Investigating the Threats of IPv6-Based Botnets

Bastien DHIVER

Kent ID: 17903179

bfrd2@kent.ac.uk

School of Computing, University of Kent

Tuesday 11th September, 2018

Abstract

Botnets are part of the Internet security landscape for quite some time and are constantly evolving along with the protocols supporting the worldwide infrastructure. Internet users and organizations are at risk from these malicious networks because their impacts are real and powerful, as demonstrated many times by the discontinuation of online services such as Dyn DNS and Github. The latest version of the Internet Protocol, the version 6, is gaining more ground over the worldwide network and the research community is still assessing the security and privacy matters. This study was conducted in order to avoid any delay in the preparation of the next defensive measures. The observations and documentation provided by this project are key factors that will contribute to this preparation. An extensive reading approach of the current IPv6 security state-of-the-art was performed to enumerate potential future usage of IPv6 features by botnets along with a low cost live experimentation monitoring early signs of IPv6 adoption. Reconnaissance patterns were observed from multiple sources and some IPv6 concepts might be leveraged by botnets to improve their key properties such as attack attribution, target reconnaissance and stealth communications. IPv6 full acceptance is just a matter of time, anticipation is one of the best defence strategy available. Early monitoring must be put in place by Internet organisations in order to take the appropriate measures promptly.

Number of words: ~9750

Acknowledgements

I would like to thank the University of Kent for letting me choose the subject that occupied three months of my life (and allowed us to submit our dissertations online). My advisor, Budi Arief, for his availability, his feedbacks, valuable advice on how research is conducted and structured (and for listening to me when I was hastily talking about all my discoveries). The research community for writing interesting papers (nearly all the time). My last thank goes to my parents for making this all possible (and for helping me carry my heavy suitcases back to France).

Contents

1 Introduction **1**

2 Background and Related work **2**
2.1 Botnets . . . 2
2.2 IPv6 Security and Privacy . . . 3

3 Methodology **5**
3.1 Extensive Literature Review . . . 5
3.2 Live Experiment . . . 5

4 Design and Experiments **6**
4.1 Leveraging IPv6 Features . . . 6
4.1.1 Attribution . . . 6
4.1.2 Stealthiness . . . 6
4.1.3 Reconnaissance . . . 6
4.1.4 Propagation . . . 8
4.1.5 Infiltration . . . 8
4.1.6 Attacks . . . 8
4.1.7 Limitations and Mitigations . . . 8
4.1.8 Moving Target IPv6 Defence . . . 9
4.2 Setting Up the Observation Point . . . 9
4.2.1 Getting IPv6 addresses . . . 9
4.2.2 Advertising addresses and prefixes . . . 10
4.2.3 Infrastructure overview . . . 12
4.2.4 Technical network aspect . . . 13
4.2.5 Network captures . . . 14
4.2.6 Analysis of captures . . . 14
4.2.7 Honeypot services . . . 14
4.2.8 Evaluation . . . 15
4.2.9 Risks . . . 15

5 Results and Analysis **16**
5.1 The Budapest native endpoint . . . 16
5.2 The Los Angeles tunnelled endpoint . . . 16
5.3 The London tunnelled endpoint . . . 17
5.4 The New York tunnelled endpoint . . . 17
5.5 The other tunnelled endpoints . . . 18

6 Discussion and Evaluation **19**

7 Conclusion and Suggestions for Future Research **20**

8 References **21**

9 Appendices **27**
9.1 Simple HTTP server source code . . . 27

1 Introduction

Botnets attacks are often mentioned in the news. One of the most common usages of them is to launch Distributed Denial of Service (DDoS) attacks to take down Internet services. For example, the latest and most powerful attacks recorded were the Mirai botnet in September 2016 targeting some OVH customers [1] and the one that leveraged the Memcached UDP amplification vector which made Github unavailable in February 2018 [2, 3]. With the unprecedented number of devices that are, and will be, connected on the Internet and due to the IPv4 address exhaustion, these devices are assigned IPv6 addresses. As a consequence of this, the attack surface grows a little larger as new protocols and implementations are in use, devices are directly connected to a global network (with or without filtering) and they still have an IPv4 address (dual-stacked). The more exposed devices, the greater the interest for botnet herders, the stronger potential attacks can be if they get compromised. The interest shown for this evolution of the Internet Protocol by attackers is increasingly taken into account by attack toolkits [4], so IPv6 full adoption seems to be the next logical step for bots networks. As the IPv6 standards bring a lot of new features, new security and privacy concerns were highlighted by the research community. Ensuring service availability is a crucial element for all services that are accessible online. Failure to do so directly impacts the profits made by a company and can damage its image and in particular the trust placed in it by its customers. Organizations must therefore protect themselves against these risks by keeping pace with recent developments but also by anticipating future uses that could potentially affect the maintenance of a stable and high-performance service. Unfortunately, the studies and literature currently available on this subject are lacking. This is what this project is trying to tackle.

New architectures and communication schemes will emerge relying on side effects of IPv6 new features and usages. The research questions addressed in this work are the following: To what extent this evolution can affect the Internet security landscape in term of service availability and what potential countermeasures will be of interest? Is it possible to perceive early signs of such appropriation on the worldwide global network as of now?

Detecting such usages and taking botnets down will always require more creativity and is a timely subject. The outcomes of this project would help company security teams to anticipate and detect fraudulent usages of their infrastructures and help to increase the global Internet security. Understanding the magnitude of the potential evolutions is of importance to make the Internet a bit safer.

The aims of this project are:

- to enumerate potential future usage of IPv6 features that can be leveraged by botnets to ensure their attribution, reconnaissance and stealthiness
- to describe potential countermeasures and to document some visible evidence of their presence on the IPv6 side of the Internet

Most of the work was devoted to document the impacts of the IPv6 adoptions by botnets using public knowledge mainly from research sources. A low budget observation infrastructure was set up with publicly available services to monitor the first indications of activity from those networks. The hardest part was to be able to anticipate future trends in the field and for that monitoring is a precious ally.

Section 2 below, provides some background to the reader about botnets and the current state of research on IPv6. The methodology used for this work is presented in section 3. In section 4, details are given about a design and the conducted experiment. Results and Analysis are provided in section 5, followed in section 6 by an evaluation of this work. Finally, suggestions for future research are presented in section 7.

2 Background and Related work

The Internet Protocol has evolved over the years since the original specification of IP version 4 (RFC 791) [5] in 1981 followed, 17 years after, in 1998 by the version 6 (RFC 2460) [6]. IPv4 is well understood and is today the most widely adopted protocol version on the Internet. Mainly due to IPv4 addresses exhaustion, an IPv6 transition took place bringing a new capacity of 2^{128} addresses. Not as quick as some would have expected, the transition is still an ongoing work. Similar to IPv4, IPv6 suffers from design drawbacks which lead to security and privacy issues. A lot of research has been done on botnets over the years almost exclusively on those who use IPv4. More and more services and devices are IPv6 ready and connected every day. IPv6 support by botnets is inevitable to follow the evolution of the Internet.
Only a few occurrences can be found in the literature about the two main keywords that are IPv6 and Botnets. In 2012, Li *et al.* [7] described for the first time the potential impacts of IPv6 as an evasion tool for botnets and its usage in malware distribution networks. Previous techniques used in IPv4 to defend against botnets seem no longer appropriate while new techniques and algorithms are yet to be invented to defend networks. Network policies are to be updated accordingly as well as blacklisting techniques. A moving target IPv6 defence concept using address hopping, MT6D 4.1.8, is mentioned. In 2014, Code24 [8] gave a conference at Derbycon 4 and enumerated the main drawbacks of IPv4 botnets. Their reliance on DNS for obfuscation makes them easy to sink-hole and attribution is easier due to the limited address space offered by IPv4. An IPv6 botnet design blueprint that leverages an address hopping technique is described and enables a strike and move attack pattern.

2.1 Botnets

IPv4 botnets are quite heterogeneous in their architectures, aims, behaviours, and evolution. First botnets had a simple centralised client-server architecture, the botmasters were mostly operated through the IRC protocol. The vast range of their usage varies from performing DDoS attacks, mining cryptocurrency, phishing, gathering personal data, or even launching spam campaigns. One can rent such services online. [9, 10, 11]. In order for a botnet to stay up and running, key properties must be satisfied. Robustness against being taken down, resilience in the communication channels, stealthiness to avoid detection and observation, propagation to gain more and more resources (computational power and network bandwidth), infiltration by third-parties must be prevented and any single point of failure avoided. A wide range of architectures exists such as the centralised one, the P2P or the Hybrid [12] and novel approaches [13] are all aiming to increase those malicious network resilience. Creative ways are even relying on computational trust mechanism to avoid botnet enumeration and tracking. One property is to consider the sharing of malicious commands as illegal so researchers and agencies are not inclined to follow this path [14]. Recent infection mechanisms are often targeting the Internet of Things (IoT) as they suffer from a lack of security by design. Such mechanisms are simple but quite efficient. The Mirai and Hajime botnets are good examples [15, 16, 10]. The typical infection process is via "pseudo-random" scanning of the IPv4 address space looking for open ports such as the Telnet service (tcp/23) to attempt multiple credential guesses. Detection and monitoring techniques are mainly relying on network inspection. Either via payload-based or flow-based analysis approaches, even if the flow-based approach can be applied to encrypted traffic [17]. Large-scale monitoring with a high level of interaction can be deployed as it was done by Antonakakis *et al.* study of the Mirai botnet [15]. Selected sources were network telescopes, active scanning, honeypots, malware binary analysis, DNS data, C2 milkers, and target logs analysis. Reconnaissance on botnets is as well often conducted by crawling and sensor injection techniques [18]. Attacks on botnets exploit sinkholing and partitioning techniques. The aim is to disrupt the internal communications in order to isolate bots or groups of bots from the malicious network. Lastly, in 2016, a malware supporting IPv6 has been analysed and is able to send IPv6 packets and spoof IPv6 addresses [4].

