Header Steganography

Abusing IPv6 Headers & Prefix Delegation for Data Exfiltration

CackalackyCon 2019

Before We Dive In...

Standard disclaimers apply, especially the one about using this information for illegal shit.

CackalackyCon is open to attendees of all skill levels, so I tried to include information for the beginners just breaking into security/networking all the way up to veterans in the field.

Questions are welcome,
but *please* hold them until the end.

SYN

- New Orleans area native (504/985)
- Worn many hats: Datacenter Engineer, Internet Engineer, CMTS Engineer, Systems & Network Administrator
- Day Job: Sr Cyber Security Analyst
- Work with several animal rescue organizations, certified pilot, and nature photographer
- ISO 8601 Compliant
- CISSP (Don't Judge Me…)

@__wavelength__

darkwavelength@protonmail.ch

Data Exfil…. What?

Data Exfiltration

The unauthorized copying or transfer of data from an environment. Also known as data theft, data exportation or data extrusion.

Motivations

Espionage (Industrial or State-sponsored)

Blackmail / Extortion

Financial Crimes - identity theft for financial or medical fraud

Hacktivism - Wikileaks, Panama Papers, Ashley Madison, etc.

Intelligence gathering - law enforcement, three-letter agencies

Lulz and other, unclear, motives - “The Fappening”, etc.

Penetration testing, CTF, etc.

Analog Methods

Printing

Photocopying

Photography

Transcription

Or just steal it…

“Sneakernet” Methods

External Storage
(portable drives / removable media)

Network-based Methods

The Obvious Protocols

FTP File Transfer Protocol - tcp/20 & tcp/21

SFTP Secure File Transfer Protocol - tcp/22

HTTP HyperText Transfer Protocol - tcp/80

HTTPS Secure HyperText Transfer Protocol - tcp/443

SMTP Simple Mail Transfer Protocol - tcp/25

POP3 Post Office Protocol version 3 - tcp/110

Network-based Methods

More Exotic Exfiltration Methods

DNS

Domain Name Service - udp/53 & tcp/53

Exfiltrated data is placed in the query portion of a DNS request to a domain under the adversary's control, where it is logged and reassembled

ICMP

Internet Control Message Protocol

Exfiltrated data is placed in the data field of an ICMP Echo Request.

TCP

Transmission Control Protocol headers

Exfiltrated data is placed in the port or sequence number field (4 bytes) of a TCP packet.

A Common Thread

The Risks

Firewalls

Outbound ports or protocols are blocked

Session logs provide a trail of evidence

Failed attempts trigger drops logs that generate SIEM alerts

IDS/IPS

Signatures are capable of detecting many types of exfiltration

Examples: Snort SIDs 1-45967, 1-39341, 3-30881, 1-31212, etc.

SIEM

Correlation rules can flag data exfiltration activities

Data Loss Prevention (DLP)

DLP systems are capable of detecting data in transit

So why use them?

1. They are easy to use…

Utilities are readily available
wget, curl, ftp, scp, sftp, dig, ping, etc.

Libraries are readily available
C, C++, Go, Java, Python, Perl, etc.

Mature ecosystem of servers

Flow-Control

<u>Most</u> networks do <u>not</u> block all of the commonly used protocols
If a network is usable, it is probably vulnerable to exfiltration…

2. Space

In IPv4, further down the OSI model, less space for data…

IPv4 Header

Image Source: nmap.org

TCP Header

Image Source: nmap.org

TCP Header

Image Source: nmap.org

TCP Header

Image Source: nmap.org

TCP Header

TCP Flags

C	E	U	A	P	R	S	F

Congestion Window
C 0x80 Reduced (CWR)
E 0x40 ECN Echo (ECE)
U 0x20 Urgent
A 0x10 Ack
P 0x08 Push
R 0x04 Reset
S 0x02 Syn
F 0x01 Fin

Congestion Notification

ECN (Explicit Congestion Notification). See RFC 3168 for full details, valid states below.

