Global Information Assurance Certification Paper

Do random IP lookups mean anything?

GIAC (GCIA) Gold Certification

Author: Jay Yaneza, jay_yaneza@trendmicro.com
Advisor: Chris Walker, CISSP, CCISO, GSEC, GCED, GWEB
Accepted: April 30, 2018

Abstract

Being able to identify the external IP address of a network is usually a benign activity. Applications may opt to use online services via an HTTP request or API call. Currently, there are some web-based applications that provide this kind of service openly, and some with possibly malicious uses. In fact, malware threats have been using these services to map out and identify their targets for quite some time to already – an acknowledged fact hidden in technical write-ups but which hold little recognition for an active defender. The goal of looking into these web services is to isolate threats that had abused the network service and identify this kind of network activity. If we can associate an external IP lookup to a suspicious activity, then we would be able to assume that an endpoint requires some form of investigation. Endpoint identification through IP addresses may pose a challenge, but the correct placement of the identification methods proposed in this paper may be considered. This paper will also look into the associated malicious activity that had used online services, the use of such services over time, differentiate the threats that use them, and finally how to detect them using open source tools, if applicable.

1. Introduction

In the early 2000's the IPv4 space was in short supply, and IPv6 was still in its infancy. Networks were rapidly expanding at an astronomical rate and, with that, the lack of IP address space. Because of this, RFC 1918 was proposed and, this lead to network segmentation.

The idea behind RFC 1918 was to separate a section of the IPv4 address space and call it "local" and the rest to be publicly accessible – the internet. With this implementation, "local" IP addresses were non-routable within the "internet". This notion of network segmentation also was the start of network address translation (NAT), where routable public IP traffic encapsulates non-routable local IP addresses.

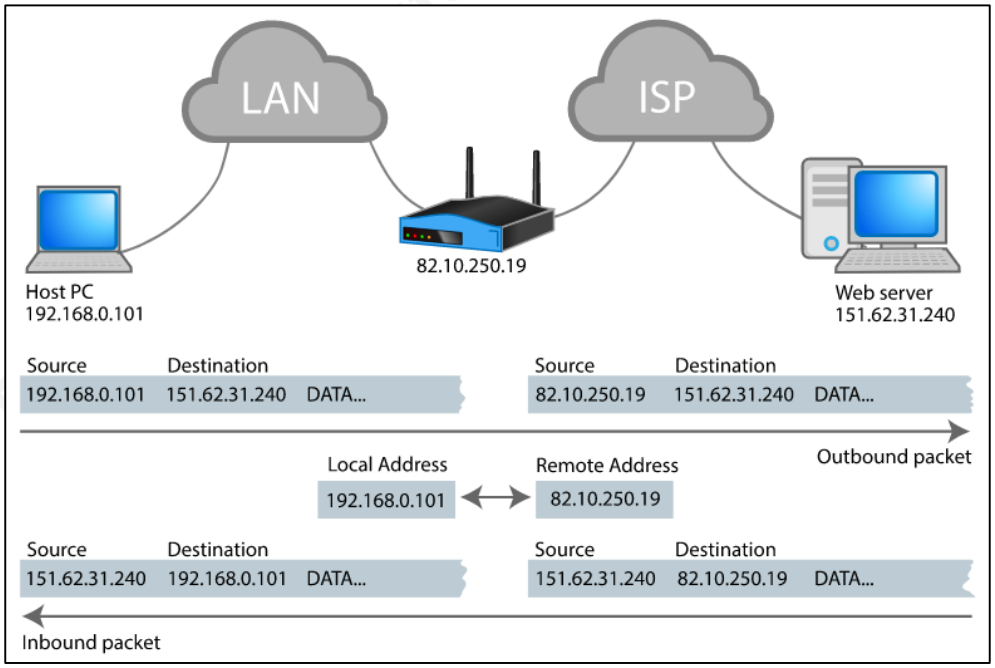


Figure 1: Network Address Translation, By Milesjpool, [CC BY-SA 4.0], from Wikimedia Commons

To the internet service receiving the traffic, it would seem that the request would be coming from the NAT-ted IP address and would be oblivious its origin. The traffic may come from an endpoint within an RFC 1819 network within the intranet (LAN), passing through two firewalls, ending up on the border gateway to be routed to the receiving internet service, say, a website out on the public internet. The receiving web server would be relatively unaware of the web traffic's history – all it knows is that the

web request needs a reply. This client-server transaction describes how most internet traffic occurs.

As for the method to ascertain the internet-facing IP address, it would entirely depend on the client or the server application.

1.1. Server applications

Internet-facing server-side applications would only need to listen for incoming requests and wait for the connection to happen – for example, a web server.

A web server application responding to an HTTP request needs to verify the connecting IP address (be it IPv4 or IPv6) and the client port (i.e., the client socket) that the request had used to initiate so that the web server can respond. The HTTP client (web browser) will then respond to the receiving server, represented by additional HTTP request headers as defined in RFC 2616:

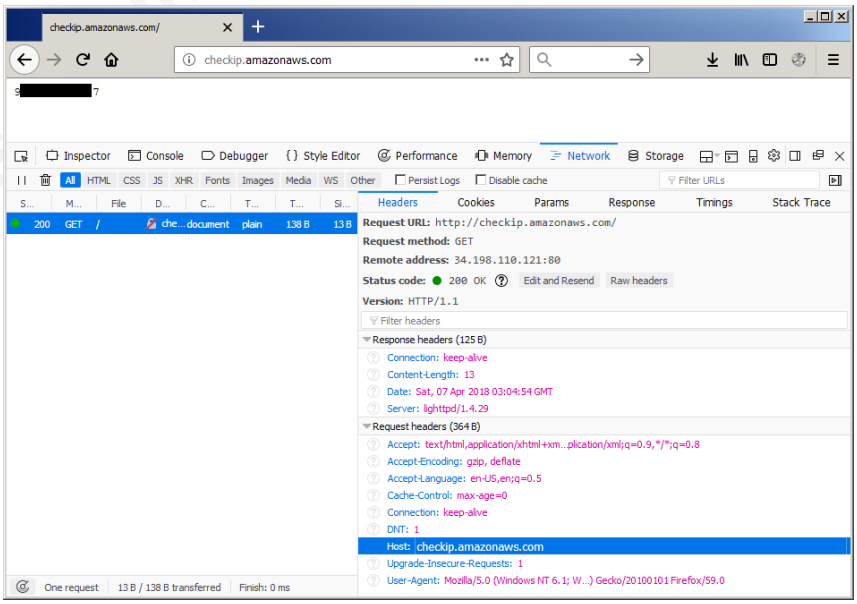


Figure 2: A simple web browser lookup for an external IP address

The web server can already use all this gathered information even before responding to the requesting HTTP client, mostly in two ways:

1. Restricting access (i.e., access control list) based on IP address, or preparing a custom response based on the HTTP request headers (e.g., Accept-Language,

User-Agent, Referrer, etc.). For example, a website can redirect HTTP requests if connections come from a specific range of IP Addresses:

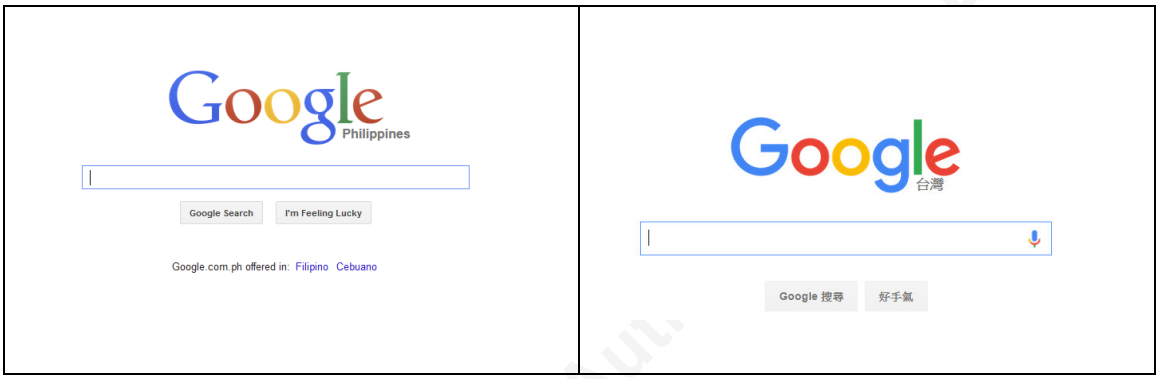


Figure 3: Differences of going to http://www.google.com, depending on location

The example above would be familiar to those who often travel: Google would redirect requests to their search engine (https://www.google.com) to a country-specific search page. Called "URL geo redirection", redirection of the HTTP request to another URL happens based on its origin. This process happens transparently, as seen below with the request going to https://google.com:

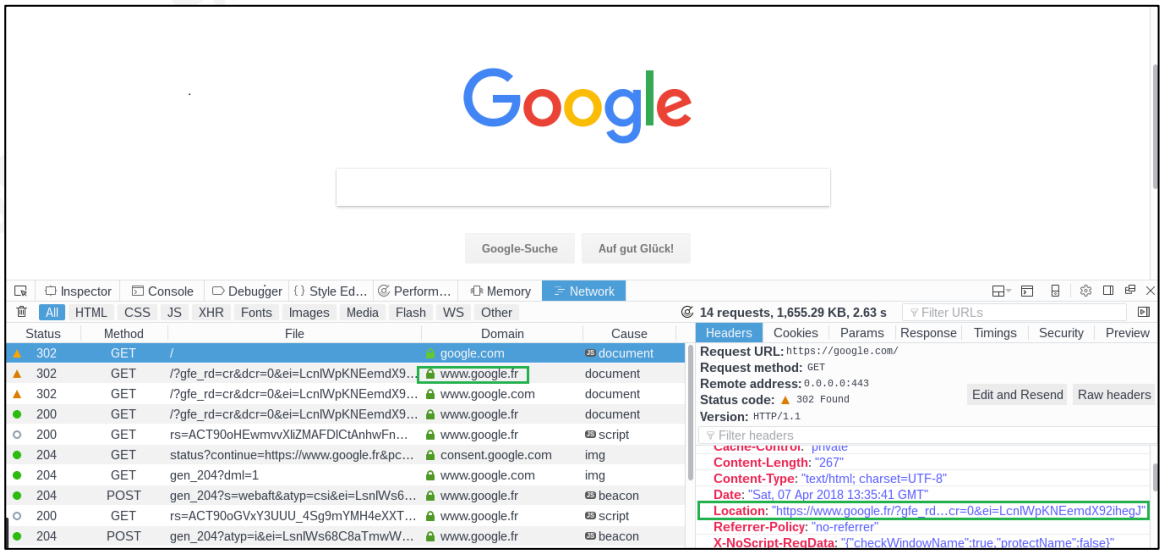


Figure 4: HTTP 302 redirect going to the initial http://www.google.com

In this example, the originating IP address is in France and the redirection corresponds to the Google landing page for France (https://www.google.fr).

Jay Yaneza, jay_yaneza@trendmicro.com

Of course, this means that the receiving web server would have checked the incoming HTTP request of its associated location and redirected to the appropriate web resource.