2.2 IPv6 Security and Privacy

The majority of hosts are IPv6 capable and IPv6 is on by default. Devices are dual-stack (IPv4 and IPv6) so the vulnerability surface is combined increasing the attack surface and the weakest link will be chosen. The security of the protocol is a timely subject where training resources are easy to find online [19, 20, 21]. A new addressing architecture is in place and the evolution of side protocols such as ICMPv6 and DHCPv6 and new protocols like NDP requires to maintain staff competency up to date. The overall adoption is progressing gradually and the adaptation of network security policies should follow [22]. Implementations of the protocol are not complete yet and some interoperability issues persist. One of the major security obligation with IPsec support is no longer imposed but remain recommended. A large number of transition mechanisms were used during early adoption leaving high complexity and doors opened. Security implementation on local-link is mostly deployed on switches (First Hop Security).
Most of the IPv6 security and privacy vulnerabilities associated with countermeasures are systematized in a 2014 survey by Ullrich *et al.* in [23]. Partly due to the original design that put trust at the core of the protocol, the Internet Protocol version 6 suffers from some common well-known vulnerabilities present in the previous version, version 4. As an example, IP spoofing is still possible. The study is completed with three challenges in the area that will be interesting to investigate in a near future by the research community namely:

- the address assignment and structure
- the security of the local network discovery
- the address selection for reconnaissance.

Two particular IPv6 well studied subjects of interest for botnets are reconnaissance and covert channels.

Reconnaissance Scanning methods are changing as the address space of IPv6 is much larger. Default subnets in IPv6 have 18,446,744,073,709,551,616 addresses (2^{64}). It is common for IPv6 hosts to have multiple addresses and clients often use privacy addresses [24, 25] to prevent device tracking by using a randomly generated Interface Identifier (IID). The IID is generated by the node in most cases if stateful DHCPv6 if not in use. Multiple techniques exist like manual configuration, semantically opaque IID which are stable within each subnet [26] or Cryptographically Generated Addresses (CGA) [27]. RFC 8064 [28] gives the latest recommendation on this topic.
A lot of effort is put into research. In 2013, network telescopes were deployed to measure background radiation in the IPv6 space but no evidence of large-scale scanning was found. Target Generation Algorithms are developed and aim to generate hitlists from IPv6 seed addresses. In 2015, Ullrich *et al.* present a pattern-based algorithm [29] that require user-specified address range and focuses on IID part of the address. In 2016, Foremski *et al.* present the Entropy/IP algorithm [30] that uses machine learning techniques and is able to discover IIDs and network identifiers. Gasser *et al.* generate a 150M IPv6 addresses hitlist from passive flow data, rDNS and tracerouting in [31]. RFC 7707 [25] by Gont *et al.* describe address and assignment patterns and list reconnaissance techniques before giving advice to limit them. In 2017, Murdock *et al.* present another TGA algorithm 6Gen [32] and compare it with Entropy/IP. Widespread IP aliasing in IPv6 is uncovered and a detection technique is described. In [33], Fiebig *et al.* collect IPv6 addresses by exploiting DNS NXDOMAIN answers. In 2018, Gasser *et al.* inspect Certificate Transparency logs [34] to gather domain names and thus addresses. In another study [35], Gasser *et al.* describe a technique to improve hitlist quality and a method to detect aliased prefixes. The same year, Beverly *et al.* [36] use the Yarrp6 tool for Internet-wide IPv6 topology mapping and current IPv6 hitlists are analysed for their efficiency. DNSSEC-signed Reverse Zones are leveraged by Borgolte *et al.* in [37] and uncover security and privacy issues due to misconfiguration of IPv6 host filtering. Lastly, Scheitle *et al.* perform the first comprehensive study on the quality of hitlists in [38].

Covert channels One of the first reviews of IPv6 covert channels was done in 2006 by Lucena *et al.* in [39]. 22 covert channels were found in IPv6 and it's extensions headers. Parsing IPv6 extension headers is difficult since their usage is flexible which end up being difficult to process in depth for filtering. Instant messaging over IPv6 Destination Options proof-of-concept was developed in [40]. A covert channel was also implemented in the side protocol ICMPv6 by Murphy in [41]. Blumbergs *et al.* in [42] leverage transition mechanisms to implement two novel covert channels. The first one uses dual-stack networks with sequential IPv4 and IPv6 sessions and, the other IPv6 encapsulation in IPv4 via tunnelling mechanism. Protocol scrubbers, traffic normalizers and active wardens can be used as defensive mechanisms to defeat those channels.

Proofs-of-Concept and Tools A list of IPv6 Proofs-of-Concept and security tools with a description is provided to the reader.

- THC IPv6 Attack Toolkit [43] - Security assessment and troubleshooting tools for IPv6
- SI6 Networks' IPv6 Toolkit [44] - Security assessment and troubleshooting tool for IPv6 similar to the THC suite
- Messaging over IPv6 Destination Options Proof-of-Concept [40] - Covert channel PoC using the Destination Options header
- Chiron [45] - IPv6 Security Assessment framework
- Nmap [46] - Network discovery and security auditing tool that comes with scripts for IPv6
- MT6D [47] - C implementation of the MT6D network layer defence
- Zmap [48] - Fast single packet network scanner supporting IPv6
- mdns-scan [49] - Tool for scanning for mDNS/DNS-SD published services on the local network (host discovery)
- Pholus [50] - An mDNS and DNS-SD security assessment tool
- mzclient [51] - CLI library for multicast DNS service discovery (host discovery)
- Yarrp6 [36] - Large-scale topology measurement tool
- scamper [52] - Large-scale topology measurement tool
- Scapy [53] - Packet manipulation program
- Ettercap [54] - Man-in-the-middle attack tool

3 Methodology

In this section is described the approach taken to get relevant information about IPv6 as well as the design of the experiment.

3.1 Extensive Literature Review

Very little literature is available on IPv6 botnets. They are quite vague, not very technical and details are missing. In contrast, many references can be found on IPv6 security and IPv4 botnets. IPv6 security and privacy key challenges should be carefully considered as future applications and usages can emerge from there. Extensive reading of slides, RFCs, papers, blogs that were found during the literature review was performed in order to get the overall picture of the public knowledge in the field. Third-party sources of information such as hacking forums will have to be found to get as many data as possible. The study of IPv6 specifications and related protocols such as ICMPv6 should be undertaken, as well as the state-of-the-art security and privacy matters related to those protocols and their usages. Potential recommendations from network equipment vendors and institutions should be read thoroughly as potential countermeasures can arise from them. The constant adaptation of search keywords is very important because interesting areas and applications useful to address parts of the research questions can be uncovered. This extensive literature review is somewhat similar to the one made by Ullrich *et al.* in [23]. The approach chosen for this dissertation is nevertheless less focussed on IPv6 technical specifications (RFC) but more on already documented cases and new developments. Usage of IPv6 has been sought on popular hacking forums at the beginning of the project without tangible results.

3.2 Live Experiment

A live experiment is conducted as part of the project so one may observe the current state-of-the-art in real conditions. The experimentation is conducted in order to observe IPv6 reconnaissance and scanning patterns in the wild (on the Internet). The aim is to deploy probes in several places and advertise some IPv6 addresses/prefixes linked to them in an effort to draw attention. Doing so will probably reveal any attempt made by third parties exploring the IPv6 space to find vulnerable machines. This approach is comparable to some extent to the experimentation conducted by Czyz *et al.* in [55]. However, the approach presented in this dissertation uses smaller IPv6 prefixes available for free and commercial services, active address advertising is performed and honeypot services are deployed. Choosing free and commercial solutions increases our chances to observe interesting traffic because prefixes have already been announced for some time and some addresses have potentially been used by previous owners. Address advertisement is done by leveraging current IPv6 reconnaissance techniques available in the literature and implemented in tools. Honeypot services could help to have more insights about the TCP incoming traffic since interaction is mandatory during the handshake. The choices of the protocols and ports must reflect the current scanning practices related to IPv4 networks. Multiple geographic zones should be selected on different continents to get an idea of the extent of the scans. The focus should be put on the usual botnets targets as those shown by [56]. Also, close IPv6 subnets should be chosen, so we can potentially see the spread of scanning in larger subnets.

4 Design and Experiments

4.1 Leveraging IPv6 Features

IPv6 features and concepts that can be exploited by botnets are given in this section.

4.1.1 Attribution

Attack attribution is to be avoided as much as possible as drawing attention would reveal botnets participants. Analysis and monitoring by researchers and authorities should be as long and difficult as possible. Exploiting the dual-stack connectivity to perform malicious actions would require correlation capacities by investigating tools. Client address hopping is allowed and expected. RFC 4941 [24] states that a privacy address can change little as every 10 minutes. Other address hopping techniques should be as close as Privacy Extensions to keep a low profile. Hopping too fast could be suspicious. Static filtering of known malicious URLs is insufficient since more hosts can acquire a vast number of global IP addresses. The number of IPv6 addresses makes blacklisting and reputation of addresses impractical. Prefixes are shared and even if some would be flag some, the number of prefixes is still high. The MT6D technique, presented in section 4.1.8, can really make debugging and attribution much more difficult.

