IPv6 Covert Channels in the Wild

Wojciech Mazurczyk
Krystian Powójski
w.mazurczyk@tele.pw.edu.pl
krystian.powojski@gmail.com
Warsaw University of Technology
Warsaw, Poland

Luca Caviglione
luca.caviglione@ge.imati.cnr.it
National Research Council of Italy
Genoa, Italy

ABSTRACT

The increasing diffusion of malware endowed with steganographic techniques requires to carefully identify and evaluate a new set of threats. The creation of a covert channel to hide a communication within network traffic is one of the most relevant, as it can be used to exfiltrate information or orchestrate attacks. Even if network steganography is becoming a well-studied topic, only few works focus on IPv6 and consider real network scenarios. Therefore, this paper investigates IPv6 covert channels deployed in the wild. Also, it presents a performance evaluation of six different data hiding techniques for IPv6 including their ability to bypass some intrusion detection systems. Lastly, ideas to detect IPv6 covert channels are presented.

CCS CONCEPTS

• **Security and privacy → Network security**; *Distributed systems security*; *Information flow control*; Pseudonymity, anonymity and untraceability.

KEYWORDS

IPv6, information hiding, network covert channels, network steganography

ACM Reference Format:
Wojciech Mazurczyk, Krystian Powójski, and Luca Caviglione. 2019. IPv6 Covert Channels in the Wild. In *CECC '19: Central European Cybersecurity Conference, Nov. 14–15, 2019, Munich, DE*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/xxxxxxx.xxxxxxxx

1 INTRODUCTION

Nowadays, information hiding is increasingly applied to network traffic to perform a wide range of tasks. For instance, steganographic techniques are used to watermark network flows for tracing how data propagate through the Internet [7] or to improve the privacy of users wanting to bypass censorships or blocks [15]. However, one of the most important usage trends of information hiding deals with the development of malware able to remain unnoticed for a long time. A prime technique exploits network covert channels, which are used to exfiltrate data or communicate in a stealthy manner with a remote command & control server [12]. As a consequence, analyzing the risks of information hiding when used with network traffic is mandatory to fully assess the cybersecurity of the Internet. To this aim, literature abounds of works investigating the different features of the traffic (e.g., unused fields in the header of packets, ambiguities of protocols or timing behaviors of a flow) that can be used as carriers for embedding secrets [11], [19], [20].

In general, the majority of works dealing with network steganography focuses on IPv4 and only hints that many techniques can be also applied to IPv6. A notable exception is [9], where authors analyzed 22 covert channels that can be created by directly injecting data within the header of IPv6 or manipulate some protocol behaviors. Also [8] and [10] focus on IPv6 but instead of proposing novel steganographic methods, they discuss the development of mechanisms to block covert channels or limit their capabilities. Unfortunately, none of the works considering IPv6 provide a thorough evaluation of the proposed covert channels [8, 9]. Moreover, [13] hints at the possibility of exploiting IPv6 even if its popularity is limited compared to IPv4, while [17] only reviews the security hazards of IPv6, including those arising from the injection of hidden data to create covert channels. At the best of our knowledge, the work in [3] is the only one providing a performance evaluation of the proposed information hiding approaches. However, it focuses on detecting exfiltration attempts using transitional mechanisms rather than functionalities of IPv6.

Therefore, our work aims at filling such a gap by evaluating the feasibility of deploying in the wild some covert channels targeting IPv6 proposed so far [8, 9, 13, 17]. In fact, the behavior of IPv6 traffic has an impact in terms of "capacity" that can be used to hide information and its popularity also plays a major role in terms of stealthiness. Besides, the presence of a mixed set of middleboxes (e.g., network address translation), protocol implementations, and transitional mechanisms [18] could alter the expected functioning of IPv6 in a manner difficult to predict and cause the disruption of the covert channel. Lastly, advancements in the development of tools for network security and intrusion detection systems could partially void the original vision [4], as IPv6 covert channels were initially introduced back to 2006 [9].

Summing up, the contributions of the papers are threefold: *i*) investigate IPv6 traffic to evaluate its real suitability for acting as a carrier for network covert channels, *ii*) evaluate the feasibility and the performances of IPv6-based covert channels, and *iii*) assess the impact of modern security tools in terms of their ability to stop steganographic threats.

