Useful Header Fields

Think About

As you examine this this Resource, think about situations in which you would have to use pOf and situations in which you wouldn't be able to use pOf.

IP Headers

Time-to-live (TTL)

The Time-to-live (TTL) indicates how long a packet can stay on the wire. It is decremented by one for each hop (router) that the packet passes through. An initial TTL value can say a lot about an operating system (OS). The TTL is the maximum number of routers a packet can pass before it is being dropped. It is initialized by the sender and then decremented by every router, or any layer 3 device, handling the packet. When the value reaches 0, the packet is dropped and an ICMP message is returned to the sender. The following are examples of typical OS packet specifications:

OS	TTL	TCP Window Size
Linux kernel 2.x	64	5840 kilobytes
Android & Chrome	64	5720 kilobytes
Windows XP	128	65535 kilobytes
Windows 7 and Server 2008	128	8192 kilobytes
Cisco routers	255	4128 kilobytes

Total Packet Length

The total packet length field tells how long the whole packet is. This includes the embedded header, IP header, and the payload. The total length of a packet is very dependent on the OS (especially a SYN packet). Each operating system sets its own TCP options along with NOP's, these options affect the total length and therefore make each OS unique in total packet length. An operating system can be identified in some cases just by the packet length alone on the SYN packet. Also, when looking at TCP packets keep in mind that almost all operating systems use at least one TCP option on the SYN packet. Normally, this is the MSS option and that option is 4 bytes in length (making the minimum total length = 44 bytes). Therefore, seeing a 40 byte packet coming in or out of the network would be suspicious because it is a crafted packet.

IP ID

The IP ID is an identifying value assigned by the sender to aid in assembling the fragments. It is used mainly in fragmentation but can play a big role in passive OS identification.

DF (Don't Fragment)

The DF is the value set if the packet is not to be broken into smaller fragments. The DF field can be used to help identify an operating system but the problem is most operating systems set the DF flag by default; therefore, this field becomes useless in identifying many operating systems.

TCP Headers

Source Port

The source port can give some hints to help with an OS determination, but not enough to accurately identify it alone. For example, RedHat Linux will use ports 1024 through 4999 by default. If NAT is being used, you won't be able to rely on this at all, as NAT uses high source ports.

Window

This field is used by the receiver to indicate to the sender the amount of data that it is able to accept. Once a connection has been made, this number can change depending on how much data needs to be passed. The initial window value can be different from OS to OS.

TCP Header Options

When it comes to passive OS fingerprinting this is the most important field in the TCP header. This field by itself can almost identify most operating systems. Listed below are helpful TCP header options:

- Maximum Segment Size (MSS) — The largest amount of TCP data, specified in bytes, which can be handled in a single unfragmented session.
- Timestamp — Measures delays
- Window Scale (wscale) — Indicates that the TCP is prepared to both send and receive window scaling. Communicates a scale factor to be applied to its' receive window.
- Selective Acknowledgement (SackOK) — Informs the sender of all received data so that the sender can re-transmit only the data that has not been received.

- NOP — 1 byte in size. Used to pad the TCP options field to number that can be divided by 4.

p0f Command line Options

-f fname — Reads fingerprint database (p0f.fp) from the specified location. The default location is ./p0f.fp. To install p0f, it may help to change FP_FILE in config.h to /etc/p0f.fp.

-i iface — Asks p0f to listen on a specific network interface. On un*x, reference the interface by name (e.g., eth0). On Windows, use adapter index instead (0, 1, 2...).

Multiple -i parameters are not supported; must run separate instances of p0f for that. On Linux, specify 'any' to access a pseudo-device that combines the traffic on all other interfaces; the only limitation is that libpcap will not recognize VLAN-tagged frames in this mode, which may be an issue in some of the more exotic setups. If an interface is not specified, libpcap will probably pick the first working interface in the system.

-L — Lists all available network interfaces, then quits. Particularly useful on Windows, where the system-generated interface names are impossible to memorize.