4.1.2 Stealthiness

Exfiltration of collected data and stealth communications can benefit from some IPv6 covert channels. Those channels should mimic real traffic as much as possible. As mentioned in the literature review section 2, the IPv6 and extension headers can be used to secretly transmit information. The unlimited size of the IPv6 header chain can make filtering difficult (there is an order though). Yet, about 20-40% of packets with Extension Headers are dropped over the Internet according to RFC 7872 [57] which was published in 2016. But as more and more network equipment takes into account recent IPv6 RFCs, this number is set to decrease. Transition technologies such as tunnelling, translation (NAT64, DNS64) and dual-stack are a great way to covertly exchange data in light of their large number. Also, among other things, hosts must be able to establish the path MTU which implies the ICMPv6 messages used for PMTU discovery must not be filtered blindly (RFC 4890 [58]).

4.1.3 Reconnaissance

IPv6 comes with an enormous address space, Network Address Translation (NAT) is not needed any more. Hosts are globally reachable if no filtering policies are put in place. Clients are autoconfiguring themselves with Stateless Address Autoconfiguration (SLAAC), Duplicate Address Detection (DaD) and Neighbour Discovery Protocol (NDP). In IPv6, hosts are usually multi-addressed with at least a link-local and a global IPv6 address. Hosts addresses are actively sought to measure the deployment of the "new" protocol as this is done by SBA Research [59]. Network prefix can be found via addressing patterns, traceroute and DNS direct and reverse resolution. IID patterns can be of little help to determine the type of the machine (router/server/client). Algorithms and techniques are used to determine additional addresses from IPv6 seeds and can indicate how relevant each portion of an address is. Techniques to detect large aliased regions to check for network telescopes/darknets are in constant evolution. This means that detecting such regions is possible by bots during their initial scanning phases.
A compilation of the different methods cited in the literature for gathering addresses is presented to the reader:

- Be part of an NTP pool (Shodan demonstrated the feasibility [60])
- Scan for patterns in host addresses (well described in RFC 7707 [25])
- IPv6 block allocation patterns by IPv6 address providers

- DNS reverse enumeration [33]
- DNS datasets (e.g. Alexa, Umbrella, Majestic, Rapid7, CAIDA, Farsight Security Passive DNS project) including NS and MX DNS entries
- DNS zone transfers (TLDR [61])
- Leverage search engines to enumerate subdomains
- Leverage certificate transparency
- Look for lists of public/leaked registered domain names [62]
- Address collection from network taps at exchange points
- Tracerouting known addresses
- Monitoring peer-to-peer networks such as BitTorrent (nodes references by trackers) and the Bitcoin network (via the Bitnodes API) [35]
- Local-link multicast discovery (MLD - Multicast Listener Discovery)
- Local name resolution and service discovery services like mDNS, DNS-SD and LLMNR
- Network snooping (NDP, routing traffic, …)
- Gather domains from public mailing-list or Usenet news messages archives
- Inspect IPv6 neighbour cache and routing table (locally or via SNMP)
- Inspect hosts logs of servers with a high number of client connections such as CDNs
- Passive listener of the routing protocol [63], BGP advertised IPv6 prefixes (CAIDA [64])
- Gleaning information from IP Flow Information Export (IPFIX)
- IPv6 compliant Internet search engines like Shodan.io [65]
- Spam/phishing blacklists from Spamhaus, APWG and PhishTank.
- Host IPv6 neighbour cache, routing table and log files

Host-probing consideration ICMPv6 echo request packets can be filtered on the targeted host, different approaches are to be considered such as using a transport layer protocol techniques as nmap does [1] or use an unrecognised extension header or an unrecognised option of type 10xxxxxx in an IPv6 packet to solicit answers.

[1] `https://nmap.org/book/man-port-scanning-techniques.html`

4.1.4 Propagation

Infection like the Mirai botnet is not feasible because the IPv6 address space is too wide to be randomly scanned with good results. In 2006, some worm propagation strategies in the IPv6 world are listed by Bellovin *et al.* in [66]. The mentioned local information sources of addresses are neighbour discovery and routing tables, multicast ping, host configuration and log files, DNS zone transfers and passive eavesdropping. SNMP read-only information of a router could be used to read IPv4/IPv6 neighbour caches. The vast majority of IPv6 hosts is dual-stack by default. Even if IPv6 is not configured on a host, a local attacker can silently ask for IPv6 to be configured by emitting a Router Advertisement (RA) message. Target Generation Algorithms are not feasible by a resource-limited host such as a single bot in a scan-to-infect scenario. Lot of results can be generated and good IPv6 seed addresses must be provided. However, a lightweight implementation of a TGA with fed with local-link gathered addresses could help to find active hosts. A target generation host that is fed by the botmaster or external sources mentioned in section 4.1.3 can be a complementary solution. The targets/ranges are then dispatched (pull or push) among the bots by this target generation host. The infection process must be relatively quick regarding clients since their addresses lifetime is in used for a limited time. An assigned address will remain constant, minimally, until the network session is terminated.

4.1.5 Infiltration

Secure communications from trusted sources is a must to avoid crawlers and sensors. IPv6 security mechanisms such as IPsec, CGA and SeND [67] can be used for bots communication. Some information security properties such as integrity, confidentiality and authentication would be ensured. Network appliances should support them. Unfortunately, the mentioned security mechanisms are not widely adopted. It could attract attention in this case. A bot could check if the address of an incoming packet possess certain properties (similarly to CGA) before accepting a connection.

4.1.6 Attacks

Without local-link security mechanisms enabled, the attacks that can be carried out in IPv6 are very similar to those already present in IPv4. The implementation and uses of the recent versions of the IPv6 protocol are still an ongoing work and updates are not deployed on all equipment. A set of vulnerabilities is most likely hidden within there. Aggressive scanning such as scanning an entire default subnet (/64) could create more entries in the neighbour cache of the last-hop router that it can handle resulting in a Denial-of-Service. It is still worth noting that the IPv6 load impact is smaller than in IPv4 because IPv6 header is smaller, fixed in size and thus easier to process if no extension header is used [68].

4.1.7 Limitations and Mitigations

Organisations and companies are regularly issuing recommendations and best practices to follow [19, 20, 21]. For instance, CISCO is issuing multiple advice on filtering and global usage. Privacy Extensions are to be used for external communication but not for internal networks. Doing so enables troubleshooting and attack trace back. Also, the required extension headers must be allowed and packets without layer-4 header are to be dropped. A warning is issued about most IPv4/IPv6 transition mechanisms because they don't have authentication mechanisms built in which leads to potential spoofing and firewalls bypassing. As for the use of IPsec, it must be used in the same circumstances as IPv4. Multiple security implemented on switches can be set up such as RA-GUARD [69], DHCPv6-Shield [70], Neighbour Discovery and source/prefix inspection. Filtering policies have to be chosen carefully regarding ICMPv6, Extension headers, Fragments and transition mechanisms. The host is responsible for his reachability. Every host can have multiple IPv6 addresses simultaneously and switch between IPv6 and IPv4 instantly, a correlation need. Tools like Network intrusion detection systems (NIDS) must be updated

and take into consideration that IPv6 addresses can be written in multiple ways [71] (e.g. the address `2001:db8:cafe::a40:4242` is the canonic form of `2001:0db8:cafe:0000:0000:0000:0A40:4242`). Standard log files parsing need to evolve accordingly. Network administrators can disable SLAAC enforce stateful DHCPv6 with section 4.1.2 of RFC 7707 [25] taken into account to avoid sequential address assigning. Employing RFC 7217 [72] to eliminate address patterns is a good practice, in addition, to the advice given in [25]. Only filter some ICMPv6 messages (RFC 4890 [58]). Network flow analysis will be more and more solicited on IPv6 networks.

4.1.8 Moving Target IPv6 Defence

Moving Target IPv6 Defence (MT6D) is one of the moving target defence techniques and provides interesting security and privacy features. The subject gained in importance from 2010 and a lot of effort is invested in the MT6D technique [73, 74, 75, 76, 47, 77]. IPv6 vast address space allows it to work by contrast to IPv4 which is small and densely populated. Address hopping technique with dynamic address generation algorithm makes devices hard to track as the host address is no longer an attack vector. Similar to Privacy Extensions and Cryptographically Generated Addresses (CGAs) but uses a very short change interval. This technique is implemented at the network layer, so it is application independent. Three parameters are used to compute the host IID: a host identifier, a secret and a changing value known by both parties. The changing value is used as a rotation interval and is often set to 3 seconds. Packets are encapsulated in UDP to keep TCP connections open and full encryption is available is an option. Proofs-of-concept are developed in C and Python and implementation details closer to the hardware level exists [77]. Two different deployments are possible, either embedded in the host or used through a gateway. Small overheads were measured on packet loss, latency and packet size but optimizations at hardware and kernel level are put in place. A table with profiles of other trusted hosts is maintained by the system. Communicating parties have to share a common secret for the system to work. A direct public key exchange can be used to do so but can reveal the two communicating parties. Alternatives exist as the one mentioned by Morrell *et al.* in [76] which allow to securely exchange information through BitTorrent Distributed Hash Table (DHT) among other advantages such as enabling mobility for clients and providing large key exchange infrastructure. Monitoring technique to identify addresses that were discarded by an MT6D host and later acquired by third-parties [75] exists. Security advantages are numerous. It makes Man-in-the-Middle attacks impossible and drastically reduces the effects of a DDoS attack. The potential damages that can be inflicted are limited by the interval of address-hopping. Reacquiring the target is therefore infeasible without the secret parameter. It also prevents an attacker from knowing if the same two hosts are communicating. And replay attacks are not possible if past IIDs are correctly purged from the hosts. Privacy is addressed in details. Ports are pseudo-randomly obscured or mimic normal network traffic. MACs in the Ethernet frames are also overwritten to improve the privacy. Anonymity is preserved as it does not rely on the use of DNS for instance. This technique is well fitted for embedded systems with limited resources such as the IoT if some parameters such as hash algorithms are chosen carefully [78]. Time-based and energy-based attacks which are aiming to make the device run out of sync and run out of battery power.