The rest of the paper is structured as follows. Section 2 reviews IPv6 covert channels, while Section 3 investigates traffic features

CECC '19, Nov. 14–15, 2019, Munich, DE

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/xxxxxxx.xxxxxxxx

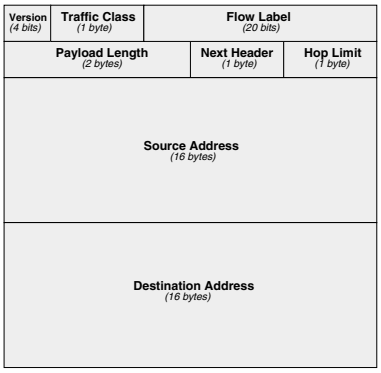

Figure 1: Header of the IPv6 protocol (borrowed from [9]).

that can be used to embed data. Section 4 showcases the performance evaluation of some IPv6 covert channels and proposes some countermeasures. Lastly, Section 5 concludes the paper and hints at some possible future developments.

2 IPV6-BASED COVERT CHANNELS

Originally presented in [5] and [6], IPv6 has been designed to enhance IPv4 in different areas, e.g., mobility, security and addressing. However, the deployment of IPv6 is mainly driven by its 128 bit long address space allowing to recover to issues caused by the depletion of IPv4 addresses. Due to the slow rollout of IPv4, the two protocols are expected to cohabit for a long period, thus proper transitional mechanisms have been proposed [16]. Concerning network covert channels targeting IPv6, references [9] and [8] show several steganographic methods embedding data in the header or in additional/optional extensions. To evaluate the feasibility of using IPv6 covert channels in the wild, we selected 6 methods targeting the header, which is depicted in Figure 1. The used fields and the related hiding mechanisms are described as follows.

(1) *Traffic Class*: it is a 8 bit long field specifying the service expected from the network. The first 6 bits define the Differentiated Services Code Point (DSCP) and classify the traffic according to quality criteria. The remaining 2 bits are used for Explicit Congestion Notification (ECN) for managing the flow in an end-to-end flavor. The information contained in the Traffic Class can be replaced with hidden data to set a covert channel with a bandwidth of 8 bit/packet. This field can be altered by intermediate nodes, thus disrupting the covert channel.
(2) *Flow Label*: it is 20 bit long and helps network nodes to route traffic towards the most appropriate path [1]. In general, labels should be pseudo-random and future values should not be predictable. Intermediate nodes should not switch labels as to not disrupt the flow. Part of the bits composing the Flow Label can be replaced with hidden data leading to a covert channel with a capacity of 20 bit/packet.
(3) *Payload Length*: it defines the size of the data field of the datagram, which can be up to 65,536 bytes. Information can be hidden by manipulating the Payload Length as to append arbitrary data to the payload. To avoid misbehavior of the IPv6 protocol, the checksum has to be properly updated as to prevent that packets will be discarded by intermediate nodes. Also, the hidden information should be removed before the datagram is delivered to the receiver. The bandwidth of the covert channel varies according to the amount of embedded data, which cannot exceed the maximum size of the datagram.
(4) *Next Header*: it identifies the next header that is present in the payload of the packet. Typical values are: 6 - TCP, 58 - ICMPv6, 17 - UDP, and 1 - ICMP. In case of extensions, the most common specific values are: 0 - Hop-by-Hop, 44 - Fragment, 60 - Destination, 51 - Authentication, and 43 - Routing. The information can be hidden by adjusting the Next Header as to point to a "fictitious" extra header containing data. Similarly to the previous case, the IPv6 datagram has to be properly restored before it is delivered to the destination. The resulting bandwidth varies according to the size of the fake headers injected.
(5) *Hop Limit*: it defines the maximum number of "hops", i.e., nodes that the packet can traverse. Since it is 8 bit long, the Hop Limit can have up to 256 values. Data can be hidden by increasing or decreasing the value of the field for consecutive packets. The information is then decoded by comparing the received values (if different routes did not disrupt the secret). As a result, secret information can be sent with a rate of 1 bit/packet.
(6) *Source Address*: it contains the network address for the source. The hidden information is inserted by replacing some bits of the address with arbitrary data. The maximum capacity of the resulting covert channel is 128 bit/packet.

