Prexy's Proxies

•••

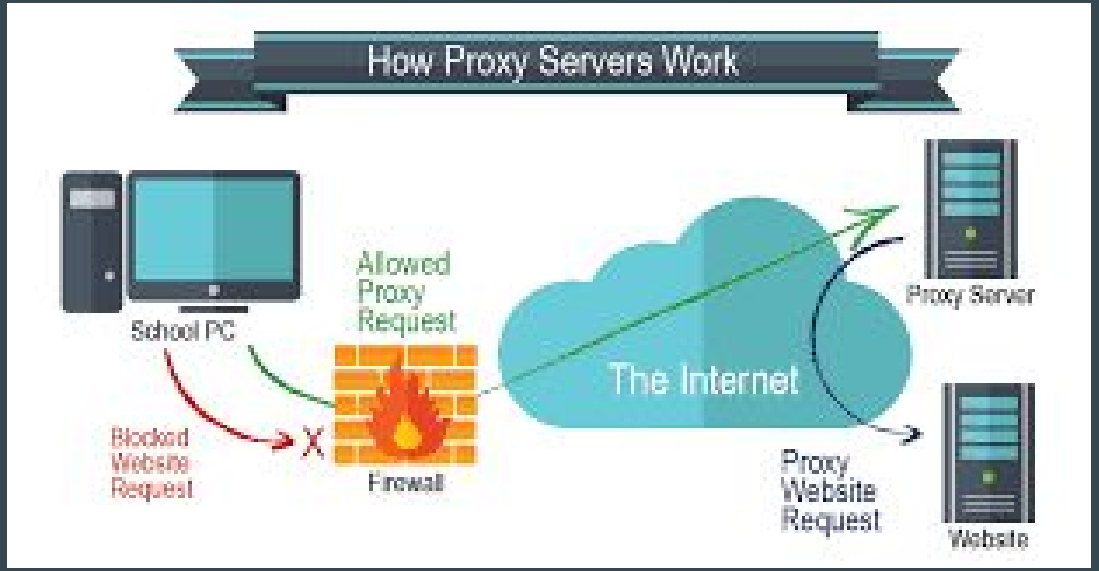
An intro

What iz?

Send packets through an intermediary

Many kinds (SOKS, Tor, Transparant, Reverse)

Necessary knowledge for
CEH Exam!