-r fname — Instead of listening for live traffic, reads pcap captures from the specified file. The data can be collected with tcpdump or any other compatible tool. Make sure that snapshot length (-s option in tcpdump) is large enough not to truncate packets; the default may be too small. As with -i, only one -r option can be specified at any given time.

-o fname — Appends grep-friendly log data to the specified file. The log contains all observations made by p0f about every matching connection, and may grow large; plan accordingly. Only one instance of p0f should be writing to a particular file at any given time; where supported, advisory locking is used to avoid problems.

-s fname — Listens for API queries on the specified filesystem socket. This allows other programs to ask p0f about its current thoughts about a particular host. Only one instance of p0f can be listening on a particular socket at any given time. The mode is also incompatible with -r.

-d — Runs p0f in daemon mode: the program will fork into background and continue writing to the specified log file or API socket. It will continue running until killed, until the listening interface is shut down, or until some other fatal error is encountered. This mode requires either -o or -s to be specified. To continue capturing p0f debug output and error messages (but not signatures), redirect stderr to another non-TTY destination, e.g.: ./p0f -o /var/log/p0f.log -d 2>>/var/log/p0f.error
Note that if -d is specified and stderr points to a TTY, error messages will be lost.

-u user	Causes p0f to drop privileges, switching to the specified user and chroot()ing itself to said user's home directory. This mode is *highly* advisable (but not required) on un*x systems, especially in daemon mode.
-p	Puts the interface specified with -i in promiscuous mode. If supported by the firmware, the card will also process frames not addressed to it.
-S num	Sets the maximum number of simultaneous API connections. The default is 20; the upper cap is 100.
-m c,h	Sets the maximum number of connections (c) and hosts (h) to be tracked at the same time (default: c = 1,000, h = 10,000). Once the limit is reached, the oldest 10% entries gets pruned to make room for new data. This setting effectively controls the memory footprint of p0f. The cost of tracking a single host is under 400 bytes; active connections have a worst-case footprint of about 18 kB. High limits have some CPU impact, too, by the virtue of complicating data lookups in the cache.
-t c,h	Sets the timeout for collecting signatures for any connection (c); and for purging idle hosts from in-memory cache (h). The first parameter is given in seconds, and defaults to 30 s; the second one is in minutes, and defaults to 120 min. The first value must be just high enough to reliably capture SYN, SYN+ACK, and the initial few kB of traffic. Low-performance sites may want to increase it slightly. The second value governs for how long API queries about a previously seen host can be made; and what's the maximum interval between signatures to still trigger NAT detection and so on. Raising it is usually not advisable; lowering it to 5-10 minutes may make sense for high-traffic servers, where it is possible to see several unrelated visitors subsequently obtaining the same dynamic IP from their ISP.

Signature Groups

type	Some signatures in p0f.fp offer broad, last-resort matching for less researched corner cases. The goal is to give an answer slightly better than "unknown", but less precise than what the user may be expecting. Normal, reasonably specific signatures that can't be radically improved should have their type specified as 's'; while generic, last-resort ones should be tagged with 'g'. Note that generic signatures are considered only if no specific matches are found in the database.
class	The tool needs to distinguish between OS-identifying signatures (only one of which should be matched for any given host) and signatures that just identify user applications (many of which may be seen concurrently). To assist with this, OS-specific signatures should specify the OS architecture family here (e.g., 'win', 'unix', 'cisco'); while application-related sigs (NMap, MSIE, Apache) should use a special value of '!'. Most TCP signatures are OS-specific, and should have

the OS family defined. Other signatures, such as HTTP, should use '!' unless the fingerprinted component is deeply intertwined with the platform (e.g., Windows Update).
NOTE: To avoid variations (e.g. 'win' and 'windows' or 'unix' and 'linux'), all classes need to be pre-registered using a 'classes' directive, seen near the beginning of p0f.fp.

name — A human-readable short name for what the fingerprint actually helps identify - say, 'Linux', 'Sendmail', or 'NMap'. The tool doesn't care about the exact value, but requires consistency – so don't switch between 'Internet Explorer' and 'MSIE', or 'MacOS' and 'Mac OS'.

flavor — Anything to further qualify the observation. Can be the version of the identified software, or a description of what the application seems to be doing (e.g. 'SYN scan' for NMap).
NOTE: Don't be too specific, for example, if there is a signature for Apache 2.2.16, and there is no reason to suspect other recent versions behave in a radically different way, just say '2.x'.