4.2 Setting Up the Observation Point

The experimentation carried out in this project is described in this section. The aim was to observe large-scale IPv6 recognition signs.

4.2.1 Getting IPv6 addresses

Since a single machine with a single IPv6 address and emitting no traffic will unlikely be scanned, multiple machines with large IPv6 subnets have to be set up and advertised in such a way to improve the chances of observing incoming traffic of interest.

The bigger the IPv6 block can be, the better, but getting multiple large IPv6 blocks is not trivial. An ISP (Internet Service Provider) will get a /32 up to /29 (and can request for more but need to supply additional information) according to the RIPE NCC [79]. The Classless Inter-Domain Routing (CIDR) notation (e.g. */32*) is used to indicate the number of bits that are defined by the network. In IPv6, a /32 subnet have $2^{128-32} - 1$ addressable hosts, that is 79,228,162,514,264,337,593,543,950,335 hosts. A /128 can only address one host. The minimum recommended IPv6 block assigned to an end user is a /64 up to a /48 depending on the provider [80]. Organisations are typically assigned a /48 and must provide documentation justifying the need for additional subnets if necessary. Luckily, few IPv6 tunnel brokers still exist and can offer up to a /48 for free. IPv6 tunnel brokers were very common during the initial IPv6 transition and one can use them to encapsulate IPv6 packets in IPv4 ones (6to4). To name some of them:

- Hurricane Electric IPv6 tunnel broker [81] is a Californian service providing /64 and /48 IPv6 tunnels for free. Up to five tunnels can be requested per account. Multiple endpoints are available around the globe.
- NetAssist IPv6 Tunnel Broker [82] is a Ukrainian service providing /64 and /48 IPv6 tunnels for free. Up to a single tunnel per account. Only one endpoint is available in Ukraine.
- Pemsy [83] is a Poland company providing /56 IPv6 tunnels starting at $9 per year.

IPv6 transition technologies are less and less in use in favour of the dual-stack approach. Tunnels brokers are closing. In fact, according to the research conducted, only those three IPv6 tunnel brokers are still offering their services. One can rent a Virtual Private Server (VPS) that posses an assigned IPv6 subnet for a decent price. This experimentation is hosted on rented commercial services which are providing IPv6 native support. The maximum subnet size offered by providers is a /48 (2^{80} addresses) and the minimum is a /128 (1 address). More than twelve VPS offers including an IPv6 subnet greater than or equal to a /64 were found during this experiment. The monthly price range varies from $8.80 for a native /48 to $2 for a /64 (with VAT). The VPS locations are mostly located in Europe and in the USA. Renting /48 is also possible in Russia and Hong-Kong for instance. A non-tunnelled /43 IPv6 block (which is the maximum via this service) has been requested at the beginning of August through ip6.im [84] but no answer has been given at the time of this writing. Entries in the RIPE database were inserted in order to complete the request form.

4.2.2 Advertising addresses and prefixes

Reference some addresses at specific places and generate specific network traffic from others seems a good way to advertise and solicit incoming traffic to our hosts. Multiple sources of IPv6 addresses are harvested by researchers in order to get an overview of the IPv6 deployment.
One must consider the usage of short-lived addresses for outbound connections. After a certain delay (typically 24 hours [24]), the short-lived address will not be used any more for new connections. Using such addresses is not impacting the experiment since the whole allocated IPv6 subnets are monitored and as long as server services are bound to a fixed IP address.

P2P Networks P2P networks are crawled to gather IPv6 addresses. The most recent example is given by Gasser *et al.* in [35] were IPv6 client addresses were gathered via the Bitnodes API [85]. Such P2P networks can be the BitTorrent network and some cryptocurrencies networks such as Bitcoin. The Bitcoin network was chosen for this experiment, more and more nodes are joining the network and nodes can be of interest since currency is probably stored in some hosts. The Ethereum network was considered but not enough IPv6 nodes are available as underlined by Gencer *et al.* in [86].

A list of Bitcoin nodes was downloaded from the Bitnodes API on the 26th of August 2018. A preview of the snapshot is available on the website [2]. One IPv6 only node was deployed on the Bitcoin P2P network by the end of August. One address from a London location is in use. External access was tested thought the Bitnodes website form.
The bitcoin core software was downloaded from the official website [87]. Due to the storage requirements of running a full bitcoin node and the reduced storage space of the VPS, the pruning mode was enabled. This mode can automatically delete old blocks to stay under a specific target size.
The corresponding command line is as followed:

```
./bitcoind -datadir=./btcdata -prune=550 -bind=[<IPv6 addr>]:8333 -externalip=<IPv6 addr>
-maxconnections=10000 -onlynet=ipv6 -disablewallet -logips -listenonion=0
```

The visibility of the node through the P2P network can be improved by directly contacting many other nodes. The `-addnode` parameter can be used to specify a node address to contact. Doing so spreads the IP of our node in the neighbour list of other nodes. Nodes addresses from the Bitcoin network were gathered via multiple sources by the end of August:

- the bitnodes API [3] (1272 nodes)
- AAAA DNS records from domain seeds [4] (90 more nodes)
- Hardcoded IPs [5] (345 more nodes)

Search engines for Internet-connected devices Multiple search engines reference Internet connected devices such as Shodan.io [65], Censys [88] and Zoomeye [89]. Search an IPv6 address in those search engines will probably reference them and lead to be scanned by the platforms. Only the Shodan.io service supports IPv6 addresses with the `has_ipv6:true` filter. Four different IPv6 addresses were queried on the search engine at different times. And one was submitted for an on-demand scanning via the API with success but was not referenced afterwards.

Network Time Protocol Contact multiple NTP servers through NTP pools. This technique has been used by Shodan.io to discover active IPv6 hosts. 171 IPv6 members from the NTP pool project were gathered via DNS queries of AAAA records. Queries were made to all of them on Thursday 30 of August. The tool ntpdate was used with the Query Only argument `-q`. Outgoing requests were originating from a single IPv6 in the Sydney subnet.

DNS The majority of IPv6 address harvesting techniques leverage DNS features and search. Register domain names and link IPv6 addresses to them is a good start as it enables DNS reverse enumeration technique to discover our domain name at the same time.

Public TLD zonefiles Having a Top-Level Domain (TLD) that publishes the registered domains is also a good idea as it simplifies the enumeration of a tld domain by third-parties. A TLD is one of the domains at the highest level in the hierarchical DNS such as .com, .org or .net. Multiple TLD zone files are available publicly with or without form request. For instance, the .com and .name can be accessed

[2] `https://bitnodes.earn.com/nodes/1535319017/`

[3] `https://bitnodes.earn.com/api/#list-nodes`

[4] `https://github.com/bitcoin/bitcoin/blob/427253cf7e19ed9ef86b45457de41e345676c88e/src/chainparams.cpp#L132`

[5] `https://github.com/bitcoin/bitcoin/blob/427253cf7e19ed9ef86b45457de41e345676c88e/src/chainparamsseeds.h#L10`

after submitting a form to Verisign [90]. The .se and .nu are freely and publicly available via AXFR zone transfer [91]. gTLD zone files can be requested via the Centralized Zone Data Service [62] provided by the ICANN. At the beginning of August 2018, a .science domain was rented and let non advertised. By the end of the month, a random IPv6 address from the New York subnet was added. According to Verisign Domain name report Q2 2018 [92], the .net TLD is the largest gTLD by the number of reported domain names. A .net domain was rented and a random IPv6 address was added as a AAAA DNS record. The gTLD with the most access granted from November 2014 to July 2018 on the CZDS platform is the .technology. Such a domain was ordered by the end of August 2018 and a random IPv6 address was added as well. Finally, another random IPv6 address was added at the beginning of September 2018 to a .fr domain owned for the last three years.

Certificate Transparency The obtention of an SSL certificate might also help to advertise a domain thanks to the Certificate Transparency service. The caddy [93] web server was used to request a TLS certificate from Let's encrypt [94] for the .science domain by the end of August. The certificate is referenced by the crt.sh web service [95] and by Google Transparency Report [96]. NAT prerouting was configured with ip6tables so that only one IPv6 address from the whole /48 New York subnet can reach the caddy web server.

4.2.3 Infrastructure overview

The setup infrastructure consists of multiple hosts: (IP addresses have been redacted)

- VPS from a French cloud company (5 IPv4 addresses)
  - 5 /48 HE tunnels (London, Los Angeles, Sydney, New York, Toronto)
  - 1 /48 NA tunnel (Ukraine)
- VPS from a London hosting company (1 IPv4 address)
  - 1 /48 native (London)
  - 1 /48 HE tunnel (Tokyo)
  - 1 /48 NA tunnel (Ukraine)
- VPS from a Russian hosting company (1 IPv4 address)
  - 1 /48 native (Russia)
  - 1 /48 HE tunnel (Paris)
  - 1 /48 NA tunnel (Ukraine)
- VPS from a Hungarian hosting company (1 IPv4 address)
  - 1 /56 native (Hungary)

Rent a VPS with a /48 block in China and in Brazil was not possible. IPv6 was not supported by one of the major Chinese cloud provider at the time of this experiment after asking the support service. The location of the IPv6 tunnels was chosen depending on the most targeted countries as shown by DDoSmon web service[56]. Native /48 ordered to check if tunnels traffic is representative. The first VPS was rented on the 2nd of August. The whole infrastructure was up ten days later.

4.2.4 Technical network aspect

Addresses and prefixes have been deliberately obscured as the experiment is an ongoing personal project.