Packet State	DSB	ECN bits
Syn	0 0	1 1
Syn-Ack	0 0	0 1
Ack	0 1	0 0
No Congestion	0 1	0 0
No Congestion	1 0	0 0
Congestion	1 1	0 0
Receiver Response	1 1	0 1
Sender Response	1 1	1 1

TCP Options

0 End of Options List
1 No Operation (NOP, Pad)
2 Maximum segment size
3 Window Scale
4 Selective ACK ok
8 Timestamp

Checksum

Checksum of entire TCP segment and pseudo header (parts of IP header)

Offset

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

RFC 793

Please refer to RFC 793 for the complete Transmission Control Protocol (TCP) Specification.

Image Source: nmap.org

UDP Header

Image Source: nmap.org

UDP Header

Image Source: nmap.org

3. Network Address Translation

Well, technically Network Address/Port Translation (NAPT)

NAT changes parameters at Layer 3 (IP) and Layer 4 (TCP & UDP)

Source IP

Source Port

TCP sequence numbers
(Yes, some firewalls do this)

Destination IP

Destination Port
(Port Forwarding)

End-to-end integrity of data encoded in these layers is not guaranteed

Some Historical Context

IPv4 was formalized in 1981,
but, by the early ‘90s, it was clear there was a problem.

Address space exhaustion!

32 bits of address space is 4,294,967,296 addresses, but…

Addresses reserved for Multicast:	268,435,424
Addresses reserved for “Future Use”:	268,435,424
Addresses tied up in “Legacy” /8 assignments:	~300,000,000
Addresses reserved for the “Loopback”:	16,777,216
Addresses reserved for “Private Addressing”:	~26,200,000

Leaving ~3.4 Billion addresses…

Seems like a lot…

Earth's Population in 1995 - 5.7 Billion

Earth's Population in 2006 - 6.7 Billion

Number of connected devices in 2009 - 4.8 Billion (est.)

Cell Phones in the world in 2015 - 4.5 Billion

Number of connected devices in 2020 - 25-75 Billion (est.)

Network Address/Port Translation

Quick Intro / Refresher

Image Source: practicalnetworking.net

This forces data into the upper layers of the OSI model.

"Immune" to NAPT*

**Application Layer Gateways may still be a problem… Let's not go there. ;)*

So, what if we lived in a world without NAT?

Introducing ~~IPv5~~ IPv6

FYI, “IPv5” DOES exist…
https://en.wikipedia.org/wiki/Internet_Stream_Protocol

History of IPv6

1990 - Initiation of the effort to define the replacement for IPv4

1992 - The first proposals for IPv4 replacement start appearing

1995 - RFC 1883 defines the first draft of an IPv6 “standard”

1998 - RFC 2460 supersedes RFC 1883

2017 - RFC 8200 supersedes RFC 2460

Some of the changes in IPv6

(Not an exhaustive list…)

WARNING: Just a few walls of text ahead…

Changes from IPv4

1. Address space quadrupled to 128 bits from 32 bits

IPv4: 4,294,967,296 addresses

IPv6: 2^{128} or 3.4028237×10^{38} addresses

340,282,366,920,938,463,463,374,607,431,768,211,456

Just for comparison…

Galaxies in the observable universe: 1.0×10^{10}

Stars in the average galaxy: 1.0×10^{11}

Stars in the universe: 1.0×10^{21} or 1 billion trillion

1,000,000,000,000,000,000,000 stars

Number of grains of sand on earth (est.): 7.5×10^{18}

7,500,000,000,000,000,000 grains

Changes from IPv4

2. Address format changed

IPv4: decimal digits in a “dotted quad” format

Examples: 10.100.1.1, 4.2.2.2, 204.251.13.1, etc

IPv6: eight, colon-separated hexadecimal groups

Examples:

fe80::289:1f1:a010:8b38

2a03:2880:f111:83:face:b00c::25de

2607:f8b0:4002:808::200e

2600:1402:f000:79e::1aca

Changes from IPv4

3. Subnetting changes

IPv6: like IPv4, subnets work the same…

But the /64 bit subnet is special. This is the smallest subnet recommended when hosts (PCs, phones, etc.) are attached to it.

Why?