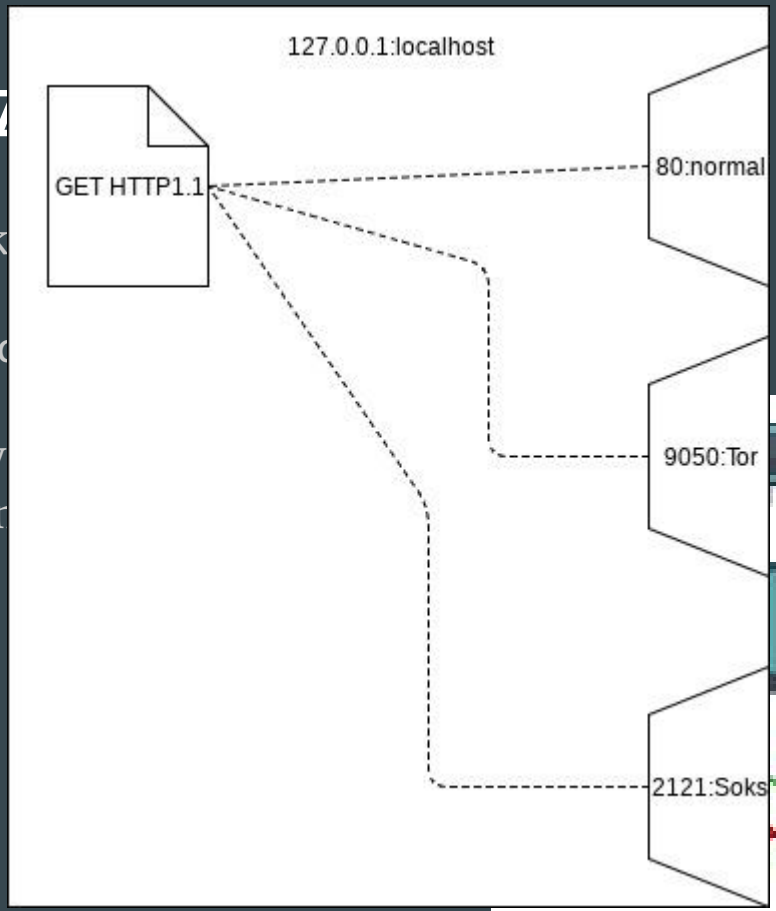


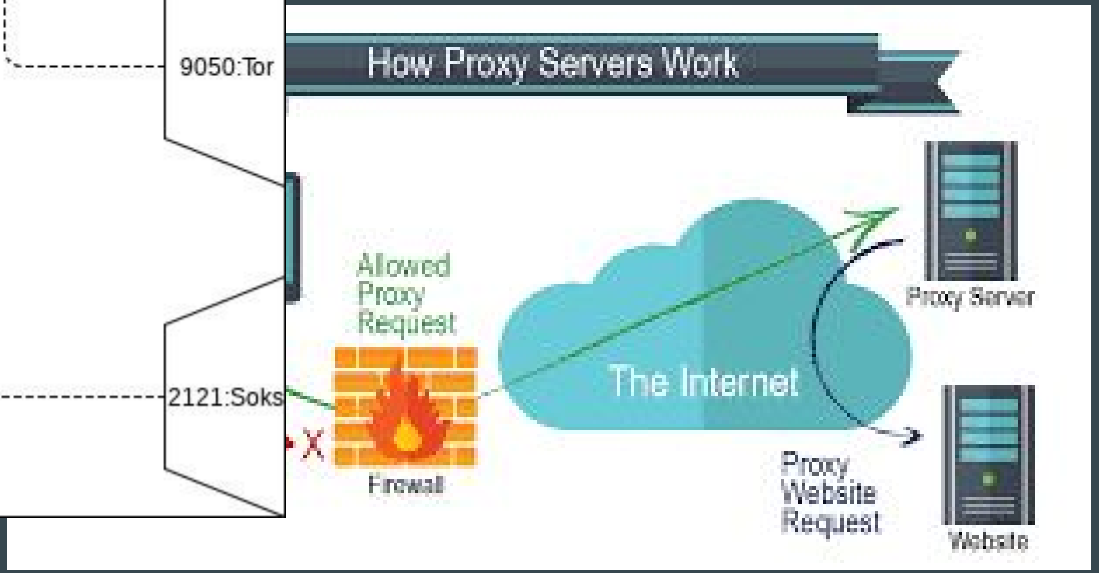
What iz

Send pack

Many kinc

Necessary
CEH Exar





RFC 1919 - Classical IP Proxies

A classical application proxy is a special program that knows one (or more) specific application protocols. Most application protocols are not symmetric; one end is considered to be a "client", one end is a "server".

A classical application proxy implements both the "client" and "server" parts of an application protocol. In practice, it only needs to implement enough of the client and server protocols to accomplish the following:

Classical IP Proxies

a) accept client sessions and appear to them as a server;
b) receive from a client the name or address of the final target server (this needs to be passed over the "client-proxy" session in a way that is application-specific);
c) setup a session to the final server and appear to be a client from the server's point of view;
d) relay requests, responses, and data between the client and server;
e) perform access controls according to the proxy's design criteria (the main goal of the proxy, after all).

Classical IP Proxies

The most visible problem of classical application proxies is the need for proxy-capable client programs and/or user education so that users know how to use the proxies.

Transparent IP Proxies

When somebody thought of modifying proxies in such a way that normal user procedures and normal client applications would still be able to take advantage of the proxies, the transparent proxy was born.

A transparent application proxy is often described as a system that appears like a packet filter to clients, and like a classical proxy to servers. Apart from this important concept, transparent and classical proxies can do similar access control checks and can offer an equivalent level of security/robustness/performance, at least as far as the proxy itself is concerned.

Transparent IP Proxies

The following information will make it clear that small organizations that wish to use proxy technology for protection, that wish to rely entirely on one proxy system for network perimeter security, that want a minimal (or zero) impact on user procedures, and that do not wish to bother with proxy-capable clients will tend to prefer transparent proxy technology.

Organizations with one or more of the following characteristics may prefer deploying classical proxy technology:

Transparent IP Proxies

Organizations with one or more of the following characteristics may prefer deploying classical proxy technology:

a) own a substantial internal IP router network, and wish to avoid adding "external" routes on the network

b) wish to deploy "defense in depth", such as internal firewalls, packet filtering on the internal network

c) wish to keep their DNS environment fully isolated from the "other side" of their proxy system, or that fear that their internal DNS servers may be vulnerable to data-driven attacks

d) use some IP networks that are in conflict with the "other side" of their proxy system

e) wish to use proxy applications that are easily portable to different operating system types and/or versions

f) wish to deploy multiple proxy systems interconnecting them to the SAME remote network without introducing dynamic routing for external routes on the internal network

Transparent IP Proxies

ollowing charact

rtable
ons

cting
ing

firewalls, packet filtering on the internal network

c) wish to keep their DNS environment fully isolated
the "other side" of their proxy system, or that fear that
internal DNS servers may be vulnerable to data-driven
attacks
d) use some IP networks that are in conflict with the "other
side" of their proxy system

Types TLDR;

Transparent - Does not modify the request or response except for means of AuthN and ID. Often do not conceal information about the source request.