The infrastructure put in place has evolved several times iteratively during the project phase. Only the current iteration is presented in this dissertation. The VPS were running the Debian operating system. First, network interfaces are set up in the network configuration file `/etc/network/interfaces`. IP failover had to be configured on one host in order to open multiple IPv6 tunnels. Couple IPv4 addresses were ordered and configured on the VPS so that multiple IPv4 addresses could be assigned to a single network interface. Here is a snippet of the configuration: (`ens3` is the name of the primary network interface)

```
auto ens3:0
iface ens3:0 inet static
        address 198.51.100.42
        netmask 255.255.255.255
```

Once an IPv6 tunnel is created via the web interfaces of tunnel brokers, tunnels configurations are added.

```
auto he-ipv6
iface he-ipv6 inet6 v4tunnel
address 2001:db8:1::2
        netmask 48
        endpoint 192.0.2.1
        local 198.51.100.1
        ttl 255
        gateway 2001:db8:1::1
```

Note that configuring an IPv6 tunnel on a virtualised host using the OpenVZ technology is not always possible due to missing kernel modules and configuration of the virtualisation technology. Network interfaces are isolated from one another thanks to the Linux network namespaces. The process goes as followed:

1. the network namespace is first created
2. the tunnel interface is moved into the namespace
3. the primary IP address of the tunnel is configured
4. the tunnel and the loopback interfaces are turned on
5. routing instructions are added to route all IPv6 traffic through the tunnel interface

Here are the corresponding instructions:

```
ip netns add ns-london
ip link set he-ipv6 netns ns-london
ip -n ns-london addr add 2001:db8:1::2 dev he-ipv6
ip -n ns-london link set he-ipv6 up
ip -n ns-london link set lo up
ip -n ns-london route add ::/0 dev he-ipv6
```

One more command is executed during the network configuration which is `ip -n ns-london route add local 2001:db8:1::/48 dev he-ipv6`. The AnyIP [97] feature of the Linux kernel is used to assign the whole IPv6 subnet to the specified interface so that any packets received on any address in the subnet

will be received and treated as if it has been received on the primary IP address. This makes possible to record incoming traffic on any addresses from a local interface. The modification can be seen in the routing table via the command: `ip -6 route show`.
An IPv6 DNS resolver address was added in the `/etc/resolv.conf` file so it is possible to resolve IPv6 domains from our IPv6 only interface.
The sysctl parameters `net.ipv6.conf.<interface name>.use_tempaddr` were set to 0 by default on all the servers. Short-lived addresses were not in use during this experiment.

4.2.5 Network captures

Incoming traffic is recorded with the tcpdump tool and captures are stored in the standard PCAP format that enables future analysis by tools. The command goes as follow:

```
tcpdump -i <INTERFACE> -G 6000 -w captures/<INTERFACE>-<NETWORK PREFIX>-%s-%F-%H_%M_%S.pcap
-s 0 -K -n -U
```

where:

- `<INTERFACE>` is the name of the network interface
- `-G 5400` is the rotation time in seconds of the capture file (the value had to be reasonably high in order not to create to many files)
- `<NETWORK PREFIX>` is the notation of the network prefix (e.g. 2001:db8:1::_48 for 2001:db8:1::/48)
- `%s-%F-%H_%M_%S` is a notation of the current date
- `-s 0` is the snapshot length. Here the default value 262144 bytes is set.
- `-K` specifies to not verify checksums
- `-n` address and port translations are disabled
- `-U` the captured packets are not buffered and are directly written on disk

Network captures were regularly retrieved from the virtual servers thanks to the rsync software.

4.2.6 Analysis of captures

Traffic analysis is done using Tshark [98] and Wireshark [99] tools. Filters had to be added in order to mask some specific packets. Here is a tshark command line used to inspect the capture files with a filter that hides local-link traffic: `tshark -r <pcap capture file name> -t ud -n -Y !(ipv6.src == fe80::/10)`

4.2.7 Honeypot services

To get a better understanding of what the probes are about, honeypot services were deployed. The Cowrie SSH/Telnet honeypot has been installed and configured to monitor SSH and Telnet usage on their default ports, respectively tcp/22 and tcp/23 and also tcp/2222 and tcp/2223 as they are often used as low privileges substitutes. IPv6 port redirection is done with the IPv6 version of IPTables (`ip6tables`) so that cowrie can run with low privileges.

```
ip6tables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222
ip6tables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --to-port 2223
```

A short program that behaves like an HTTP server was written. It returns the HTTP code 200 (OK) for any route called. There is an option to enable TLS. This programme is listening on ports tcp/80, tcp/8080 and tcp/443 (with TLS). Here is the `--help` output:

```
./http --help
Usage of ./http:
  -addr string
     addr to listen on (default ":8080")
  -cert string
     TLS certificate (default "cert.pem")
  -key string
     TLS key (default "key.pem")
  -tls
     serve HTTP over TLS
```

A self-signed TLS certificate was issued with openssl. The source code of this program can be found in Appendix 1. All honeypot services are up on the 19th of August 2018 and are still running as of this writing.

4.2.8 Evaluation

Evaluation will be done on the number of hits regarding the addresses probed and where they were advertised. A histogram will be generated in order to highlight the evolution of the number of probes before and after having advertised the addresses.

4.2.9 Risks

Incoming traffic could have been redirected to a single machine but VPS were kept isolated from one another in case of compromise. The IPv6 address space is so large that it might not be possible to see any reconnaissance probes on any of the hosts deployed. If so, contacting organisations with this type of data may have been considered as an alternative.

5 Results and Analysis

The results of the experiment are presented in this section. The first network packets were received on August 3rd, 2018. The experimentation is still ongoing at the time of this writing. Only the most relevant collected traffic is described here, the analysis is presented depending on the endpoint locations.

5.1 The Budapest native endpoint

On the 2018-08-09, probing patterns were observed on a gateway with a low-byte address from the Budapest endpoint from the native /56 IPv6 block. According to the WHOIS record of this single source address, it belongs to a German University network and seems to be used as part of the PlanetLab [100] project. Probing traffic was sent on ports tcp/80, tcp/443, udp/53 (AAAA DNS query) and udp/443 (QUIC [101]). ICMPv6 echo requests were sent as well. In most cases, short bursts of 7 requests were seen after the previous probing. Interestingly, the targeted ports are encoded in the source addresses (e.g. `2001:db8::1:80:6` for an address that is used to scan port tcp/80). This probing pattern is repeating itself multiple times almost every day until the termination of the VPS the 2018-08-30. No addresses in the /56 prefix were advertised and no traffic was received on any of the included addresses.

5.2 The Los Angeles tunnelled endpoint

On the 2018-08-19, a week after the tunnel creation, scanning patterns were observed from the Los Angeles endpoint who has a routed /48. A single IP address from a Romanian cloud service provider sent multiple TCP SYN packets on ports tcp/22 and tcp/80 of the tunnel client address. No further actions were undertake. Eight hours later, probes targeting the same ports were seen on addresses belonging to the /48 block. TCP SYN traffic was received on low-byte addresses from 0 to f on the last byte (e.g. `2001:db8::[0-f]`). This time, after the TCP handshake was completed on port 80, HTTP/1.1 GET requests were emitted. The Host value was set to `localhost` and the requested URI to `/`. Our honeypot service answered with the HTTP 200 code (OK) and an empty body. The described scanning technique is repeated on the third IPv6 subnet of the block (e.g. `2001:db8:2::[0-f]`). And then on the second (e.g. `2001:db8:1::[0-f]`). The next day, on the 2018-08-20, approximately 5 hours after that the last packet was received from this host, another burst of TCP SYN was emitted on port tcp/8080. The scanned range was the same as previously mentioned (`2001:db8:[0-2]::[0-f]`). HTTP/1.1 GET requests were also sent. Scanning bursts of subnets are usually taking a single second. Break times between subnets are in the range of one and a half to two hours.
On the 2018-08-21, ICMPv6 echo requests were observed on low-byte addresses from a single source address (`2a01:190:1709::2`). The WHOIS information of this address indicates that this scan is part of an academic research on IPv6 deployment carried by SBA Research and mentions a web page that presents the research experiment [59]. The source of the scanned IP addresses is mentioned and their probing technique is described. The scanned addresses are following this format: `2001:db8:0:[0-f]{4}::1`). The bold part appears to be randomly chosen.
On the 2018-08-28, seven UDP packets on port 80 were received on a low-byte address ending with 8 (e.g. `2001:db8::8`) belonging to the routed /48. The size of the packets is 80 bytes and the 32 bytes data section contains the following printable text:

```
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
```

According to the WHOIS information, the source address belongs to a European cloud computing company operating in Canada.
On the 2018-08-30 probing of the tunnel client address happened on port tcp/80 and tcp/443. Standard HTTP GET queries were emitted. The source IP address is from the China Education and Research

Network (CERNET) in Beijing, China and the scan appears to be for research purpose.
On the 2018-08-31 three ICMPv6 echo requests were emitted from a different source address with a different network prefix but are still belonging to CERNET and are linked to the Tsinghua University.
Different HTTP GET requests came from the first address a day later on port tcp/80. This time, the HTTP Host header was not the visited IP address as last time but a .top domain that have nothing to with the ongoing project. Eight requests mentioning this domain were made on different ports (80, 443) and URLs ending with `/deadurl.html` were sometimes added resulting in a 400 Bad Request response from the web server because this format is not allowed in the Host header. The different .top subdomains are pointing to IPv4 addresses in IP ranges allocated to Aliyun Computing Co and Alibaba.com LLC. An hour later ICMPv6 echo requests were emitted on the tunnel client address and on low-byte addresses of the routed /64 block that comes by default with the tunnel `2001:db8:d:0::[0-f]`
On the 2018-09-01, the same scanning patterns that came from the Romanian IP address appeared again.
On the 2018-09-09, ten TCMPv6 echo requests were captured on the first address of the /48 subnet. According to the WHOIS information, the source IP address belongs to the Helsinki University of Technology in Finland.