In the following, we will also refer to the covert channels by using their number (e.g., (2) for the method using the Flow Label field). We point out that discussing all the internals of IPv6 as well as its addressing scheme is outside the scope of this paper and more details can be found in [1, 5, 6].

3 CAPACITY ANALYSIS OF IPV6 TRAFFIC

In general, the performances of an information hiding method heavily depend on the availability of a suitable carrier. As an example, injecting data in a field within the header will lead to a covert channel with a bandwidth proportional to the packet rate, i.e., a x bit/packet injected in a flow of y packet/s. Moreover, the steganogram (i.e., the carrier plus the embedded message) should not appear as an anomaly. For instance, for the case of channels presented in Section 2, fields containing hidden data should not deviate too much from average values as to not void the stealthiness of the covert channel. Understanding the behavior of the overt traffic is also critical to engineer suitable detection techniques or countermeasures. Alas, prior works did not quantify the capacity in terms of hidden messages or the stealthiness of real IPv6 traffic. To this aim, we investigated traffic captures collected on a backbone link (Tier 1 link) between Chicago and Seattle in four different days and made available by Center for Applied Internet Data Analysis[1]. To process data, we used custom Python scripts, the Scapy library and `tshark`.

[1]CAIDA UCSD Anonymized Internet Traces 2016 - Used traces: Jan. 21st, Feb. 18th, March 17th, and April 6th. Available online: https://www.caida.org/data/passive/passive_2016_dataset.xml [Last Accessed: July 2019].

3.1 Analysis of the Header Fields

We now focus on investigating the embedding capacity of real IPv6 traffic. We found that IPv6 is about 4% of the entire traffic, thus confirming that it offers a reduced volume of carriers compared to IPv4 [13]. We now describe the behavior of the related protocol fields as well as their suitability to create IPv6 covert channels.

Traffic Class. This field is the concatenation of DSCP and ECN. As regards the DSCP, we observed only three possible values in the traces: 0 (`0b000000`), which is the default for many network devices, has been observed in 5.5% of packets, while 2 (`0b000010`) and 3 (`0b000011`) in 17.5% and 77.5% of packets, respectively. According to [2] such values do not require any special handling from the network, thus datagram can be served in a best effort manner. Then, if DSCP is manipulated to contain secret data, any value different from those observed will represent an anomaly and could be used to detect the covert channel. Instead, values of the ECN were equal to 0 (`0b00`) in 99.99% of collected packets, hence the field is not stealth enough to contain secrets. As a consequence, the Traffic Class field can only encode 3 values out of 2^8 possibilities, thus limiting the bandwidth of the covert channel to a maximum of 2 bit/packet. Therefore, the estimate of 8 bit/packet of [9] is too optimistic for real-world use cases.

Flow Label. Firstly, we quantified how many packets have a "zero" label. Our dataset leaded to mixed results. Specifically, two traces were characterized by the 96% of packets having the Flow Label set to zero, whereas the other two traces had a zero value in the 21% and 24% of packets, respectively. We do not have enough details, to explain this behavior. According to [1], zero values are acceptable but not recommended as they can be misused. However, in the perspective of forcing the Flow Label to arbitrary values to embed data, having a non-negligible amount of packets with non-zero values can still lead to some steganographic capacity of the network covert channel. Thus, as a second step, we analyzed the statistical properties of the field. Specifically, we used the Federal Information Processing Standard 140-2 suite to assess the randomness of the Flow Label. Results indicate that the collected values are actually pseudo-random. Therefore, embedding data in Flow Label (even by respecting the 0 vs non-0 proportion) will cause perturbations of statistical properties of the field. As a possible workaround, before being injected, secret data could be pre-processed with some encryption algorithm allowing to "scramble" bits and increase the randomness. Typically, the Flow Label does not vary across the connection and the huge amount of zero-valued packets could limit its exploitability for steganographic purposes. Thus, its precise embedding capacity is difficult to predict.