2. The information can also be gathered and stored for future use, and webmasters or website owners may use the information builds visitor profile. For example, identifying which countries their site attracts visitors, which operating systems (e.g., Mac OSX, Linux, Windows), or which platforms (e.g., desktop, phone, tablet), which is quite common for website analytics software:

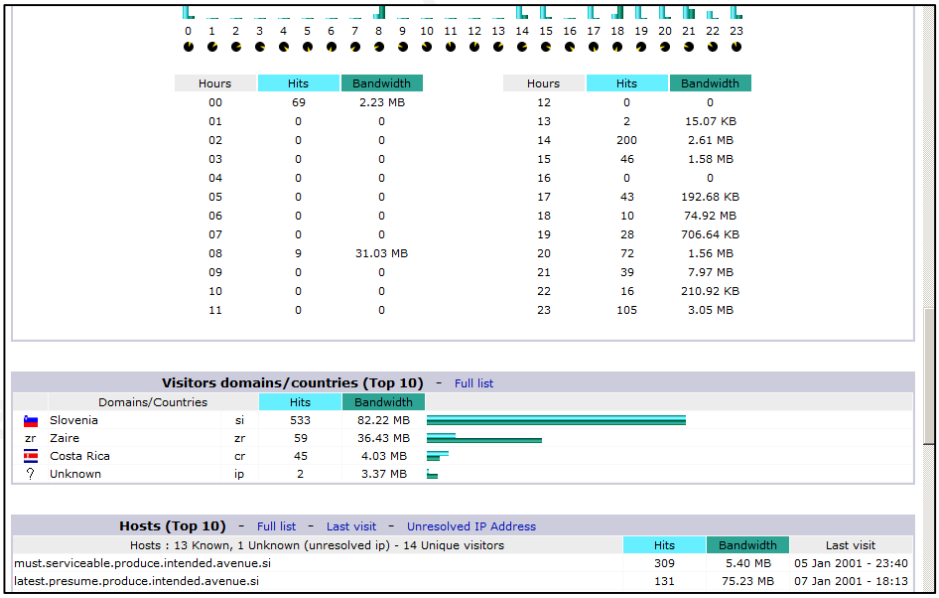


Figure 5: a website analytics program showing details about top visitors

The example above is from a software called **AWStats** (https://www.awstats.org/), an open-source web analytics tool that can generate reports for visitors, pages visits, search engines and the keywords used to find your site, broken links, etc. Therefore, remember that visiting a website (or any service, for that matter) does generate internet traffic, and the web server can gather and store this information for owners' purposes.

1.2. Client-side applications

On the other side, client applications have no direct way to discover their internet-facing IP address. Should the software were to check the IP address of the host where it is

currently running (e.g., calling the GetAdaptersInfo function in Microsoft Windows), it may acquire an RFC 1918 IP address as the endpoint is located behind a NAT-ted network. The most straightforward manner to determine the internet-facing IP address of a network would be to use an external service to do that determination. An often valid and useful case for applications wanting to know the internet-facing IP address of an endpoint would be for hosting a game server, such as Minecraft (Official Minecraft Wiki, n.d.).

However, the reasons why client-side applications may want to determine the internet-facing IP address would vary on the purpose of the software. On the other hand, we can compare this use case to how malware command-and-control (sometimes abbreviated as C2, C&C or CnC) would assist security threats in enabling a malicious purpose. Malware command-and-control server/s are controlled by threat actors that can either be used to send commands to compromised systems, or could be used to receive stolen data from a target network. Let us compare the functionality in a table:

Example	**Server (hosting a service)**	**Client-side application**
Everyday use	Game server	Game (installed on an endpoint)
Malicious	Malware C&C	Malware (on an infected endpoint)

For the most part, the general security interest would be the prevention (or blocking) of malware reaching out to their respective command-and-control servers. Yet malware, being just client-side applications, can change behavior based on information it can gather about the endpoint. In fact, malware does not even have to reach out to its command-and-control server to identify this basic information.

In the next parts of the paper, we would look into how easy it is for client-side applications to identify the environment where it is running.

2. Making use of the host-based IP address

Both built-in and optionally installed command-line utilities can acquire details about a host. Let us go through a simple example that uses a 3rd party service:

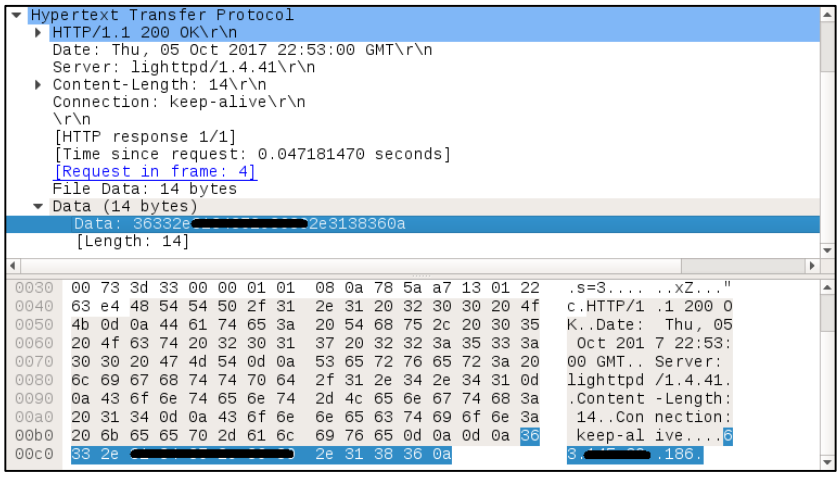
1. First, we can acquire the host IP address by command like utilities like 'ip' or 'ifconfig':

```
$ ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP group default qlen 1000
        link/ether 28:d2:44:xx:xx:xx brd ff:ff:ff:ff:ff:ff
        inet 10.10.xx.xx/23 brd 10.10.xx.255 scope global dynamic eth0
               valid_lft 249295sec preferred_lft 249295sec
        inet6 fe80::2ad2:44ff:fe5b:166e/64 scope link
               valid_lft forever preferred_lft forever
```

2. Next, we can use curl and send a request to a 3rd party site to get our public IP address:

```
$ curl -s checkip.amazonaws.com
63.xxx.xxx.186
```

The curl tool does a simple HTTP GET to the 3rd party site, and the 3rd party site just responds back with the public IP address it sees connecting to its service. This translates to a straightforward HTTP transaction:



3. Finally, we can get more details using a tool such as *geoiplookup* to determine the location of the IP address:

```
$ curl -s checkip.amazonaws.com | xargs geoiplookup
GeoIP Country Edition: US, United States
GeoIP City Edition, Rev 1: US, TX, Texas, Irving, 75062, xx.xxxxxx, -
xx.xxxxxx, xxx, xxx
GeoIP Organization Edition: Century Link
GeoIP ASNum Edition: AS209 Qwest Communications Company, LLC
```

The result of the *geoiplookup* tool uses the MaxMind geoip database, and is not without some inaccuracy. There are a lot of dependencies to accurately locate an IP address' location like the ISP where the endpoint is connected, networking variables like if they are going through a VPN or a proxy, the type of connection (broadband or cellular), and the effect of the combination of these variables even varies per country. In fact, MaxMind explicitly states that there is indeed a margin of error when using their service and they have these results public. Be it inaccurate as it may, a location range with the accuracy of somewhere between 25 km – 50 km (12.34 mi – 31.07 mi) is close.

Even without the benefit of the output of *geopiplookup*, the knowledge of the public IP can already have some effect on the programming logic of the client-side application:. Most likely, a program that does pre-execution determination would have the logic stored in the program binary. The program that does post-execution determination would have to rely on further communication with a back-end service (e.g., a website that is owned or operated publisher of the program, or a malware command-and-control site). For example:

- A program that does a pre-execution determination can check if the endpoint where it is running is in the list of its "targets" – similar to a server-side access control list (ACL).
- On the other hand, a program that does post-execution determination of the public IP address can store this information and make use of it later (e.g., use it to profile the endpoint).

Jay Yaneza, jay_yaneza@trendmicro.com

As mentioned previously, client-side applications would most likely use a 3rd party service to determine the public IP address of the environment where it is executing. In the case of malware, this trend may have started around 2014 as researchers and analysts started to write about this in their technical reports:

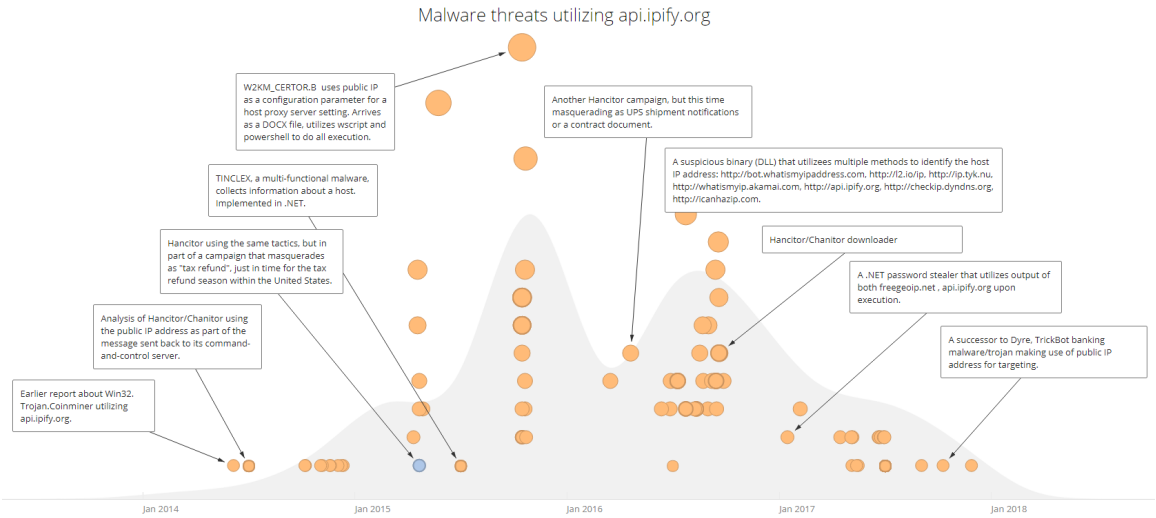


Figure 6: Documented reports malware threats using api.ipify.org, rendered via Recorded Future

The previous timeline only shows the known usage of api.ipify[.]org over time , but there are other sites that provide similar services. We would be looking into the following 3rd party services as well:

Service	**Example**
api.ipify[.]org	$ curl 'https://api.ipify[.]org?format=json'
wtfismyip[.]com	$ curl "http://wtfismyip[.]com/json"
checkip.amazonaws[.]com	$ curl "http://checkip.amazonaws[.]com"
ipecho[.]net	$ curl "http://ipecho[.]net/plain"
icanhazip[.]com	$ curl "http://icanhazip[.]com"
ipinfo[.]io	$ curl "http://ipinfo[.]io"

Aside from these above, there are yet others like ipgeoapi[.]com, myexternalip[.]com, ip.anysrc[.]net, www.myipaddress[.]com. To identify if 3rd party services like these were ever utilized in conjunction with malicious activity, malware sample sets collected between January 2017 to December 2017 were sandboxed. We would talk more about this in the next section.

Jay Yaneza, jay_yaneza@trendmicro.com

3. Location-aware threats

Reflecting on positively malicious artifacts, like malware, allows us to have insight into the usage of these services. Based on the sample set we had identified and tested (around 7747 unique file hashes), we have identified the most popular IP lookup site to be icanhazip[.]com:

IP Lookup site	Detection count
icanhazip[.]com	2343
myexternalip[.]com	2266
www.myipaddress[.]com	1786
ipinfo[.]io	1013
ipecho[.]net	919
wtfismyip[.]com	496
ipgeoapi[.]com	323
api.ipify[.]org	205
checkip.amazonaws[.]com	32
whatismyipaddress[.]com	16
www.showipaddress[.]com	16
checkip.dyndns[.]com	16
ip.anysrc[.]net	12
myexteranlip[.]com	3
wftismyip[.]com	1
Grand Total	**9447**

The results above have merged individual results of myexternalip[.]com and www[.]myexternalip[.]com as they shared the same IP address. Noticeable in this list are the last two, myexteranlip[.]com and wftismyip[.]com, which were mistyped spellings of myexternalip[.]com and wtfismyip[.]com respectively. Not surprisingly, they are of the same malware family (TeslaCrypt ransomware) that closed-down operations around May 2016 (BleepingComputer.com, May 2016).