5.3 The London tunnelled endpoint

On the 2018-08-05, 2018-08-06 and the 2018-08-19, the same TCP SYN scanning patterns on port tcp/22, tcp/80 and tcp/8080 from the Romanian cloud service provider were captured 5.2.
On the 2018-08-07, 61 UDP packets were received from the same address belonging to the European cloud computing company operating in Canada mentioned in 5.2 and with the same content.
From the 2018-08-21, a traffic similar to the one mentioned above 5.2 from SBA Research was observed.
On the 2018-08-29, after running the bitcoin core node for the first time, the tcpdump filter was updated to discard Bitcoin traffic on the default port 8333 as well as ICMPv6 echo request and reply in order to avoid overloading the disk space of the virtual server.
The same day, TCP SYN probes on port tcp/80 and tcp/443 were observed from one host with an address from the Japanese WIDE project [102].
The same probing was performed twice the next day by the same host.
Unfortunately, a mistake was made in the infrastructure bootstrapping script which stopped the recording of the traffic for this endpoint after the 2018-09-01.

5.4 The New York tunnelled endpoint

The observation was carried until the 2018-09-07.
On the 2018-08-19, 2018-09-01, 2018-09-02, and 2018-09-03 the same TCP SYN scanning patterns on port tcp/22, tcp/80 and tcp/8080 from the Romanian cloud service provider were captured 5.2.
Here is the number of HTTP requests per registered domains:

- The address linked to the .net domain received 13 HTTP GET requests from Amazon EC2, AT&T Internet Services, the Austrian EDIS GmbH company, MFC LLC, Google bot.
- The address linked to the .fr domain received 14 HTTP GET and 1 HTTP POST requests from a Polish virtual machine provider, an Irish hosting company, ColoCrossing, Google bot.
- The address linked to the .technology domain received 18 HTTP GET requests from Facebook, the Austrian EDIS GmbH company, OVH, Verisig Labs.
- The address linked to the .science domain received 2 HTTP GET requests from VeriSign Infrastructure & Operations.

On average, less than two days after adding an IPv6 address to a domain name, HTTP requests were already appearing. The vast majority of those requests were on port tcp/80, only 4 were using TLS on port tcp/443. The Uniform Resource Identifier (URI) of the HTTP URLs were similar to the following: `/wp-login.php`, `/robots.txt`, `/favicon.ico`, `/libraries/sfn.php`, `/index.php?wap2`, `/smf/index.php?wap2`, `/MessageBoard/index.php?wap2`, `/Forums/index.php?wap2` and `/xmlrpc.php?rsd`.

One of the GET requests on the .fr domain had this User-Agent header:

```
}__test|O:21:"JDatabaseDriverMysqli":3:{s:4:"\0\0\0a";O:17:"
    JSimplepieFactory":0:{}s:21:"\0\0\0disconnectHandlers";a:1:{i:0;a:2:{i
    :0;O:9:"SimplePie":5:{s:8:"sanitize";O:20:"JDatabaseDriverMysql":0:{}s
    :5:"cache";b:1;s:19:"cache_name_function";s:6:"assert";s:10:"javascript
    ";i:9999;s:8:"feed_url";s:54:"eval(base64_decode($_POST[111]));JFactory
    ::get();exit;";}i:1;s:4:"init";}}s:13:"\0\0\0connection";i:1;}
```

It seems to be the payload of an old Joomla Unserialize Vulnerability [103]. The only one POST request had the following form item (in one line):

```
111=JHBocG9zPSJsaW51eCI7aWYgKHN0cnRvdXBwZXIoc3Vic3RyKFBIUF9PUywwLDMpKT09Ild
JTiIpeyRwaHBvcz0id2luIjt9OyRjdWRpcj1kaXJuYW1lKF9fRklMRV9fKTtpZiAoJHBocG9zPT
0id2luIikgeyAkZm49dXJsZW5jb2RlKCRjdWRpcik7ICRmbiA9IHN0cl9yZXBsYWNlKCIlNUMiL
CIlMkYlMkYiLCRmbik7JGZuID0gc3RyX3JlcGxhY2UoIiUyRiUyRnNpbXBsZXBpZSIsIiIsJGZu
KTsgJGZuID0gJGZuLiIvL3Nmbi5waHAiO31pZiAoJHBocG9zPT0ibGludXgiKSB7ICRmbj11cmx
lbmNvZGUoJGN1ZGlyKTskZm4gPSBzdHJfcmVwbGFjZSgiJTJGc2ltcGxlcGlleiIsIiIsJGZuKTy
AkZm4gPSAkZm4uIi9zZm4ucGhwIjt9ICRmbj11cmxkZWNvZGUoJGZuKTtmaWxlX3B1dF9jb250Z
W50cygkZm4sYmFzZTY0X2RlY29kZSgiUEQ5d2FIQWdJSEJ5WldkZmNtVndiR0ZqWlNnaUwyeGhk
R1Z5WVdsdUwyVWlMQ0FpWlhZaUxpSmhiQ2duSWk0a1gxSkZVVlZGVTFSYkoyWjFZMnQ1YjNVME1
6SXhKMTB1SWljcElpd2dJbXhoZEdWeVlXbHVJSFJsYzNScGJqa2lLVHNnUHo0NU9EUXpNREE9Ii
kpOw%3D%3D
```

Nothing has been found regarding the aim of this parameter.
On the 2018-09-02, 2018-09-03 and 2018-09-04 traffic from the German University mentioned in 5.1 was captured. Only TCP SYN on port tcp/80 and ICMPv6 echo requests were sent on what appear to be randomly generated addresses within the /48 block. About 33 addresses were selected each time.

5.5 The other tunnelled endpoints

Traffic from the Romanian cloud provider was noticed on almost all the deployed tunnel endpoints (Toronto, Sydney, New York, Los Angeles, London, Tokyo, Paris and two of the Ukrainian endpoints).
Traffic from the PlanetLab project presented in 5.1 also appeared on the native /48 London block.
UDP traffic on port udp/80 from the European cloud computing company operating in Canada mentioned in 5.2 was seen from time to time with a small number of packets (from 2 to 14) from Tokyo, London (tunnel), Russia and Paris.
Traffic from the SBA Research scanning experiment mentioned in 5.2 was seen from native Russia IPv6 block, Paris, and two of the Ukrainian endpoints.

6 Discussion and Evaluation

Discussion The experiment showed that, for the time being, the majority of large-scale scans were originating from the academic research community. Nevertheless, one actor is heavily scanning a lot of prefixes from a single source IP address. The scans are mostly performed with HTTP (port 80) and ICMPv6 protocols. The advertising part on search engines for Internet-connected devices was not successful since no probes were observed on any of the addresses promoted. Due to a configuration mistake, no traffic was recorded after the deployment of the Bitcoin node. On the other hand, all four domain names with IPv6 addresses linked to them were visited and this without the need for advertising the domain names. This proves that an active zone file retrieval is performed by several actors. One web exploit has been spotted on an IPv6 address linked to a domain name. More traffic was expected after the request for an SSL certificate that was available from the Certificate Transparency logs. No traffic was observed after the NTP advertisement. So far, no scan on port tcp/23 (Telnet) has been seen which is interesting since botnets like Mirai are often targeting this port.
A lot of research is being invested in IPv6 and the changes brought about by this protocol, particularly in the field of reconnaissance but also its inherent security and privacy. The experimental phase revealed as a side observation that not all VPS providers are configuring their network very well. Packets that should not be visible by others are not filtered, or not properly routed by the software used by such providers. Those packets mostly involve DNS queries and SSH connections. The RFCs seem well thought out and the advice is to follow them. Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS) are still required to detect scanning and exfiltration of data across networks. Filtering strategies must be carefully designed and implemented for both IPv4 and IPv6.

Evaluation The extensive literature review to get up to speed on the IPv6 protocols (ICMPv6, NDP, DHCPv6), their security and the botnets world has been long and tedious. The scope of the subject would have benefited from being reduced in order to provide more detailed results on a high value-added part such as the experimentation carried out. Nevertheless, the literature review conducted in parallel with the experiment was beneficial in order to take into account the latest advances in the field of reconnaissance.

7 Conclusion and Suggestions for Future Research

Conclusion This project has shown that active gathering and scanning of IPv6 addresses is performed by multiple actors, mostly by the academic research. More and more IPv6 gathering techniques are described in the literature, smart scanning techniques are developed relying on target generation algorithms. IPv6 covert channels can be leveraged for stealth communications as proof-of-concepts demonstrated. Fast address hoping and transition mechanisms will make attribution and correlation much harder. The Moving Target IPv6 Defence seems a very promising approach both for the defensive side but also for the attacking side.
We are still in the early ages of IPv6 botnets according to the captured traffic but more evidence need to be gathered over time. The overall feeling that emerges from all this research is a sense of progress in the way IPv6 is being tackled by the whole community.

Knowledge remains the best defence.

Suggestions for Future Research Future research will be for the most part focused on the experiment. A more robust and monitored infrastructure is needed. Address advertising needs to be done in all the listed sources of addresses listed. In particular, DNS records have to be inserted in popular datasets to widen the reach of the linked addresses. Be part of the BitTorrent peer-to-peer network and be referenced by multiple trackers is an other way to gather client addresses. Relevant information for this may be found in this memo [104]. Scripts for automating the set-up of the infrastructure have been written but they need to be more generic before being released. Also, it would be interesting to see how many IPv6 client addresses can be gathered from Ad-campaigns. Crawling bots in a botnet can as well be used to discover new client addresses :p

8 References

[1] OVH. The ddos that didnt break the camels vac. https://www.ovh.com/world/a2367.the-ddos-that-didnt-break-the-camels-vac, 2016. Last accessed: 2018-09-05.