Payload Length. Even if the maximum packet length for an IPv6 datagram is 56,536 bytes, the values observed in the wild limit bandwidth of the method, as packets with uncommon sizes could be easily recognized as outliers by using a traffic sniffer or a protocol analyzer. In the used dataset, the maximum packet size was equal to 1,460 bytes, which is the typical size of the Maximum Transmission Unit (MTU) supported by an IEEE 802.3/Ethernet L2 interface. Such a value was also the most common together with small packets of 32 bytes containing TCP ACKs. Other observed sizes of IPv6 datagram were equal to 1,240, 1,400, 1,420, and 1,430 bytes. Therefore, the bandwidth of a covert channel using the payload modulating approach appears as less than the theoretical one [9]. In fact, assuming an MTU of 1,500 bytes, the maximum amount of space available to transport hidden data is up to 1,416 bytes, i.e., the size of the MTU minus 24 bytes, 40 bytes, and 20 bytes, needed to contain the Ethernet, IPv6 and TCP headers, respectively.

Next Header. This field is 8 bit long, thus allowing up to 2^8 possible values. However, collected traffic shows that the 99.15% indicated the presence of TCP, whereas only 0.55% and 0.3% pointed to UDP and ICMP protocols, respectively. Therefore, as suggested by [9], introducing extensions to hide data for implementing the network covert channel may be easily spotted, as it could represent an anomaly. The resulting bandwidth could be very limited as only few packets can be artificially manipulated to act as carriers.

Hop Limit. Similar to the previous case, also this field can encode 256 values. The packets with a Hop Limit in the $51-54$ range are the most common, along with those in the $242-245$ range. This can be explained by the fact that 64 is the default and higher values could be produced by the neighbor discovery protocol, which accounts for automatic configuration and resolution of network addresses, just to mention the most important operations. Therefore, by paying attention to remain in such ranges, modulating the Hop Limit between adjacent packets allows to implement network covert channels with 1 bit/packet capacity as envisaged in [9].

Source Address. This method is highly unreliable, since a covert channel altering the Source Address may disrupt the network connection. In general, address manipulation should happen only if both the secret endpoints are co-located within the overt nodes. Yet, widespread protections against spoofing could easily detect the alteration of the address, thus blocking the covert communication attempts.

3.2 Analysis of IPv6 Conversations

In the perspective of creating network covert channels lying in an IPv6 overt traffic flow, the per-field analysis does not give insights on the packet rate, the duration of the flow or the evolution of fields in time. The major findings are as follows.

- the 99.5% of traffic targets ports 80 and 443 and carries HTTP and HTTPS conversations. As regards the duration, we found two main classes of flows. The first class groups short-living transport connections, for instance, spawned by HTTP to retrieve inline objects. In this case, the average duration is 1.33 minutes and the average packet rate is of 40 packet/minute. The second class contains longer connections, mainly carrying streaming traffic like YouTube and Vimeo. In this case, the average duration is about 18 minutes and the rate is equal to 120 packet/minute.
- the most variable fields were the Payload Length and the Hop Limit. As regards the Payload Length, its behavior is mostly influenced both by higher-layer constraints (e.g., ACK traffic vs. applications without time constraints filling the payload at maximum) and limits imposed by the MTU. Instead, the Hop Limit could vary due to the different amount of intermediate nodes processing the traffic;
- the other fields present in the IPv6 header remain almost constant within the timeframe of the conversation. Besides, the observed

Table 1: Feasibility analysis of IPv6 covert channels in different scenarios natively providing IPv6.

Test Case	Traf. Class Burst	Traf. Class Interleaved	Flow Label Burst	Flow Label Interleaved	Payl. Len. Burst	Payl. Len. Interleaved	Hop Limit Burst	Hop Limit Interleaved
Nodes in Digital Ocean from Multiple Locations (Berlin, New York, Bangalore, London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services from Multiple Locations (Singapore, North Virginia, Oregon, London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services (Singapore and Bangalore) and in Digital Ocean (New York and London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services in a Single Location								
Linux - Win								
Win - Linux								
Linux - Linux								

traffic completely lacks of additional extension headers, thus limiting the feasibility of using the steganographic technique targeting the Next Header field.