Key IPv6 features expect host-attached networks are configured as /64:

Extended Unique Identifier (EUI-64) - allows host to automatically configure valid IPv6 address using NIC MAC address (v6 equiv. of RFC 3927)

Neighbor Discovery Protocol (NDP)

Router Advertisement (RA) - router advertises local subnet to hosts allowing them to use EUI-64 to automatically configure an address

Neighbor Discovery (ND) - neighboring systems advertise their link-local addresses to each other (v6 “equivalent” of ARP and RARP)

Changes from IPv4

4. Prefix Delegation (PD)

IPv4: no equivalent feature; not enough addresses

Customers assigned a single IPv4 address (NAPT)

IPv6: scaleable way to assign subnets to customers

1. Customer assigned /64 subnet for their network
2. Customer assigned < /64 subnet for their network

Cable Provider X - Assigns /60 to residential customers

Equivalent to *16 x /64 networks*

Cable Provider Y - Assigns /56 to residential customers

Equivalent to *256 x /64 networks*

Prefix Delegation Illustrated

Overview

Prefix Delegation Illustrated

Typical Site

I would be remiss…

Bear with me for just one moment…

Nothing about IPv6 prevents existing methods from being used…

But IPv6 gives threat actors new options for data exfiltration...

Abusing the IPv6 Header

Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 6 structure only.

Traffic Class

8 bit traffic class field.

Flow Label

20 bit flow label.

Payload Length

16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Any extension headers are considered part of the payload.

Source Address

128-bit address of the originator of the packet.

Next Header

8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.

Destination Address

128-bit address of the intended recipient of the packet (possibly not the ultimate recipient, if a Routing header is present).

Hop Limit

8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.

RFC 2460

Please refer to RFC 2460 for the complete Internet Protocol version 6 (IPv6) Specification.

Copyright 2008 - Matt Baxter - mjb@fatpipe.org - www.fatpipe.org/~mjb/Drawings/

Abusing the IPv6 Header

Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 6 structure only.

Traffic Class

8 bit traffic class field.

Flow Label

20 bit flow label.

Payload Length

16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Any extension headers are considered part of the payload.

Source Address

128-bit address of the originator of the packet.

Next Header

8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.

Destination Address

128-bit address of the intended recipient of the packet (possibly not the ultimate recipient, if a Routing header is present).

Hop Limit

8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.

RFC 2460

Please refer to RFC 2460 for the complete Internet Protocol version 6 (IPv6) Specification.

Copyright 2008 - Matt Baxter - mjb@fatpipe.org - www.fatpipe.org/~mjb/Drawings/

Abusing the IPv6 Header

Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 6 structure only.

Traffic Class

8 bit traffic class field.

Flow Label

20 bit flow label.

Payload Length

16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Any extension headers are considered part of the payload.

Source Address

128-bit address of the originator of the packet.

Next Header

8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.

Destination Address

128-bit address of the intended recipient of the packet (possibly not the ultimate recipient, if a Routing header is present).

Hop Limit

8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.

RFC 2460

Please refer to RFC 2460 for the complete Internet Protocol version 6 (IPv6) Specification.

Copyright 2008 - Matt Baxter - mjb@fatpipe.org - www.fatpipe.org/~mjb/Drawings/

Abusing the IPv6 Header

Hexadecimal

0 1 2 3 4 5 6 7 8 9 A B C D E F

Another name for hexadecimal is Base-16…

And data can be encoded in Base-16…

HELLO!

48656C6C6F21

And if Hex encoded data looks just like valid IPv6 address bits…

Steganography!

Steganography?

Noun

The art or practice of concealing a message, image, or file within another message, image, or file

`Meet@10P` **Convert to Hex** `4D6565744031305 0`

2001:DB8::4D65:6574:4031:3050

2001:DB8::0D1C:4FE3:230A:EFA3

2001:DB8::5245:4345:4956:4544

2001:DB8::56DF:DEAD:BEEF:0AB3

Steganography?

Noun

The art or practice of concealing a message, image, or file within another message, image, or file

`Meet@10P` **Convert to Hex** `4D6565744031305 0`

2001:DB8::4D65:6574:4031:3050

2001:DB8::0D1C:4FE3:230A:EFA3

2001:DB8::5245:4345:4956:4544

2001:DB8::56DF:DEAD:BEEF:0AB3

Steganography?