[2] Sam Kottler. February 28th ddos incident report. https://githubengineering.com/ddos-incident-report/, 2018. Last accessed: 2018-09-05.

[3] Carlos Morales. Netscout arbor confirms 1.7 tbps ddos attack; the terabit attack era is upon us. https://asert.arbornetworks.com/netscout-arbor-confirms-1-7-tbps-ddos-attack-terabit-attack-era-upon-us/, 2018. Last accessed: 2018-09-05.

[4] unixfreaxjp. Mmd-0059-2016 - linux/irctelnet (new aidra) - a ddos botnet aims iot w/ ipv6 ready. http://blog.malwaremustdie.org/2016/10/mmd-0059-2016-linuxirctelnet-new-ddos.html, 2016. MalwareMustDie!, Last accessed: 2018-09-11.

[5] Jon Postel. Internet Protocol. RFC 791, September 1981.

[6] Bob Hinden and Dr. Steve E. Deering. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, December 1998.

[7] Qing Li, Chris Larsen, and Tim van der Horst. Ipv6-a catalyst and an evasion tool for botnets and malware distribution networks. *Computer*, page 1, 2012.

[8] Code24. Building better botnets with ipv6. https://www.youtube.com/watch?v=irB_Id8oZGA, 2014. Derbycon 4, Last accessed: 2018-09-11.

[9] Nazrul Hoque, Dhruba K Bhattacharyya, and Jugal K Kalita. Botnet in ddos attacks: Trends and challenges. *IEEE Communications Surveys and Tutorials*, 17(4):2242–2270, 2015.

[10] Kishore Angrishi. Turning internet of things (iot) into internet of vulnerabilities (iov): Iot botnets. *arXiv preprint arXiv:1702.03681*, 2017.

[11] Nathan Goodman. A survey of advances in botnet technologies. *arXiv preprint arXiv:1702.01132*, 2017.

[12] Dennis Andriesse, Christian Rossow, Brett Stone-Gross, Daniel Plohmann, and Herbert Bos. Highly resilient peer-to-peer botnets are here: An analysis of gameover zeus. In *2013 8th International Conference on Malicious and Unwanted Software:*, pages 116–123. IEEE, 2013.

[13] Diogo Mónica and Carlos Ribeiro. Leveraging honest users: Stealth command-and-control of botnets. In *WOOT*, 2013.

[14] Emmanouil Vasilomanolakis, Jan Helge Wolf, Leon Böck, Shankar Karuppayah, and Max Mühlhäuser. I trust my zombies: A trust-enabled botnet. *arXiv preprint arXiv:1712.03713*, 2017.

[15] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1093–1110, Vancouver, BC, 2017. USENIX Association.

[16] G. Kambourakis, C. Kolias, and A. Stavrou. The mirai botnet and the iot zombie armies. In *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, pages 267–272, Oct 2017.

[17] Fariba Haddadi, Duc Le Cong, Laura Porter, and A Nur Zincir-Heywood. On the effectiveness of different botnet detection approaches. In *Information Security Practice and Experience*, pages 121–135. Springer, 2015.

[18] Christian Rossow, Dennis Andriesse, Tillmann Werner, Brett Stone-Gross, Daniel Plohmann, Christian J Dietrich, and Herbert Bos. Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 97–111. IEEE, 2013.

[19] David Holder. Ipv6 security fundamentals. `http://www.ipv6.org.uk/wp-content/uploads/sites/204/2017/07/Holder-ipv6-security-2017.pdf`, 2017. Erion, Last accessed: 2018-09-11.

[20] Eric Vyncke. Ipv6 security - a quick overview. `https://indico.cern.ch/event/592622/contributions/2576502/attachments/1475412/2286370/20170613_ipv6_security_hepsysman.pdf`, 2017. Cisco, Last accessed: 2018-09-11.

[21] RIPE NCC. Training course material. `https://www.ripe.net/support/training/cd/20180510_RIPENCC_TrainingMaterial.zip`, 2018. Last accessed: 2018-09-07.

[22] Jakub Czyz, Matthew J Luckie, Mark Allman, and Michael Bailey. Don't forget to lock the back door! a characterization of ipv6 network security policy. In *NDSS*, 2016.

[23] Johanna Ullrich, Katharina Krombholz, Heidelinde Hobel, Adrian Dabrowski, and Edgar Weippl. Ipv6 security: Attacks and countermeasures in a nutshell. In *8th USENIX Workshop on Offensive Technologies (WOOT 14)*, San Diego, CA, 2014. USENIX Association.

[24] Thomas Narten, Richard Draves, and Suresh Krishnan. Privacy extensions for stateless address autoconfiguration in ipv6. Technical report, 2007. RFC 4941.

[25] F Gont and T Chown. Network reconnaissance in ipv6 networks. Technical report, IETF, 2016. RFC 7707.

[26] Fernando Gont. A method for generating semantically opaque interface identifiers with ipv6 stateless address autoconfiguration (slaac). Technical report, 2014. RFC 7217.

[27] Tuomas Aura. Cryptographically generated addresses (cga). Technical report, 2005. RFC 3972.

[28] Fernando Gont, Alissa Cooper, Dave Thaler, and Will Liu. Recommendation on stable ipv6 interface identifiers. Technical report, 2017. RFC 8064.

[29] Johanna Ullrich, Peter Kieseberg, Katharina Krombholz, and Edgar Weippl. On reconnaissance with ipv6: a pattern-based scanning approach. In *Availability, Reliability and Security (ARES), 2015 10th International Conference on*, pages 186–192. IEEE, 2015.

[30] Pawel Foremski, David Plonka, and Arthur Berger. Entropy/ip: Uncovering structure in ipv6 addresses. In *Proceedings of the 2016 Internet Measurement Conference*, pages 167–181. ACM, 2016.

[31] Oliver Gasser, Quirin Scheitle, Sebastian Gebhard, and Georg Carle. Scanning the ipv6 internet: towards a comprehensive hitlist. *arXiv preprint arXiv:1607.05179*, 2016.

[32] Austin Murdock, Frank Li, Paul Bramsen, Zakir Durumeric, and Vern Paxson. Target generation for internet-wide ipv6 scanning. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, pages 242–253, New York, NY, USA, 2017. ACM.

[33] Tobias Fiebig, Kevin Borgolte, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. Something from nothing (there): collecting global ipv6 datasets from dns. In *International Conference on Passive and Active Network Measurement*, pages 30–43. Springer, 2017.

[34] Oliver Gasser, Benjamin Hof, Max Helm, Maciej Korczynski, Ralph Holz, and Georg Carle. In log we trust: Revealing poor security practices with certificate transparency logs and internet measurements. In *International Conference on Passive and Active Network Measurement*, pages 173–185. Springer, 2018.

[35] Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczynski, Stephen D Strowes, Luuk Hendriks, and Georg Carle. Clusters in the expanse: Understanding and unbiasing ipv6 hitlists. *arXiv preprint arXiv:1806.01633*, 2018.

[36] Robert Beverly, Ramakrishnan Durairajan, David Plonka, and Justin P Rohrer. In the ip of the beholder: Strategies for active ipv6 topology discovery. *arXiv preprint arXiv:1805.11308*, 2018.

[37] Kevin Borgolte, Shuang Hao, Tobias Fiebig, and Giovanni Vigna. Enumerating active ipv6 hosts for large-scale security scans via dnssec-signed reverse zones. In *Enumerating Active IPv6 Hosts for Large-scale Security Scans via DNSSEC-signed Reverse Zones*, page 0. IEEE, 2018.

[38] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D Strowes, and Narseo Vallina-Rodriguez. A long way to the top: Significance, structure, and stability of internet top lists. *arXiv preprint arXiv:1805.11506*, 2018.

[39] Norka B. Lucena, Grzegorz Lewandowski, and Steve J. Chapin. Covert channels in ipv6. In George Danezis and David Martin, editors, *Privacy Enhancing Technologies*, pages 147–166, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[40] Thomas Graf. Messaging over ipv6 destination options. `http://gray-world.net/papers/messip6.txt`, 2013. Last accessed: 2018-09-11.

[41] R.P. Murphy. v00d00n3t - ipv6/icmpv6 covert channel. `https://dc414.org/download/confs/defcon14/DC-14-Presentations/DC-14-Murphy.pdf`, 2014. Last accessed: 2018-09-11.

[42] Bernhards Blumbergs, Mauno Pihelgas, Markus Kont, Olaf Maennel, and Risto Vaarandi. Creating and detecting ipv6 transition mechanism-based information exfiltration covert channels. In Billy Bob Brumley and Juha Röning, editors, *Secure IT Systems*, pages 85–100, Cham, 2016. Springer International Publishing.

[43] Marc Heuse. Thc ipv6 attack toolkit. `https://github.com/vanhauser-thc/thc-ipv6`. Last accessed: 2018-09-11.

[44] Fernando Gont. Si6 networks' ipv6 toolkit. `https://www.si6networks.com/tools/ipv6toolkit/`. Last accessed: 2018-09-11.

[45] Antonios Atlasis. Chiron. `https://github.com/aatlasis/chiron`. Last accessed: 2018-09-11.

[46] Fyodor Vaskovich. Nmap. `https://nmap.org`. Last accessed: 2018-09-11.

[47] Owen Russell Hardman. *Optimizing a network layer moving target defense by translating software from python to c*. PhD thesis, Virginia Tech, 2016.