To recap, if the IPv6 covert channel is implemented in the wild via a Man-in-the-Middle basis, the size of the hidden information should be appropriate for the capacity offered by the overt flow. If the secret endpoints create synthetic IPv6 datagrams to embed data, they should not deviate too much from the rest of the traffic.

4 PERFORMANCE EVALUATION

In this section, we investigate IPv6 covert channels when deployed in the wild. Our goals are: understanding if the technological evolution of IPv6 and its deployment (see, e.g., [14]) impact on the methods originally envisaged more than a decade ago [9], and evaluating the performances of two popular IDS solutions.

4.1 Experimental Testbed and Methodology

To test the IPv6 network covert channels, we implemented in Python/Scapy, methods (1), (2), (3), and (5) presented in Section 2. As discussed in Section 3, methods (4) and (6) have been omitted since they are unsuitable for real-world scenarios. Our tool can inject the hidden data in IPv6 traffic in two ways: *burst* – secret information is hidden in consecutive IPv6 datagrams, and *interleaved* – secret information is hidden at random time intervals as to alternate modified and plain packets. We point out that, embedding data in a greedy, bursty manner could introduce less delays, but at the price of an increased detectability according to the magic triangle rule [11, 15].

Concerning the amount of data embedded in the carrier, we used values conforming to thresholds discussed in Section 3, e.g., only 2 bits of the DSCP field are used.

To evaluate the feasibility of deploying IPv6 network covert channels in the wild, we performed three rounds of tests. In the first round, we evaluated the impact of different software implementations and network deployments. To this aim, covert endpoints hosted by Digital Ocean (in Germany, USA, India and UK) and by Amazon AWS (in Singapore, USA, and UK) have been considered. In the second round, we quantified the impact of the Teredo transitional mechanism allowing v6/v4 protocols to coexist. Lastly, for the third round, we evaluated the performances of two production-quality IDS tools, i.e., Bro/Zeek[2] and Suricata[3].

For all scenarios we performed experiments using both the burst and interleaved injection techniques as well as overt traffic flows with rates similar to those observed in the wild. For each trial, we performed 40 repetitions as to have proper statistical relevance. We also tested IPv6 covert channels to exfiltrate different amounts of data, i.e., 20 bytes, 200 bytes, and 700 bytes, 1 Mbytes, and 2 Mbytes, as to model exfiltration of different types of information ranging from a credit card number to a complete document. Moreover, we also tested the channels when used with several temporal horizons, i.e., 1, 2, 5, 10, and 15 minutes.

4.2 Results

We first check whether the underlying technology impacts on the feasibility of creating a covert channel with IPv6. To quantify this, we used progress bars (i.e.,) indicating the percentage of trials for which a given covert channel successfully allowed to transmit a secret between two endpoints. As shown in Table 1, experiments only partially failed when the secret data has been

[2]https://www.zeek.org.
[3]https://suricata-ids.org.

Table 2: Feasibility analysis of IPv6 covert channels when using Teredo.

Test Case	Traf. Class Burst	Traf. Class Interleaved	Flow Label Burst	Flow Label Interleaved	Payl. Len. Burst	Payl. Len. Interleaved	Hop Limit Burst	Hop Limit Interleaved
Nodes in Digital Ocean from Multiple Locations (Berlin, New York, Bangalore, London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services from Multiple Locations (Singapore, North Virginia, Oregon, London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services (Singapore and Bangalore) and in Digital Ocean (New York and London)								
Linux - Win								
Win - Linux								
Linux - Linux								
Nodes in Amazon Web Services in a Single Location								
Linux - Win								
Win - Linux								
Linux - Linux								

hidden in the Traffic Class field considering some OS/network configurations. Unfortunately, we did not have access to hop-by-hop traces, but by comparing the sent and received traffic, we noticed that some intermediate nodes overwrote DSCP value to `0b000000`, hence disrupting the channel. Instead, for the other methods, we were able to successfully implement them in all the considered configurations.

Table 2 reports the obtained results when the IPv6 traffic is routed via a Teredo tunnel. As shown, the worst performances were achieved when data is injected in burst. In this case, the use of UDP(v4) as the protocol encapsulating IPv6 packets for tunneling could play a major role when in the presence of packet loss, e.g., when other traffic sources burden the Teredo relay. In this case, the IPv6 covert channel appears as a very fragile. On the contrary, the use of an interleaved injection policy de-correlate losses, thus increasing the chance of correctly receiving the secret messages. The heterogeneity of endpoints and deployments also play a role. In fact, best results are achieved when setting the covert channel between nodes deployed within the same AWS location.

