Psiphon Circumvention System Design Paper

Version 0.2 - June 23, 2011

Introduction

The Psiphon Circumvention System is a tool that provides access to Internet content in locations where it is filtered or censored. Psiphon has been designed to balance ease-of-use, propagation and limitation of blocking. The result, we believe offers users a accessible and usable platform for Internet content access, with design considerations to maximize service availability.

Rationale for Development

Psiphon Inc. develops and operates a range of Internet censorship circumvention services. Our Psiphon 2 product (https://launchpad.net/psiphon) is a link-rewriting web proxy. It offers the benefit of ease of use, as users access it using their web browser and don't need to install any software.

Dynamic content is a challenge with link re-writing web proxies. To be effective at circumvention, the proxy must reliably re-write every link produced by a web site. Static parsing can re-write links in HTML, but is not effective with JavaScript and Flash and this content must be stripped from the web site, resulting in a degraded user experience.

Key user communities have indicated a preference for a simple, efficient circumvention tool that offers fully functional access to their favorite web sites and services.

The Psiphon Circumvention System is a new platform that includes a client-side application. Unlike Psiphon 2, users run a client program on their computer or smart phone to access the system. This new product offers the benefits of total system proxying -- using VPN technology, all the traffic on a user's computer or smart phone is reliably routed through the Psiphon system; and advanced resistance to blocking -- using a built-in automatic discovery mechanism.

Design Goals

Automatic Discovery

Each Psiphon Circumvention System client ships with a set of known Psiphon servers to connect to. Over time, clients discover additional servers that are added to a backup server list.

As older servers become blocked, each client will have a reserve list of new servers to draw from as it reconnects. To ensure that an adversary cannot enumerate and block a large number of servers, the Psiphon system takes special measures to control how many servers may be discovered by a single client.

Cross-platform Support

Planned support for the major desktop platforms (Windows, Mac, Linux) and mobile platforms (Android).

Zero Install

Psiphon is delivered to users as a minimal footprint, zero install application that can be downloaded from any webpage, file sharing site or shared by e-mail and passed around on a USB key. We keep the file size small and avoid the overhead of having to install an application.

Custom Branding

Psiphon offers a flexible sponsorship system which includes sponsor-branded clients. Dynamic branding includes graphics and text on the client UI; and a region-specific dynamic homepage mechanism that allows a different home page to open depending on where in the world the client is run.

Chain of Trust

Each client instance may be digitally signed to certify its authenticity. Embedded server certificates certify that Psiphon servers the client connects to are the authentic servers for that client.

User Privacy

Psiphon is designed to respect user privacy. User statistics are logged in aggregate, but no individual personally identifying information, such as user IP addresses, are retained in Psiphon log files.

Agile Transport

Psiphon features a pluggable architecture with multiple transport mechanisms, including VPN and SSH tunneling. In the case where one transport protocol is blocked by a censor, Psiphon automatically switches over to another mechanism.

IPv6 Compatibility

Psiphon is designed to be IPv6 compatible. This ensures the system is ready for the next generation Internet, and in the immediate term offers some additional circumvention capabilities as IPv6-based censorship lags behind the tools used to censor IPv4 traffic.

Statement of Limitations

Bypassing content filters comes with risks. We encourage you to consult expert resources and familiarize yourself with the laws and practices in your region in order to understand these risks, and the possible consequences. The OpenNet Initiative (http://opennet.net/), and Reporters Without Borders (http://www.rsf.org) are two good sources of information.

Psiphon is intended for non-technical users seeking access to denied content. Psiphon traffic is encrypted to avoid blocking. It should not be construed as a way of enhancing the privacy or security of your Internet connection. Psiphon does not provide anonymity and does not provide protection from traffic analysis attacks. Psiphon does not disguise the fact that you are using Psiphon. If you are engaged in politically sensitive behaviour and are at risk of high levels of surveillance, be aware that Psiphon does not have ability to defend you from such attacks.