TCP Signatures

ver	Signature for IPv4 ('4'), IPv6 ('6'), or both ('*'). NEW SIGNATURES: p0f documents the protocol observed on the wire, but the user should replace it with '*' unless some actual differences between IPv4 and IPv6 traffic have been observed, or unless the software supports only one of these versions to begin with.
ittl	Initial TTL used by the OS. Almost all operating systems use 64, 128, or 255; prevous versions of Windows sometimes used 32, and several obscure systems sometimes resort to odd value such as 60. NEW SIGNATURES: p0f will usually suggest something, using the format of 'observed_ttl+distance' (e.g. 54+10). Consider using traceroute to check that the distance is accurate, then sum up the values. If initial TTL can't be guessed, p0f will output 'nnn+?', and traceroute will need to be used to estimate the '?'. A handful of userspace tools will generate random TTLs. In these cases, determine maximum initial TTL and then add a - suffix to the value to avoid confusion.
olen	Length of IPv4 options or IPv6 extension headers. Usually zero for normal IPv4 traffic; always zero for IPv6 due to the limitations of libpcap. NEW SIGNATURES: Copy p0f output literally.
mss	Maximum segment size, if specified in TCP options. Special value of '*' can be used to denote that MSS varies depending on the parameters of sender's network link, and should not be a part of the signature. In this case, MSS will be used to guess the type of network hookup according to the [mtu] rules. NEW SIGNATURES: Use '*' for any commodity OSs where MSS is around 1300 - 1500, unless it's fixed. If the value is outside that range, it can be copied literally.
wsize	Window size. Can be expressed as a fixed value, but many operating systems set it to a multiple of MSS or MTU, or a multiple of some random integer. p0f automatically detects these cases, and allows notation such as 'mss*4', 'mtu*4', or '%8192' to be used. Wilcard ('*') is possible too. NEW SIGNATURES: Copy p0f output literally. If frequent variations are seen, look for obvious patterns. If there are no patterns, '*' is a possible alternative.
scale	Window scaling factor, if specified in TCP options. Fixed value or '*'. NEW SIGNATURES: Copy literally, unless the value varies randomly. Many systems alter between 2 or 3 scaling factors.

olayout

Comma-delimited layout and ordering of TCP options, if any. This is one of the most valuable TCP fingerprinting signals. Supported values include:

- eol+n Explicit end of options, followed by n bytes of padding
- nop no-op option
- mss Maximum segment size
- ws Window scaling
- sok Selective ACK permitted
- sack Selective ACK (should not be seen)
- ts Timestamp
- ?n Unknown option ID n

NEW SIGNATURES: Copy literally.

quirks

Comma-delimited properties and quirks observed in IP or TCP headers include:

- df "don't fragment" set (probably PMTUD); ignored for IPv6
- id+ DF set but IPID non-zero; ignored for IPv6
- id- DF not set but IPID is zero; ignored for IPv6
- ecn Explicit congestion notification support
- 0+ "must be zero" field not zero; ignored for IPv6
- flow Non-zero IPv6 flow ID; ignored for IPv4
- seq Sequence number is zero
- ack+ ACK number is non-zero, but ACK flag not set
- ack- ACK number is zero, but ACK flag set
- uptr+ URG pointer is non-zero, but URG flag not set
- urgf+ URG flag used
- pushf+ PUSH flag used
- ts1- Own timestamp specified as zero
- ts2+ Non-zero peer timestamp on initial SYN
- opt+ Trailing non-zero data in options segment
- exws Excessive window scaling factor (> 14)
- bad Malformed TCP options
- pushf+ PUSH flag used
- ts1- Own timestamp specified as zero
- ts2+ Non-zero peer timestamp on initial SYN
- opt+ Trailing non-zero data in options segment
- exws Excessive window scaling factor (> 14)
- bad Malformed TCP options

NEW SIGNATURES: Copy literally.

pclass

Payload size classification: '0' for zero, '+' for non-zero, '*' for any. The packets we fingerprint right now normally have no payloads, but some corner cases exist.