Lastly, we quantified the effectiveness of Bro and Suricata IDS (denoted in the following as "bro" and "sur", respectively) in detecting IPv6 covert channels. To this aim, we used ● and ○ to denote if the IDS successfully detected or ignored the threat, respectively, and with ◐ if the flow has been marked suspicious. Concerning covert channels deployed on scenarios natively supporting IPv6, both IDS tools tend to not detect them. The only exception is Suricata, which flagged the flow as suspicious for the channel using the DSCP field (i.e., the Traffic Class method). For the sake of brevity, we omitted such a table. Instead, when using the Teredo tunnel, more diversified results have been collected and presented in Table 3. As shown, the methods using the Payload Length and the Hop Limit with the interleaved strategy are never recognized as threats, whereas the other methods are detected or the embedding flow is marked as suspicious. Even if such tools are not suitable to detect IPv6 covert channels "out of the box", they can be considered as starting points for developing more effective countermeasures. We also point out that Suricata outperformed Bro.

4.3 Countermeasures

According to our investigation, threats using IPv6 covert channels, can be mitigated by updating/improving the rules and signatures used by IDS to spot anomalies. To this aim, the IDS should check how the various fields of the IPv6 header vary within the single flow/connection or compare them against a reference statistical template. For the case of the presented covert channels, some possible ideas are: Traffic Class and Flow Label should not change within the same connection, thus they can be used to mark a flow for further investigations; the Payload Length should be checked against MTU or path MTU discovery as to reveal possible discrepancies; the Next Header usually points to TCP or UDP datagrams, thus some statistical check can be done to reveal anomalies; the Hop Limit could naturally vary due to the different number of intermediate nodes traversed by the flow, but the variability is usually limited to 1/2 hops, so a sort of guard-threshold could reveal the channel.

5 CONCLUSION

In this paper we have investigated the behaviors of IPv6 network covert channels when deployed in the wild. Results indicate that, as today, the hiding capacity of real network traffic is reduced compared to the theoretical limits proposed by the related literature. Moreover, our investigation also showcased that some IDS solutions can not be considered as effective tools to detect such threats, at

Table 3: Analysis of Bro and Suricata detecting IPv6 covert channels when using Teredo.

Test Case	Traf. Class Burst		Traf. Class Interleaved		Flow Label Burst		Flow Label Interleaved		Payl. Len. Burst		Payl. Len. Interleaved		Hop Limit Burst		Hop Limit Interleaved	
	Nodes in Digital Ocean from Multiple Locations (Berlin, New York, Bangalore, London)															
	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur
Linux - Win	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
Win - Linux	●	●	○	◐	●	●	○	○	○	○	○	○	●	●	○	○
Linux - Linux	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
	Nodes in Amazon Web Services from Multiple Locations (Singapore, North Virginia, Oregon, London)															
	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur
Linux - Win	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
Win - Linux	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
Linux - Linux	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
	Nodes in Amazon Web Services (Singapore and Bangalore) and in Digital Ocean (New York and London)															
	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur
Linux - Win	●	●	○	◐	○	○	○	○	○	○	○	○	●	●	○	○
Win - Linux	●	●	○	◐	●	●	○	○	○	○	○	○	●	●	○	○
Linux - Linux	○	◐	○	◐	●	●	○	○	○	○	○	○	●	●	○	○
	Nodes in Amazon Web Services in a Single Location															
	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur	bro	sur
Linux - Win	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
Win - Linux	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○
Linux - Linux	○	◐	○	◐	○	○	○	○	○	○	○	○	○	○	○	○

least if the hidden data is injected within IPv6 traffic with a network-wide support (i.e., no transitional/tunneling mechanisms are used).

Future works aim at refining our investigation by removing limits imposed by the unavailability of traces with a hop-by-hop granularity. Besides, we are working towards the definition of suitable metrics for detecting IPv6 covert channels, for instance, by means of machine learning approaches or similar statistical techniques.