Now, looking at the most frequented domain, icanhazip[.]com, we can see that there was just one significant malware family that used this domain - UPATRE:

Malware Family	Detection count
UPATRE	2202
UNDETERMINED	128

Jay Yaneza, jay_yaneza@trendmicro.com

MEWSPY	4
DLOADER	2
TRICKBOT	1
DROPPR	1
HANCITOR	1
DYRE	1
SWISYN	1
KADENA	1
Grand Total	**2342**

As a refresher, UPATRE is a known malware family that is a frequent payload of malicious spam campaigns. The primary function of UPATRE is to be the downloader component of a secondary infection, and this has changed over time. For example, the combination of UPATRE and DRYRE (DYREZA) was a favorite around 2015 (ISC InfoSec Handlers Diary Blog, 2015 May), but the secondary infection resulting from the initial UPATRE infection has always changed over time such banking trojans or ZBOT (Comodo, 2017 July).

The resulting lookups have a larger number as compared to the number of file hashes evaluated. This means that some binaries used several services to identify the external IP address, with some using up to seven (7) methods. After comparing the malware types, the ransomware threat type had most of these types of multiple external IP lookups:

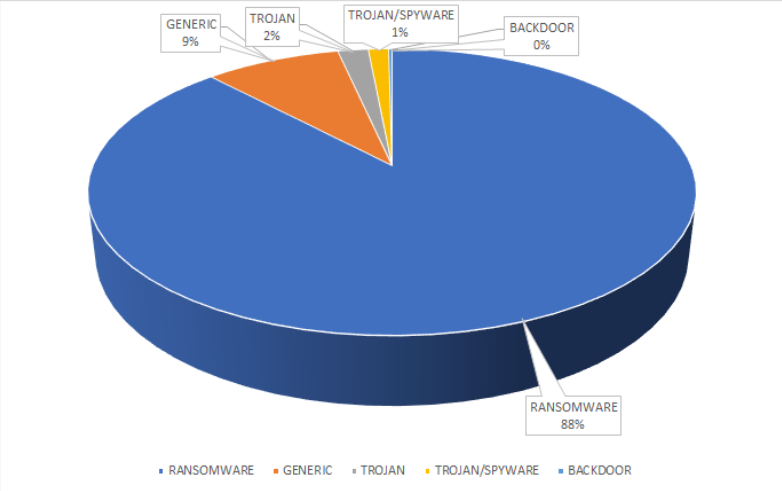


Figure 7: Proportions of malware families using multiple IP lookup methods

Jay Yaneza, jay_yaneza@trendmicro.com

This makes sense since the ransomware threat are location-based threats, with specific targeting mechanisms, payment methods and ransom note instructions depending on the location, or maybe language, of the affected endpoint (Sophos, 2016 May).

While there is no malicious intent in being able to determine the external IP address of an environment, there is likewise very little legitimate use for a normal user to do so. It may be correct not to expect these lookups to be a regular occurrence in a networked environment's daily traffic and, as such, be alerted for such attempts.

4. Detection of potential threats that rely on IP address lookups

Snort is a widely used intrusion detection/prevention system that has enjoyed a universal deployment into networks everywhere. With its ease of use and flexibility, custom Snort rules can be written to detect these suspicious IP address lookups. Let us look at a previous example:

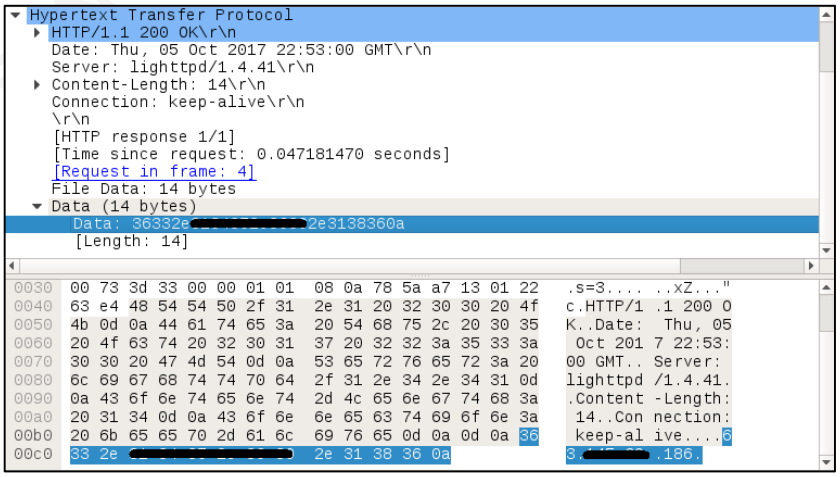


Figure 8: Packet capture of a lookup to checkip.amazonaws.com, using Wireshark

In this example, writing the Snort rule definition can be the IP address in plain text (ASCII) or the hexadecimal representation of the IP address. We can write the Snort rule in this manner, where the external IP address of the networked environment is "11.22.33.44":

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"[TCP] IP Lookup Detected"; \
```

Jay Yaneza, jay_yaneza@trendmicro.com

```
content:"11.22.33.44"; content:"|31 31 2e 32 32 2e 33 33 2e 34 34|"; \
flow:from_server,established; sid:1000984; rev:1;)
```

The definition of this rule is as follows:

- Alert on TCP traffic from the external network going to the home network, on any port, and
- Display the message "`[TCP] IP Lookup Detected`", and
- Match either the content `"11.22.33.44"` or "`31 31 2e 32 32 2e 33 33 2e 34 34`"
- Define that the flow must be "*established*" and that we're interested in the response of the server "*from_server*"

This rule can detect other forms of external IP look-ups as well, such as one the known method of using telnet [myip.gelma.net]:

```
$ telnet myip.gelma.net
Trying 97.107.131.168...
Connected to myip.gelma.net.
Escape character is '^]'.