Anonymous - Does not reveal source IP to the destination server but will often tell the destination that the request is coming through a proxy (‘X-Forwarded-For’ header)

Highly Anonymous - Does not Reveal source IP or any other information to the destination server. Request Headers from the proxy server appear no different than those sent from the source.

Transparent

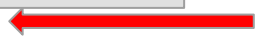
Your IP address is: [redacted]

PHP Server Information

Server Info Type	Server Info Data
Request Method	GET
Request URI	/header_check.php
Request Protocol	HTTP/1.1
Contents of HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Contents of HTTP_REFERER	
Contents of REMOTE_ADDR	[redacted]

Apache Request Headers

Header	Value
Connection	close
Via	1.1 [redacted]
X-Forwarded-For	[redacted]
X-Proxy-Id	1604160538
Accept-Language	en-US,en;q=0.9
Accept-Encoding	gzip, deflate
Dnt	1
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Upgrade-Insecure-Requests	1
User-Agent	[redacted]
Cache-Control	max-age=0
Host	[redacted]



Plain IP Address!!!

Anonymous

Your IP address is:

PHP Server Information

Server Info Type	Server Info Data
Request Method	GET
Request URI	/header_check.php
Request Protocol	HTTP/1.1
Contents of HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Contents of HTTP_REFERER	
Contents of REMOTE_ADDR	35.162.160.108

Apache Request Headers

Header	Value
Connection	close
Host	
Via	1.1 ip-172-31-16-170 AC1F10AA
Accept-Language	en-US,en;q=0.9
Accept-Encoding	gzip, deflate
Dnt	1
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Upgrade-Insecure-Requests	1
User-Agent	
Cache-Control	no-cache
Pragma	no-cache

Highly Anonymous

Your IP address is: [redacted]

PHP Server Information

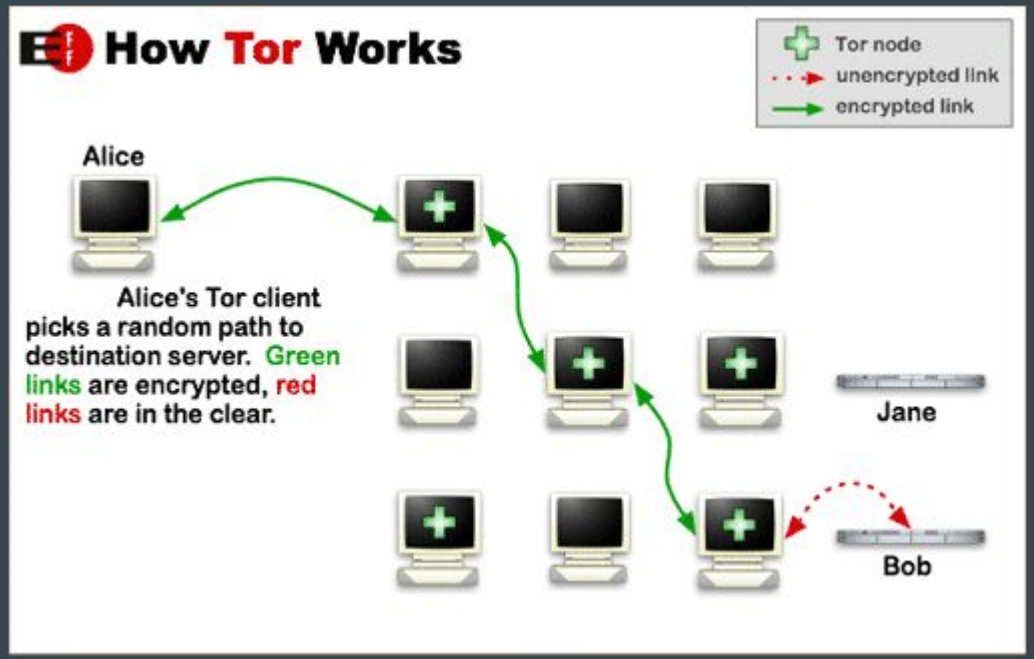
Server Info Type	Server Info Data
Request Method	GET
Request URI	/header_check.php
Request Protocol	HTTP/1.1
Contents of HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Contents of HTTP_REFERER	
Contents of REMOTE_ADDR	216.177.233.181