ACKNOWLEDGMENTS

This work has been partially supported by EU Project SIMARGL, Grant Agreement No 833042 and by the Polish National Agency for Academic Exchange, Grant Agreement No PPN/BEK/2018/00153.

REFERENCES

[1] S. Amante, B. Carpenter, S. Jiang, and J. Rajahalme. 2011. *IPv6 Flow Label Specification.* RFC 6437. RFC Editor.
[2] Steven Blake, David L. Black, Mark A. Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. 1998. *An Architecture for Differentiated Services.* RFC 2475. RFC Editor.
[3] Bernhards Blumbergs, Mauno Pihelgas, Markus Kont, Olaf Maennel, and Risto Vaarandi. 2016. Creating and detecting IPv6 transition mechanism-based information exfiltration covert channels. In *Nordic Conference on Secure IT Systems.* Springer, 85–100.
[4] Anna L Buczak and Erhan Guven. 2015. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* 18, 2 (2015), 1153–1176.
[5] Stephen E. Deering and Robert M. Hinden. 1995. *Internet Protocol, Version 6 (IPv6) Specification.* RFC 1883. RFC Editor.
[6] R. Hinden and S. Deering. 1995. *IP Version 6 Addressing Architecture.* RFC 1884. RFC Editor.
[7] Alfonso Iacovazzi and Yuval Elovici. 2016. Network flow watermarking: A survey. *IEEE Communications Surveys & Tutorials* 19, 1 (2016), 512–530.
[8] Grzegorz Lewandowski, Norka B Lucena, and Steve J Chapin. 2006. Analyzing Network-aware Active Wardens in IPv6. In *International Workshop on Information Hiding.* Springer, 58–77.
[9] Norka B Lucena, Grzegorz Lewandowski, and Steve J Chapin. 2005. Covert Channels in IPv6. In *International Workshop on Privacy Enhancing Technologies.* Springer, 147–166.
[10] Norka B Lucena, Grzegorz Lewandowski, and Steve J Chapin. 2008. Eliminating Covert Channels in IPv6 with Network-Aware Active Wardens. (2008).
[11] Wojciech Mazurczyk and Luca Caviglione. 2014. Steganography in modern smartphones and mitigation techniques. *IEEE Communications Surveys & Tutorials* 17, 1 (2014), 334–357.
[12] Wojciech Mazurczyk and Luca Caviglione. 2015. Information Hiding as a Challenge for Malware Detection. *IEEE Security & Privacy* 2 (2015), 89–93.
[13] David N Muchene, Klevis Luli, and Craig A Shue. 2013. Reporting insider threats via covert channels. In *2013 IEEE Security and Privacy Workshops.* IEEE, 68–71.
[14] Mehdi Nikkhah, Roch Guérin, and Mehdi Nikkhah. 2016. Migrating the internet to IPv6: An exploration of the when and why. *IEEE/ACM Transactions on Networking)* 24, 4 (2016), 2291–2304.
[15] Sabine Schmidt, Wojciech Mazurczyk, Radoslaw Kulesza, Jörg Keller, and Luca Caviglione. 2018. Exploiting IP telephony with silence suppression for hidden data transfers. *Computers & Security* 79 (2018), 17–32.
[16] W. Townsley and O. Troan. 2010. *IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) – Protocol Specification.* RFC 5969. RFC Editor.
[17] Johanna Ullrich, Katharina Krombholz, Heidelinde Hobel, Adrian Dabrowski, and Edgar Weippl. 2014. IPv6 security: attacks and countermeasures in a nutshell. In *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14).*
[18] Daniel G Waddington and Fangzhe Chang. 2002. Realizing the transition to IPv6. *IEEE Communications Magazine* 40, 6 (2002), 138–148.
[19] S. Wendzel, S. Zander, B. Fechner, and C. Herdin. 2015. Pattern-based Survey and Categorization of Network Covert Channel Techniques. *Comp. Surv.* 47, 3 (2015).
[20] S. Zander, G. Armitage, and P. Branch. 2007. A survey of covert channels and countermeasures in computer network protocols. *Comm. Surveys and Tutorials* 9, 3 (2007), 44–57.