Noun

The art or practice of concealing a message, image, or file within another message, image, or file

`Meet@10P` **Convert to Hex** `4D6565744031305 0`

2001:DB8::4D65:6574:4031:3050

2001:DB8::0D1C:4FE3:230A:EFA3

RECEIVED

2001:DB8::56DF:DEAD:BEEF:0AB3

Steganography

One key consideration with steganography is the amount of data that can be hidden within a given set of data without being detected.

The IPv6 minimum recommended subnet size dictates the amount of data that can be concealed in the Layer 3 header fields.

64 bits = 8 bytes / packet

IPv6 Subnetting

IPv6 Subnetting

So, 8 bytes of data fits into the host portion of a v6 address…

But where do you send it?

If only you had a spare /64 IPv6 subnet…

Or multiple subnets…

Abusing Prefix Delegation

The Setup

Abusing Prefix Delegation

The Setup

Abusing Prefix Delegation

The Setup

```
ipv6 route 2600:1:300:1f::/64 2600:1:300:10:a1bc:3dc:190b:fe1a
```

Abusing Prefix Delegation

The Setup

```
ipv6 route 2600:1:300:1f::/64 2600:1:300:10:a1bc:3dc:190b:fe1a
```

Abusing Prefix Delegation

The Setup

```
me@test-vm:~$ ip -6 route
me@test-vm:~$ sudo ip -6 route add blackhole 2600:1:300:1f::/64
me@test-vm:~$ ip -6 route
blackhole 2600:1:300:1f::/64 dev lo  metric 1024  error -22 pref medium
```

Abusing Prefix Delegation

The Setup

```
ipv6 route 2600:1:300:1f::/64 2600:1:300:10:a1bc:3dc:190b:fe1a
```

Abusing Prefix Delegation

The Setup

```
ipv6 route 2600:1:300:1f::/64 2600:1:300:10:a1bc:3dc:190b:fe1a
```

“System Requirements”

Source Configuration

1. Native IPv6 - tunneling acceptable, if permitted, and using it ***will not*** trigger alerts; other transition technologies (Teredo, etc.) likely to be blocked or trigger alerts.
2. Compression tools - zip, rar, etc.
3. Format conversion tools - binary to hexadecimal, specifically
4. Scapy OR other packet/network request tools (wget, dig, etc.), with limitations

Destination Configuration

1. Native IPv6 - tunneling acceptable; must have an unused /64 subnet configured either through prefix delegation or static configuration
2. Router capable of statically routing PD assigned subnets to an arbitrary destination
3. Compression tools - zip, rar, etc.
4. Format conversion tools - hexadecimal to binary, specifically
5. Packet capture utilities - tcpdump, Wireshark, etc.

Let’s Send a Message

“Welcome to CackalackyCon”

Let’s Send a Message

10,000ft Version

“Welcome to CackalackyCon”

↓

1. Convert String to Base-16 (Hexadecimal)

57656C636F6D6520746F204361636B616C61636B79636F6E

Let's Send a Message

10,000ft Version

"Welcome to CackalackyCon"

↓

1. Convert String to Base-16 (Hexadecimal)

57656C636F6D6520746F204361636B616C61636B79636F6E

↓

2. Divide result into 64bit segments

57656C636F6D6520 746F204361636B61 6C61636B79636F6E

Let's Send a Message

10,000ft Version

"Welcome to CackalackyCon"

↓

1. Convert String to Base-16 (Hexadecimal)

57656C636F6D6520746F204361636B616C61636B79636F6E

↓

2. Divide result into 64bit segments

57656C636F6D6520 746F204361636B61 6C61636B79636F6E

↓

3. Associate a numerical order (in hex) to each segment

0000000001 57656C636F6D6520

0000000002 746F204361636B61

0000000003 6C61636B79636F6E

Let's Send a Message

10,000ft Version

4. Put each segment into the Host ID part of destination IPv6 address and transmit with numerical identifier…

0000000001 **2001:db8::5765:6C63:6F6D:6520**

Transmitter Processing

Step-by-Step

1. Select the target data.

2. Data compression to reduce transmission time. (Optional)

3. Format conversion to Base-16 (Hex)

4. Data slicing - converting a continuous string of hexadecimal characters to 64 bit segments.

5. Data sequencing - associate an ordered hexadecimal identifier with each 64 bit string to be transmitted with the segment for loss detection and reassembly. (Optional)