NEW SIGNATURES: Copy literally.

HTTP Signatures

ver	0 for HTTP/1.0, 1 for HTTP/1.1, or '*' for any. NEW SIGNATURES: Copy the value literally, unless you have a specific reason to do otherwise.
horder	Comma-separated, ordered list of headers that should appear in matching traffic. Substrings to match within each of these headers may be specified using a name=[value] notation. The signature will be matched even if other headers appear in between, as long as the list itself is matched in the specified sequence. Headers that usually do appear in the traffic, but may go away (e.g. Accept-Language if the user has no languages defined, or Referer if no referring site exists) should be prefixed with '?', e.g. "?Referer". p0f will accept their disappearance, but will not allow them to appear at any other location. NEW SIGNATURES: Review the list and remove any headers that appear to be irrelevant to the fingerprinted software, and mark transient ones with '?'. Remove header values that do not add anything to the signature, or are request- or user-specific.
habsent	Comma-separated list of headers that must *not* appear in matching traffic. This is particularly useful for noting the absence of standard headers (e.g. 'Host'), or for differentiating between otherwise very similar signatures. NEW SIGNATURES: P0f will automatically highlight the absence of any normally present headers; other entries may be added where necessary.
expsw	Expected substring in 'User-Agent' or 'Server'. This is not used to match traffic, and merely serves to detect dishonest software. If you want to explicitly match User-Agent, you need to do this in the 'horder' section, e.g.: User-Agent=[Firefox]

Passive OS Signature Chart

OS	Version	Platform	TTL	Window	DF	TOS
DC-OSx	1.1-95	Pyramid/NILE	30	8192	n	0
Windows	9x/NT	Intel	32	5000-9000	y	0
NetApp	OnTap	5.1.2-5.2.2	54	8760	y	0
HPJetDirect	HP_Printer		59	2100-2150	n	0
AIX	4.3.x	IBM/RS6000	60	16000-16100	y	0
AIX	4.2.x	IBM/RS6000	60	16000-16100	n	0
Cisco	11.2	7507	60	65535	y	0
DigitalUnix	4.0	Alpha	60	33580	y	16
IRIX	6.x	SGI	60	61320	y	16
OS390	2.6	IBM/S390	60	32756	n	0
Reliant	5.43	Pyrmaid/RM1000	60	65534	n	0
FreeBSD	3.x	Intel	64	17520	y	16
JetDirect	G.07.x	J3113A	64	5804-5840	n	0
Linux	2.2.x	Intel	64	32120	y	0
OpenBSD	2.x	Intel	64	17520	n	16
OS/400	R4.4	AS/400	64	8192	y	0
SCO	R5	Compaq	64	24820	n	0
Solaris	8	Intel/Sparc	64	24820	y	0
FTX (UNIX)	3.3	STRATUS	64	32768	n	0
Unisys	x	Mainframe	64	32768	n	0
Netware	4.11	Intel	128	32000-32768	y	0
Windows	9x/NT	Intel	128	5000-9000	y	0
Windows	2000	Intel	128	17000-18000	y	0
Cisco	12.0	2514	255	3800-5000	n	192
Solaris	2.x	Intel/Sparc	255	8760	y	0

https://web.archive.org/web/20160805103449/http://www.sans.org/security-resources/idfaq/zombiet2.gif

Recommended Internet Sites

- p0f Homepage: http://lcamtuf.coredump.cx/p0f3/
- HoneyNet database of known OS signatures: https://web.archive.org/web/20160718145011/https://www.sans.org/security-resources/idfaq/zombiet2.gif

Please contact the Course Coordinators if you are unable to access any of the Recommended Internet Sites.