Apache Request Headers

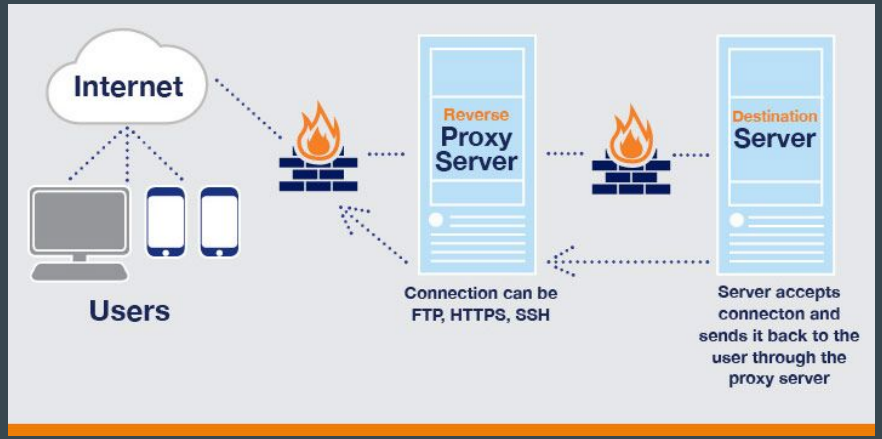
Header	Value
Connection	close
Host	[redacted]
User-Agent	[redacted]
Accept-Language	en-US,en;q=0.9
Dnt	1
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Upgrade-Insecure-Requests	1
Pragma	no-cache

Tor

The wise @Bugg has all the knowledge on this topic.



Reverse Proxies



- Load balancing
  - Traffic cop who directs traffic to different servers
- Web Acceleration
  - Compress and cache data pipeline
- Security/Anonymity
  - Protect infrastructure identities

nginx.com

SOKS

Handle [ANY] traffic b/w a client and a server (dumb explanation)

Dedicated protocol for use as a proxy(unlike HTTP|FTP which are often dedicated to only those services)

V4 has only TCP and no AuthN

V5 has TCP&UDP and AuthN methods

Proxy Chaining

Linking multiple proxies together.

No redundancy, if one chain goes down then the link is broken.

Can be done in Windows just by separating proxy IPs with a space.

Total lag is the sum of lag at each point in the chain.

IMG: infosecinstitute

Malware Analyst Cookbook

Highly recommends using proxies for any networking activities. (wget, ftp, ssh, telnet, etc)

CLI program ‘torsocks’ is a Tor wrapper for all of these (linux only, use apt install)

Use python to create a Soks proxy that uses Tor

Use port-forwarding with an SSH tunnel

Connect to server then use connected server as Soks proxy

Suggests using pre-paid mobile hotspots

REMINDER:

- Figure out what kind of logging/caching policies the service has.

Code can be found @ h-ttps://github.com/mgoffin/malwarecookbook

Hands On

Find a proxy on h-ttps://free-proxy-list.net/ and apply to your system settings.

Upload header_check.php to a web server you own

Header_check.php is modified from the Malware Analyst's source.

Notice the different headers when connected to different types of proxies. Try TOR!

What does a proxy add on to the packet?

Visit the DATDA github and download the ProxyTalk repo. We will be using the daisychain code.

- Start a python web server in the repo dir
  python -m SimpleHTTPServer 8080
- Start the Servers
  python serv.py 8181 8080
  python serv.py 8282 8181
- Start Wireshark!
- Run the Client
  python client.py 8282
- Complete the challenges!

Further Reading

Highly suggest the Malware Analyst's Cookbook

h-ttp://resources.infosecinstitute.com/proxy-chaining/#gref

The files in the notes section of daisychain repo.

Knowledge Links

h-ttp://mewbies.com/how_to_use_a_proxy_socks5.htm

h-ttps://www.alpinesecurity.com/courses/ceh/ceh-module-3-scanning-networks/

h-ttps://technet.microsoft.com/en-ca/library/cc995172.aspx

h-ttp://resources.infosecinstitute.com/proxy-chaining/#gref

h-ttps://panopticlick.eff.org ← find out how identifiable you are by your browser.

*these links have been “de-fanged” a technique where a random symbol is inserted to prevent the hyperlink being clicked and followed. If you are not a robot you should have no problem fixing that