Transmitter Processing

Step-by-Step

6. **Packet Construction: Using SCAPY, insert the associated numerical identifier for the first segment into the "Flow Label" field. Use the target PD subnet as the subnet portion of the destination IPv6 address and the first 8 byte segment as the host identifier. Source and destination port are at your discretion.**

7. **Transmit packet.**

8. **Repeat steps 6 and 7 for each segment.**

Padding

Data will NOT always end on a 64 bit boundary

HELLO → Convert to Hex → **48656C6C6F**

How to handle this case?

Prepend padding bits?

2001:db8::XXXX:XX48:656C:6C6F

Append padding bits?

2001:db8::4865:6C6C:6FXX:XXXX

Padding

How is not that important, so long as the padding method is deterministic and reversible.

Receiver Processing

Step-by-Step

1. Start a packet capture filtered for the destination PD subnet.
2. Once the final segment has been received, extract the Flow Label field and host identifier values from each packet and order the segments.
3. Concatenate the 8 byte segments into the original order.
4. Convert from hexadecimal back to the original format.
5. Decompress, if needed.

Receiver Processing

Step-by-Step

6. Tent your fingers and say…

Receiver Processing

Step-by-Step

EEEEEXCELLENT!

Weaknesses and Challenges

Traceable

All the addresses are visible and less likely to change. Pointing fingers at another user is unlikely to fly.

Hurry up & wait

Transmitting only 8 bytes at a time is not particularly fast. Good for an APT-style low and slow attack. Blasting away is an option, but more likely to attract attention.

Disinformation

A technically adept victim could discover and poison the data being exfiltrated without detection. They could embed their own malicious payloads or just waste time with garbage data allowing them time for IR.

Weaknesses and Challenges

Protocol Limits

The IPv6 flow label is field is only 20 bits. This allows for only ~1MM segments, about 8MB, before the counter rolls.

Packet Loss

Without some sort of a flow control, lost packets are detectable but not recoverable. This results in corrupted data.

Potential Detection Methods

Netflow Anomaly Detection

1:1 Netflow with Anomaly Detection is able to detect single host to many host conversations that exceed configurable thresholds. Low and slow transmission could thwart this method of detection assuming sufficiently long periods of time between transmissions.

Full Packet Capture

A victim armed with full packet capture would be able to run anomaly detection while also being able to reconstruct the data transmitted by the adversary. Encryption prior to transmission would thwart reconstruction of exfiltrated data.

So could this be detected?

Unfortunately for Blue Teams, detection is unlikely today.

IDS/IPS?

No, not currently. Signatures currently examine source/destination IP for rough filtering or blacklist matches, but do not examine those fields for embedded payloads. Doing so would be computationally intensive and potentially prohibitive to implement.

DLP?

No, these systems also do not inspect source/destination IP for payloads and doing so would have the same issues as implementing this functionality in an IDS/IPS.

So could this be detected?

Unfortunately for Blue Teams, detection is unlikely today.

Full Packet Capture and 1:1 Netflow Anomaly Detection is expensive and out of reach for many potential victims.

The rate of false positives with any detection method would be high because even legitimate source/destination addresses will contain patterns mimicking embedded data.

Single source to many destination pattern of exfiltration traffic also mimics the normal one to many pattern of legitimate traffic.

ACK

Related Works

Giving credit where credit is due…

Miller, Barrett. "*Steganography in IPv6*". University of Arkansas. 2008

Qasim, Mustafa Alaa and Dipak Pawar. "*Encryption & Steganography in IPv6 Source Address*". International Journal of Computer Engineering & Technology. Vol 4, Issue 2, March – April (2013), pp. 315-324

Both papers encode data in the Source Address field of the IPv6 header field directed to a single destination address.

The drawback to this method is it generates more noise at the source by increasing the number of neighbor advertisements. Rapid turnover of source addresses, inconsistent with normal churn, can be detected and may trigger incident response.

FIN

Questions, Comments, Flames?

@__wavelength__

darkwavelength@protonmail.ch