Psiphon does not defend users against attacks that are already present when accessing the Internet directly, including but not limited to downloading malicious executables, visiting malicious websites that attempt to exploit vulnerabilities in your operating system, browser, and browser plugin-ins etc.

Psiphon does not store usage information in a manner that allows Psiphon to correlate an individual user's IP address with a visit to any individual web site. Psiphon logs aggregate usage statistics on a per in-proxy as well as country/region basis.

Circumvention technologies should be seen as tools in a toolkit, and you should choose a solution based upon your individual needs and circumstances. For help choosing a circumvention solution, visit: www.sesawe.net or read the Citizen Lab's "Everyone's Guide to By-Passing Internet Censorship for Citizens Worldwide" (http://civisec.org/guides/everyones-guides).

Technical Design

Overview and Components

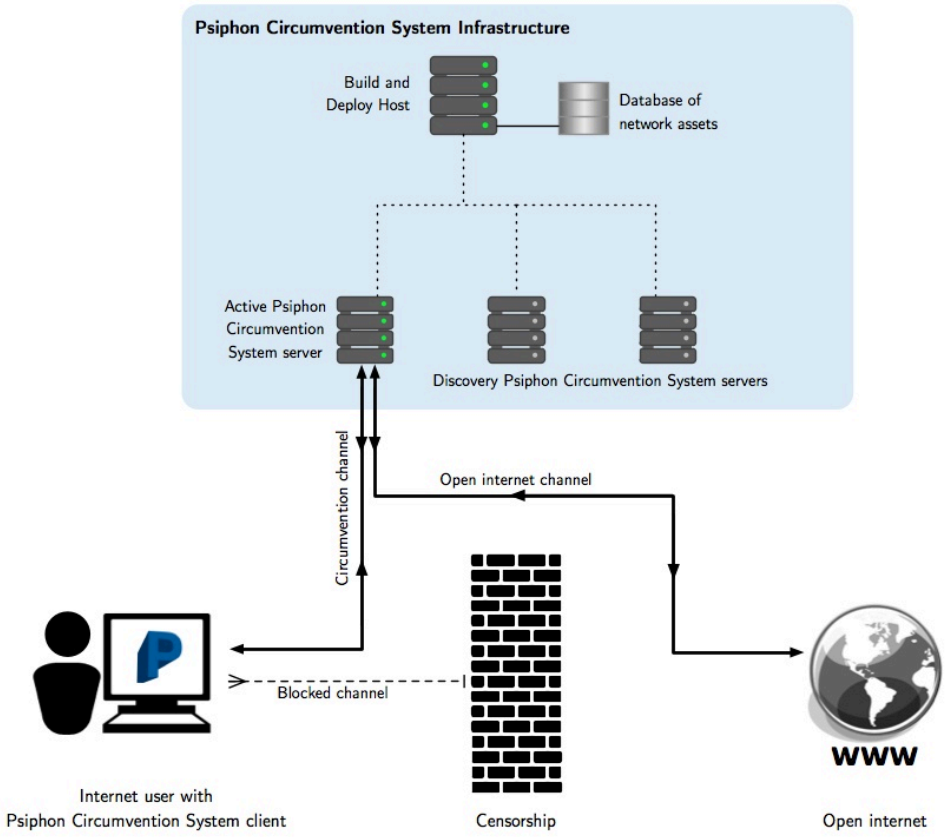
The Psiphon Circumvention System uses a relay-based mechanism to tunnel Internet traffic from the user's computer through a server. The tunneling mechanism is VPN. The client program automatically configures the network settings on the user's computer or smart phone device to route tunnel through one of a set of known servers. The network of servers is centrally managed.

The components of the system are:

- A database of network assets: hosts and servers; sponsorship configuration; discovery

schedule.

