        "name": "IE",
        "device": "Other",
        "build": "",
        "os": "Windows 7",
        "os_name": "Windows 7",
        "minor": "0"
      }
    }
  },
  "fields": {
    "@timestamp": [
      "2018-01-29T15:54:40.294Z"
    ]
  },
}
```


Figure 9: Elasticsearch Log Output Part (2/2)

Virustotal

```
input {
  tcp {
    host => "10.0.0.1"
    port => 9563
    codec => json
  }
}
filter{
  mutate {
    add_tag => [ "md5" ]
  }
  virustotal {
    apikey => "X"
  elasticsearch {
    hosts => ["10.0.0.1:9200"]
    manage_template => false
    index => "packetbeat-%{+YYYY.MM.dd}"
  }
}
```

Figure 10: Logstash Configuration for Virustotal


```
{
  "_index": "packetbeat-2018.01.30",
  "_type": "doc",
  "_id": "usqQRmEBFkNXLp60UlZs",
  "_version": 1,
  "_score": null,
  "_source": {
    "@version": "1",
    "virustotal": {
      "verbose_msg": "The requested resource is not among the finished, queued or pending
          scans",
      "resource": "21946da169869cc9033cee0b017a8dcf",
      "response_code": 0
    },
    "host": "teamserver",
    "@metdata": {
      "ip_address": "145.100.111.131"
    },
    "@timestamp": "2018-01-30T10:15:18.600Z",
    "file": "/tmp/unique_malware.bat",
    "tags": [
      "md5"
    ],
    "md5": "21946da169869cc9033cee0b017a8dcf",
    "port": 37450
  },
  "fields": {
    "@timestamp": [
      "2018-01-30T10:15:18.600Z"
    ]
  }
}
```


Figure 11: Elastisearch Log Output for Virustotal

```
#!/usr/bin/python
import hashlib
import threading
import socket
import json
import sys

HOST = '10.0.0.1'
PORT = 9563

file_name = "/tmp/angry_payload.bat"
with open(file_name) as file_to_check:
  data = file_to_check.read()
  md5 = hashlib.md5(data).hexdigest()

msg = "{{\"md5\": \"{}\", \"file\": \"{}\"}}\n".format(md5, file_name)

try:
  sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
except socket.error, msg:
  sys.stderr.write("[ERROR] %s\n" % msg[1])
  sys.exit(1)
try:
  sock.connect((HOST, PORT))
except socket.error, msg:
  sys.stderr.write("[ERROR] %s\n" % msg[1])
  sys.exit(2)

def send_to_logstash():
  threading.Timer(300.0, send_to_logstash).start()
  sock.send(msg)

send_to_logstash()
```

Figure 12: Python Virustotal Script to Send Hash to Logstash

Python Query Script

```
Usage: query.py [options]

Options:
  -h, --help            show this help message and exit
  --host=HOST           ES host [10.0.0.1]
  --port=PORT           ES port [9200]
  --index=INDEX         ES index [packetbeat-*]
  --refresh=REFRESH     Refresh every x seconds [60]
  --paths=PATH          C2 paths [/legit/communication/uri/to/filter/width/get
                        .php,/legit/communication/uri/to/filter/width/news.php
                        ,/legit/communication/uri/to/filter/width/login/proces
                        s.php]
  --path-prefix=PATH_PREFIX
                        C2 path prefix [/legit/*]
  --user_agent=USER_AGENT
                        C2 user agent [Mozilla/5.0 (Windows NT 6.1; WOW64;
                        Trident/7.0; rv:11.0) like Gecko]
  --geo-country=GEO     C2 country [Netherlands]
  --dns=DNS             C2 dns host name
                        [rt-1.very.legit.domain.tours.prac.os3.nl]
  --dns-prefix=DNS_PREFIX
                        C2 dns host name prefix root
                        [*.domain.tours.prac.os3.nl]
```

Figure 13: Python Query Script Input Options[22]

```
<bound method Elasticsearch.info of <Elasticsearch([{'host': '10.0.0.1', 'port': '9200'}])>>

* User Agent
- Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
- Paths: ['/legit/communication/uri/to/filter/with/get.php', '/legit/communication/uri/to/
    filter/with/news.php', '/legit/communication/uri/to/filter/with/login/process.php']
- Status: 6 anomalies

* Geo Location
- Geo Country: Netherlands
- Paths: ['/legit/communication/uri/to/filter/with/get.php', '/legit/communication/uri/to/
    filter/with/news.php', '/legit/communication/uri/to/filter/with/login/process.php']
- Status: No anomalies

* Virustotal
- Status: 12 anomalies

* Communication Paths
- Paths: ['/legit/communication/uri/to/filter/with/get.php', '/legit/communication/uri/to/
    filter/with/news.php', '/legit/communication/uri/to/filter/with/login/process.php']
- Status: 42 anomalies

* DNS
- DNS: rt-1.very.legit.domain.tours.prac.os3.nl
- DNS Prefix: *.domain.tours.prac.os3.nl
- Status: 47 anomalies
```

Figure 14: Python Query Script Output[22]