       Your IPv4: 7x.xxx.xxx.xx8


Connection closed by foreign host.
```

The packet capture shows that the server just responds back with the connecting IP address in plain text as well:

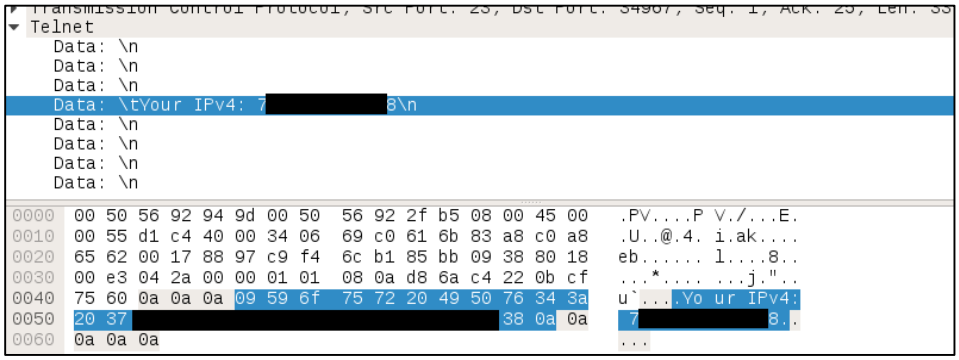


Figure 9: Packet capture of a telnet connection to determine the IP address, using Wireshark

Jay Yaneza, jay_yaneza@trendmicro.com

While not seen in the sandboxed malware sample set, the public IP of a host can be determined with DNS. For this method, we have two examples:

```
Example#1:
$ dig TXT +short o-o.myaddr.l.google.com @ns1.google.com | awk -F'"' '{ print $2}'

Example#2:
$ dig +short myip.opendns.com @resolver1.opendns.com
```

The first example (utilizing ns1.google.com) has a simple DNS TXT record that matches exactly how we can match the TCP Snort rule example:

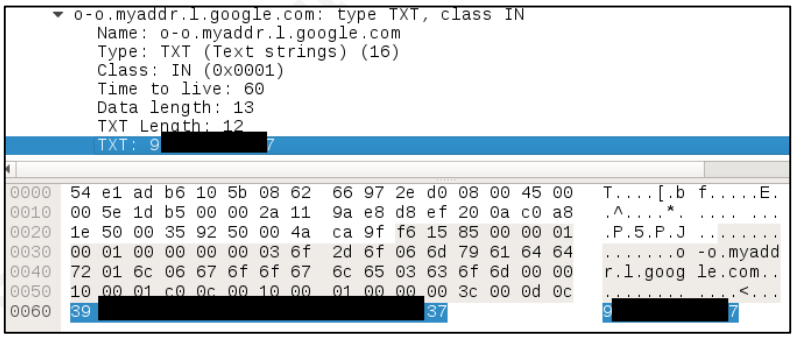


Figure 10: Packet details of external IP lookup, using first DNS method

As the representation is plain text (ASCII), then we can just convert the rule to UDP instead of TCP:

```
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"[UDP] IP Lookup Detected"; \
  content:"11.22.33.44"; content:"|31 31 2e 32 32 2e 33 33 2e 34 34|"; \
  flow:from_server; sid:1000985; rev:1;)
```

However, the second example requires a little bit more understanding of the result itself, and the prior plain text (ASCII) Snort rule does not trigger an alert. Looking at the packet details, we see the following:

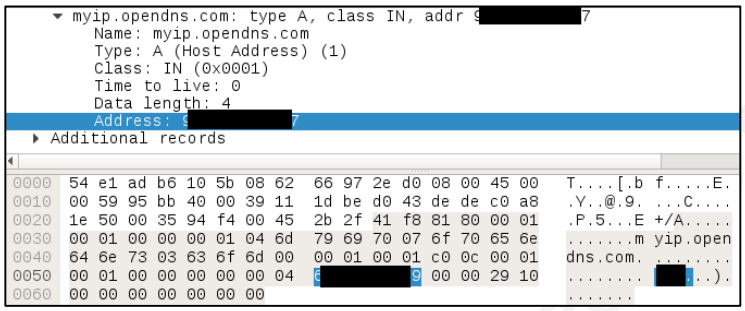


Figure 11: Packet details of external IP lookup, using second DNS method

As seen above, the result is in hexadecimal and, as such, the conversion happens directly from hexadecimal to decimal. In the example where the external IP address is "11.22.33.44", then the rule appears like so:

```
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"[UDP] IP Lookup Detected"; \
  content:"|0B 16 21 2C|"; flow:from_server; sid:1000986; rev:1;)
```

Testing the Snort rule against the external IP lookup methods used by the malware sandbox set, the results indicate that the TCP Snort rule was effective where we had a positive (HTTP 200) response:

IP Lookup Method	Detection count	SNORT Alert Testing Result
icanhazip[.]com	2343	TCP Alert triggered
myexternalip[.]com/raw	1952	TCP Alert triggered
www.myipaddress[.]com	1786	TCP Alert triggered
ipinfo[.]io/ip	995	TCP Alert triggered
ipecho[.]net/plain	459	TCP Alert triggered
wtfismyip[.]com/text	423	TCP Alert triggered
ipgeoapi[.]com/ext	323	TCP Alert triggered
myexternalip[.]com/text	265	Error - received HTTP 404
ipecho[.]net/raw	229	Error - received HTTP 404
ipecho[.]net/text	225	Error - received HTTP 404
api.ipify[.]org	202	TCP Alert triggered
wtfismyip[.]com/raw	65	Error - incorrect request
checkip.amazonaws[.]com	32	TCP Alert triggered
myexternalip[.]com	27	TCP Alert triggered
ipinfo[.]io/json	18	TCP Alert triggered
checkip.dyndns[.]com	16	TCP Alert triggered

Jay Yaneza, jay_yaneza@trendmicro.com

www.showipaddress[.]com	16	TCP Alert triggered
whatismyipaddress[.]com	16	Error - access denied
www.myexternalip[.]com	12	TCP Alert triggered
myexternalip[.]com/plain	8	Error - received HTTP 404
ip.anysrc[.]net/plain/clientip	7	TCP Alert triggered
wtfismyip[.]com	6	TCP Alert triggered
ipecho[.]net	6	TCP Alert triggered
ip.anysrc[.]net	5	TCP Alert triggered
api.ipify[.]org/?format=text	3	TCP Alert triggered
myexteranlip[.]com/raw	3	Error - NXDOMAIN
www.myexternalip[.]com/raw	2	TCP Alert triggered
wtfismyip[.]com/xml	2	TCP Alert triggered
wftismyip[.]com/text	1	Error - NXDOMAIN

Noticeable in the table above, some result contains error results. The result tells us that this may be a wrong implementation, or possibly coding error, of the sandboxed malware and these results did not trigger the TCP Snort rule.

The results also show that there is a high chance of identifying potential threats that had reached the endpoint without relying on any anti-malware vendor. As often experienced, identification of positively malicious artifacts that had reached the endpoint relies on the indicators such as detection of the malicious file itself (or attributes thereof), blocking of a malicious routine of execution, blocking of possibly known command-and-control channel that is associated to the threat.

5. Future considerations and challenges

Depending on the use case of the threat, being able to acquire benign content can be enough to elect the appropriate behavior without the need to communicate back to a command-and-control server:

a) The analyzed examples above were examples of 3rd party services that had utilized HTTP and DNS, but it is still possible to use other protocols to achieve the same effect of being able to determine the public IP address of a host. One such protocol that used in the past was Session Traversal Utilities for NAT (STUN), utilized for real-time voice, video and other messaging

services to discover a host's public IP address. STUN gained increased malicious utilization around 2014 (Palo Alto Networks Blog, 2014), particularly with the DYRE/DYREZA malware family (TrendLabs Security Intelligence Blog, 2014) as it sends this information back to its command-and-control site.

b) Public IP address lookups is just one example of benign content that can be more problematic to detect than known malicious traffic. Another outstanding example of such innocuous downloaded content is http://constitution[.]org/usdeclar.txt. It was first documented to be part of the domain generation algorithm (DGA) of the threat known as ROVNIX (Palo Alto Networks Blog, 2014) and is also a staple DGA template for the e-Banking Trojan GOZI/ISFB as well (Swiss Government Emergency Response Team, 2016).

c) Finally, another observed routine would be to check if it is running in an actual desktop or if it is running in fact a sandboxed environment. This method is practically straightforward by querying online resources as well to determine the provider or organization to which the IP address belongs, and terminating execution if it matches a pre-defined banned or its internal block list within its own code (Zscaler Blog, 2016). It is worth mentioning that a malicious program that is intent in trying to find out a hosts' public IP address may be trying to determine if it can escape the environment.

The segmentation of the protected network should support endpoint IP address identification, and there are certainly challenges in doing so:

a) DHCP or dynamic IP address allocations being for endpoints – it may be possible that the endpoint releases its assigned DHCP IP address (expired lease), before positively identifying the real endpoint. The combination of alerting and DHCP logging may be required to accurately identify the endpoint that had triggered the defined Snort rule.

Jay Yaneza, jay_yaneza@trendmicro.com

b) IPS/IDS implemented after network address translation (NAT) – if a network access device had performed NAT, and the true IP address of the endpoint is hidden behind an NAT-ted network, then the benefits of implementing this approach may be partial. The IPS would only block the identified traffic and the IDS would only notify that this particular network segment is producing such alerts. Since the identified activity is not explicitly malicious, it is quite important to identify the endpoint.

c) Requests going through application proxies – if the HTTP request is going through an HTTP proxy server and the IPS/IDS is implemented after the proxy, then the effect would also be the same as item (b) where positive identification of the real endpoint would face some challenges. Likewise, if the identification is done through DNS and the endpoint has a departmental caching server, then the IPS/IDS would not alert on the real endpoint and would throw an alert on the request of the DNS server to the upstream (company or ISP) DNS server.