- Psiphon Clients. A client is a binary executable. Users obtain clients through download links propagated by Psiphon.
- Psiphon Servers. A server is an instance of a VPN service accessible at some IP address. A server host is a computer that runs one or more servers.
- Build and Deploy Host. A central configuration system that build clients, configures servers, and manages the discovery process.



Network Database and Concepts

Clients

Client ID	A unique identifier for this client build. Used in the discovery process to issue different server IP addresses to clients propagated through different channels. This value is embedded in client builds and passed to the servers as part of the handshake protocol. This should be a random value with sufficient entropy that it can't be guessed.

Propagation Channels	Reserved.

Hosts

Host ID	A unique identifier of the server host. Used in the Servers table to designate servers belonging to this host.
IP Address	IP Address to use to connect to host. Used by automated deployment and configuration scripts.
SSH Username/Password	SSH credential to be used by automated deployment and configuration scripts.
SSH Host Key	Automated connections ensure the host's SSH public key matches this value.

Servers

Host ID	The host this server is located on.
Server ID	Random, human-readable alias used to reference server in place of IP address in stats and reporting -- to keep IP addresses confidential.
IP Address	The IP address of this particular server. Each server, including those on one host, has a distinct IP address.
Web Server Secret	A secret value to be supplied as a GET request parameter when making requests to the web server on this server. When this value isn't correct, the web server responds with a generic 404 error. This is to defend against mass scanning of IP address ranges to find web servers that appear to be Psiphon -- only clients that have properly discovered the server will have this secret.
Web Server Certificate/Private Key	The HTTPS credential for the web server. The certificates are self-signed. There's no CA, as this could be used in a scanning attack. Trust is established by distributing the self-signed public key for each server directly to the clients.
Discovery Client ID	The propagation channel -- i.e., client build/download channel -- this server belongs to.
Discovery Start Time	The date when to start disclosing this server in the discovery protocol. When this field is NULL, the server is a "propagation" server and its connection information is embedded in client builds. When the field is supplied, the server is a "discovery" server and its connection information is disclosed dynamically.
Discovery End Time	The date when to stop disclosing this server in the

	discovery protocol.

Sponsors

Sponsor ID	A unique identifier that's embedded in the client and supplied to the server in the handshake. It's used to determine the set of home pages to return to the client. Unlike the Client ID, this value isn't used in discovery and doesn't have security requirements beyond keeping the sponsor identity private -- i.e., use a random value.
Banner Filename	The location of a banner bitmap that's used at build time. The sponsor's banner appears prominently in the client user interface.

Home Pages

Sponsor ID	The Sponsor ID that is supplied by clients.
Region/Home Page URL	A mapping of region to web site URL. When a client connects, the server performs a GeoIP region lookup and returns the home page for the client's Sponsor ID and current region. The client launches a web browser and loads this page once a secure connection is established.

Psiphon Client

A client is a binary executable that displays a simple UI, establishes VPN connections using the built-in system VPN client (on Windows, Mac, and Android), and launches a sponsor web page. The client also uses local storage on the user's computer or smart phone to store a list of known servers to attempt to connect to.

The client has the following embedded attributes:

- Client ID
- Sponsor ID
- Version number
- Initial Server Entry List

Server Entry List

An initial server list is embedded in the client.

Each server entry is a hex-encoded line containing the following space-delimited values:

- Server IP address
- Web server port number
- Web server secret value
- Web server certificate

Once the initial server list is edited with new discovered servers, the server list will be placed locally in the system registry.

The client sorts its server entry list according to connection success or failure. An entry resulting in a successful connection will be kept at the top of the list. On the other hand, if the client fails to connect to one of the servers, that server will be moved to the bottom of the list.

Client Build Process

An automated build process generates a client build for each: Client ID (propagation channel), Sponsor ID, and platform.

The server entry list embedded in each client build is extracted from the network database based on the Client ID.

Windows client executables are compressed using UPX (http://upx.sourceforge.net/). This minimizes the download size.