[48] Zakir Durumeric. Zmap. `https://github.com/zmap/zmap`. Last accessed: 2018-09-11.

[49] Lennart Poettering. mdns-scan. `http://0pointer.de/lennart/projects/mdns-scan/`, 2008. Last accessed: 2018-09-11.

[50] Antonios Atlasis. Pholus. https://github.com/aatlasis/Pholus. Last accessed: 2018-09-11.

[51] Aaron Bockover. mzclient. https://github.com/mono/Mono.Zeroconf/tree/master/src/MZClient, 2008. Last accessed: 2018-09-11.

[52] Matthew Luckie. Scamper. https://www.caida.org/tools/measurement/scamper/, 2018. Last accessed: 2018-09-11.

[53] Philippe Biondi. Scapy. https://scapy.net/. Last accessed: 2018-09-11.

[54] Ettercap. Ettercap. https://www.ettercap-project.org/index.html. Last accessed: 2018-09-11.

[55] Jakub Czyz, Kyle Lady, Sam G. Miller, Michael Bailey, Michael Kallitsis, and Manish Karir. Understanding ipv6 internet background radiation. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 105–118, New York, NY, USA, 2013. ACM.

[56] Netlab 360. Ddosmon. https://ddosmon.net/insight/. Last accessed: 2018-09-11.

[57] Fernando Gont, Jen Linkova, Tim Chown, and Will Liu. Observations on the dropping of packets with ipv6 extension headers in the real world. Technical report, 2016. RFC 7872.

[58] E Davies and J Mohacsi. Recommendations for filtering icmpv6 messages in firewalls. Technical report, 2007. RFC 4890.

[59] SBA Research. Academic research on ipv6 deployment. https://scanning.sba-research.org/, 2018. Last accessed: 2018-09-09.

[60] Dan Goodin. Using ipv6 with linux? you've likely been visited by shodan and other scanners. https://arstechnica.com/information-technology/2016/02/using-ipv6-with-linux-youve-likely-been-visited-by-shodan-and-other-scanners/, 2016. Ars Technica, Last accessed: 2018-09-11.

[61] Matthew Bryant. Tldr. https://github.com/mandatoryprogrammer/TLDR, 2018. Last accessed: 2018-09-09.

[62] ICANN. Centralized zone data service. https://czds.icann.org. Last accessed: 2018-09-11.

[63] University of Oregon. Routeviews. www.routeviews.org, 2018. Last accessed: 2018-09-09.

[64] Center for Applied Internet Data Analysis. Routeviews prefix to as mappings dataset (pfx2as) for ipv4 and ipv6. https://www.caida.org/data/routing/routeviews-prefix2as.xml, 2018. Last accessed: 2018-09-09.

[65] John Matherly. Shodan. https://www.shodan.io/. Last accessed: 2018-09-09.

[66] Steven M Bellovin, Bill Cheswick, and Angelos Keromytis. Worm propagation strategies in an ipv6 internet. *LOGIN: The USENIX Magazine*, 31(1):70–76, 2006.

[67] Jari Arkko, James Kempf, Brian Zill, and Pekka Nikander. Secure neighbor discovery (send). Technical report, 2005.

[68] Marek Šimon, Ladislav Huraj, and Marián Host'oveckỳ. Ipv6 network ddos attack with p2p grid. *Creativity in Intelligent, Technologies and Data Science*, pages 407–415, 2015.

[69] E Levy-Abegnoli, G Van de Velde, C Popoviciu, and J Mohacsi. Ipv6 router advertisement guard. Technical report, 2011. RFC 6105.

[70] Fernando Gont, Will Liu, and G Van de Velde. Dhcpv6-shield: Protecting against rogue dhcpv6 servers. Technical report, 2015. RFC 7610.

[71] Seiichi Kawamura and Masanobu Kawashima. A recommendation for ipv6 address text representation. Technical report, 2010. RFC 5952.

[72] A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC). RFC 7217, 2014.

[73] Matthew Dunlop, Stephen Groat, William Urbanski, Randy Marchany, and Joseph Tront. Mt6d: A moving target ipv6 defense. In *Military Communications Conference, 2011-Milcom 2011*, pages 1321–1326. IEEE, 2011.

[74] Stephen Groat, Matthew Dunlop, William Urbanksi, Randy Marchany, and Joseph Tront. Using an ipv6 moving target defense to protect the smart grid. In *Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES*, pages 1–7. IEEE, 2012.

[75] Dileep Kumar Basam. *Strengthening MT6D Defenses with Darknet and Honeypot capabilities*. PhD thesis, Virginia Tech, 2015.

[76] Christopher Morrell, Reese Moore, Randy Marchany, and Joseph G. Tront. Dht blind rendezvous for session establishment in network layer moving target defenses. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, MTD '15, pages 77–84, New York, NY, USA, 2015. ACM.

[77] J. Sagisi, J. Tront, and R. Marchany. System architectural design of a hardware engine for moving target ipv6 defense over ieee 802.3 ethernet. In *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, pages 551–556, Oct 2017.

[78] K. Zeitz, M. Cantrell, R. Marchany, and J. Tront. Changing the game: A micro moving target ipv6 defense for the internet of things. *IEEE Wireless Communications Letters*, 7(4):578–581, Aug 2018.

[79] RIPE NCC. Ipv6 address allocation and assignment policy. `https://www.ripe.net/publications/docs/ripe-707`, 2018. Last accessed: 2018-09-11.

[80] Thomas Narten, Geoff Huston, and Lea Roberts. Ipv6 address assignment to end sites. Technical report, 2011. RFC 6177.

[81] Mike Leber. Hurricane electric ipv6 tunnel broker. `https://tunnelbroker.net/`. Last accessed: 2018-09-11.

[82] Netassist ipv6 tunnel broker. `http://tb.netassist.ua/`. Last accessed: 2018-09-11.

[83] Piotr Pabian. Pemsy ipv6 tunnel broker. `https://www.pemsy.com`. Last accessed: 2018-09-11.

[84] ip6.im. `http://ip6.im/`. Last accessed: 2018-09-11.

[85] Addy Yeow. Bitnodes. `https://bitnodes.earn.com/`. Last accessed: 2018-09-11.

[86] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. *arXiv preprint arXiv:1801.03998*, 2018.

[87] Satoshi Nakamoto. Bitcoin core. `https://bitcoin.org/en/download`. Last accessed: 2018-09-11.

[88] Zakir Durumeric. Censys. `https://censys.io/`. Last accessed: 2018-09-11.

[89] Zoomeye. https://www.zoomeye.org/. Last accessed: 2018-09-11.

[90] VeriSign. Tld zone file access. https://www.verisign.com/en_US/channel-resources/domain-registry-products/zone-file/index.xhtml. Last accessed: 2018-09-11.

[91] IIS. Access to zonefiles for .se and .nu. https://www.iis.se/english/domains/tech/zonefiles/, 2016. Last accessed: 2018-09-11.

[92] VeriSign. Verisign domain name report q2 2018. https://www.verisign.com/assets/domain-name-report-Q22018.pdf, 2018. Last accessed: 2018-09-11.

[93] Matthew Holt. Caddy web server. https://caddyserver.com/. Last accessed: 2018-09-11.

[94] Internet Security Research Group. Let's encrypt. https://letsencrypt.org/, 2018. Last accessed: 2018-09-11.

[95] Comodo CA Limited. crt.sh. https://crt.sh. Last accessed: 2018-09-11.

[96] Google. Google transparency report. https://transparencyreport.google.com/https/certificates. Last accessed: 2018-09-11.

[97] Wido den Hollander. Anyip: bind a whole subnet to your linux machine. https://blog.widodh.nl/2016/04/anyip-bind-a-whole-subnet-to-your-linux-machine/, 2016. Last accessed: 2018-09-11.

[98] Wireshark. Tshark. https://www.wireshark.org/docs/man-pages/tshark.html. Last accessed: 2018-09-11.

[99] Wireshark. Wireshark. https://www.wireshark.org/. Last accessed: 2018-09-11.

[100] The PlanetLab Consortium. Planetlab. https://www.planet-lab.org/. Last accessed: 2018-09-11.

[101] Google. Quic, a multiplexed stream transport over udp. https://www.chromium.org/quic. Last accessed: 2018-09-11.

[102] Jun Murai. Wide project. http://www.wide.ad.jp/index_e.html, 2018. Last accessed: 2018-09-11.

[103] Pasha Kravtsov. A different kind of pop: The joomla unserialize vulnerability. https://blog.cloudflare.com/the-joomla-unserialize-vulnerability/, 2015. Last accessed: 2018-09-11.

[104] Eric Vyncke and Martin Defeche. Measuring ipv6 traffic in bittorrent networks. 2012.

9 Appendices

9.1 Simple HTTP server source code

```
package main

import (
        "flag"
        "log"
        "net"
        "net/http"
)

var (
        addr     = flag.String("addr", ":8080", "addr to listen on")
        useTLS   = flag.Bool("tls", false, "serve HTTP over TLS")
        certPath = flag.String("cert", "cert.pem", "TLS certificate")
        keyPath  = flag.String("key", "key.pem", "TLS key")
)

func main() {
        flag.Parse()
        l, err := net.Listen("tcp6", *addr)
        if err != nil {
                log.Fatalf("could not listen on addr '%s': %s\n", *addr, err)
        }
        defer l.Close()

        http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {
                w.WriteHeader(http.StatusOK)
        })

        if *useTLS {
                log.Fatalf("serving https error: %s\n", http.ServeTLS(l, nil, *certPath, *keyPath))
        } else {
                log.Fatalf("serving http error: %s\n", http.Serve(l, nil))
        }
}
```