While we have discussed the high potential of this approach in identifying potential threats, correct implementation of the IPS/IDS is still necessary. Furthermore, after the identification of endpoint, the correct steps according to the organization's incident response handling procedures should take place – potentially directly into the Containment phase if positive identification of the scope and business impact has already been assessed (The Incident Handlers Handbook, 2011).

6. Conclusion

For sure, blocking the cited benign activities in the examples can be done, but it must be first determined if such lookups are normal for the environment. Random IP address lookups that attempt to determine an environment's external IP address do not directly pose as a threat to the environment, but it may characterize a suspicious event if the request looks out of place and unexpected. If it is expected to see external IP address

lookups within the network, then it may be necessary to determine where these requests may originate, and why do they exist.

To monitor external IP address lookups, the implementation would need to be closest to the workstations. For example, in a product called **pfSense** (https://en.wikipedia.org/wiki/PfSense), a known open source firewall/router that has a Snort package, the interface that first receives the traffic should have the custom Snort rule, as seen in the example below:

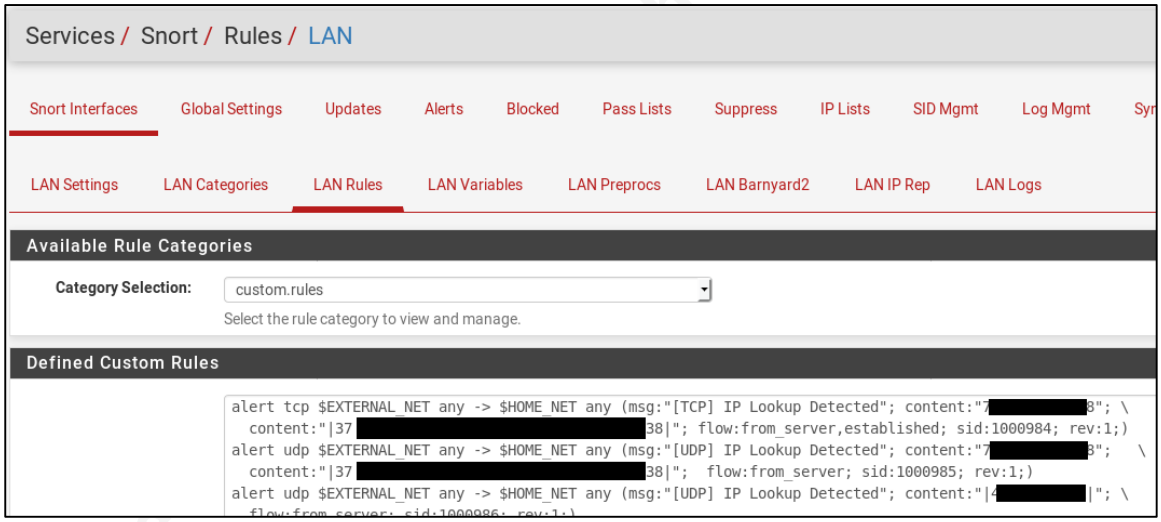


Figure 12: a pfSense firewall that has the suggested Snort rules implemented at the LAN interface

With this in mind, it may also be useful to re-check egress filtering methodologies and placement of the network's IPS/IDS. Stated in prior work, the deployment of an IDS or IPS at each of the department network segment provides a means to monitor and block attempts to violate the policy of the environment (Network IDS & IPS Deployment Strategies, 2008). The methods of IP lookup may be part of an automation routine, but may not be within operating boundaries of standard workstations and normal users. The correct deployment of IPS/IDS that places it nearer to the workstation, or endpoint, would enable successful identification of the host.

After being able to identify that it is unusual to see environment's external IP address in the network traffic (e.g., HTTP or DNS) as responses, then it may be important to turn to the stated procedures of the organization for incident handling.

Jay Yaneza, jay_yaneza@trendmicro.com

References

Geo Redirection | Redirect Website Visitors By Location | Geolify. (n.d.). Retrieved from https://geolify.com/website-geo-location-url-redirection/

GetAdaptersInfo function (Windows). (n.d.). Retrieved from https://msdn.microsoft.com/en-us/library/windows/desktop/aa365917(v=vs.85).aspx

Tutorials/Setting up a server – Official Minecraft Wiki. (n.d.). Retrieved October 5, 2017, from https://minecraft.gamepedia.com/Tutorials/Setting_up_a_server#Firewalling.2C_NATs_and_external_IP_addresses

IP Geolocation and Online Fraud Prevention | MaxMind. (n.d.). Retrieved from https://www.maxmind.com/en/hom

GeoIP2 City Accuracy | MaxMind. (n.d.). Retrieved from https://www.maxmind.com/en/geoip2-city-database-accuracy

TeslaCrypt shuts down and Releases Master Decryption Key. (2016, May 18). Retrieved from https://www.bleepingcomputer.com/news/security/teslacrypt-shuts-down-and-releases-master-decryption-key/

InfoSec Handlers Diary Blog - Upatre/Dyre - the daily grind of botnet-based malspam. (n.d.). Retrieved from https://isc.sans.edu/diary/UpatreDyre+-+the+daily+grind+of+botnet-based+malspam/19657

Comodo publishes strategic analysis of 97M malware incidents in Q2 | Comodo News and Internet Security Information. (2017, July 25). Retrieved from https://blog.comodo.com/malware/comodo-publishes-strategic-analysis-of-97m-malware-incidents-in-q1/

Location-based threats: How cybercriminals target you based on where you live. (2016, May 3). Retrieved from https://news.sophos.com/en-us/2016/05/03/location-based-ransomware-threat-research/

Malware Trending: STUN Awareness - Palo Alto Networks Blog. (2014, September). Retrieved from https://researchcenter.paloaltonetworks.com/2014/09/malware-trending-stun-awareness/

Jay Yaneza, jay_yaneza@trendmicro.com

TrendLabs Security Intelligence Blog A Closer Look At DYRE Malware, Part 1 - TrendLabs Security Intelligence Blog. (2014, October). Retrieved from http://blog.trendmicro.com/trendlabs-security-intelligence/a-closer-look-at-dyre-malware-part-1/

Rovnix and the Declaration Generation Algorithm - Palo Alto Networks Blog. (2014, October). Retrieved from https://researchcenter.paloaltonetworks.com/2014/10/rovnix-declaration-generation-algorithm/

Gozi ISFB - When A Bug Really Is A Feature – Swiss Government Emergency Response Team. (2016, February). Retrieved from https://www.govcert.admin.ch/blog/18/gozi-isfb-when-a-bug-really-is-a-feature

Malicious Documents leveraging new Anti-VM & Anti-Sandbox techniques | Zscaler Blog. (2016, June). Retrieved from https://www.zscaler.com/blogs/research/malicious-documents-leveraging-new-anti-vm-anti-sandbox-techniques

Domain Name System - Wikipedia. (n.d.). Retrieved October 13, 2017, from https://en.wikipedia.org/wiki/Domain_Name_System#Address_resolution_mechanism

The Incident Handlers Handbook - SANS Reading Room (2011, December) Retrieved April 24, 2018 from https://www.sans.org/reading-room/whitepapers/incident/incident-handlers-handbook-33901

Network IDS & IPS Deployment Strategies – SANS Reading Room. (2008, April). Retrieved October 18, 2017, from https://www.sans.org/reading-room/whitepapers/detection/network-ids-ips-deployment-strategies-2143

Jay Yaneza, jay_yaneza@trendmicro.com