On Windows, each client is digitally signed with an Authenticode certificate. This serves as the root of trust:

- First, the user obtains a trusted or signed client;
- Then, the client uses embedded authentication info to connect to trusted servers, and
- Through trusted channels the client learns of more authentication info for additional servers.

Fingerprinting Client Downloads

A censorship threat is firewall DPI rules that detect executable downloads and dynamically extract strings to identify the download as a Psiphon client and also consume the embedded secrets.

This executable compression tool doesn't defend against this; however, this step in the build process may be augmented with a commercial binary obfuscation product, e.g. vmpsoft.com or oreans.com/themida.php.

The executable signature is also a fingerprint that may be identified by firewall DPI rules. We plan to use HTTPS file sharing links, e.g., Google Docs, to help defend against firewall DPI attacks against client downloads.

Psiphon Server

Each Psiphon server consists of a web server to handle functions including discovery and upgrade, and a VPN server for tunnelling.

Detailed information on server host configuration is in the INSTALL document in the open source software distribution.

Server Install/Deploy Process

Automated install and deploy of servers is driven driven by the network database. For each host in the database, SSH credentials are used to log in and copy files and execute commands, including:

- Generating and uploading VPN configuration files for each server on the host.
- Updating the server source code (i.e., web server, etc.)
- Uploading new client builds.
- Uploading network database information used for sponsorship and discovery.

Data is compartmentalized by host before uploading. A host will only receive the client builds containing Client IDs related to the servers on the host. Server entries uploaded for discovery are a subet of servers containing only those servers located on the host or servers that may be discovered by clients whose Client ID is related to the servers on the host.

VPN Transport Security

VPN Design Considerations

We selected VPN as the primary transport mechanism for the Psiphon Circumvention System as it reliably tunnels all Internet traffic on a user's computer. Other options include SSH tunneling of a localhost SOCKS proxy, which will only tunnel the Internet traffic of specifically configured applications and plug-ins that respect the applications configuration.

A number of VPN configurations were considered and measured against these requirements:

- Supported by built-in clients on target platforms and OS versions.
- Transport encryption to defend against keyword-based censorship.
- Robust server authentication to ensure the client, using some embedded or out-of-band distributed credential, can trust the servers it connects to are legitimate.
- Full support for NAT: multiple clients behind a single NAT must be able to connect; multiple clients behind different NATs, and with the same local IP address, must be able to connect.

We selected L2TP/IPSec with PSKs as the VPN configuration. Some rationale for rejecting other configurations:

- **L2TP/IPSec vs. PPTP**. PPTP has weak security properties when using the authentication protocols available on our target platforms (http://en.wikipedia.org/wiki/Point-to-Point_Tunneling_Protocol#Security_of_the_PPTP_protocol). Additionally, Android does not support encryption with PPTP (http://code.google.com/p/android/

issues/detail?id=4706).

- **OS built-in VPN vs. OpenVPN**. OpenVPN meets our security requirements but does not satisfy our zero install requirement. On Windows, a device driver must be installed. On Android, OpenVPN cannot be installed without rooting the device.
- **L2TP/IPSec PSK vs. Certificates**. We were unable to satisfy our security requirements using certificates and the Windows XP L2TP/IPSec client. We found that a Windows XP client would successfully establish a connection with a server that presented the exact same certificate that the client uses. That would allow anybody that obtains the client to man-in-the-middle or masquerade as a legitimate server. We were further disinclined to use certificates because it is not possible to configure the Windows L2TP/IPSec clients to expect a particular server certificate. Instead, any server certificate that is signed by a trusted CA is accepted. We feel that it is possible that our targeted users' computers may have CAs installed that are controlled by their regional censors. That would make it possible for the censor to create a server that will man-in-the-middle or masquerade as a legitimate server.
- **L2TP/IPSec Implementation**.
  Open source L2TP/IPSec implementations have limited support for NAT in certain configurations (http://lists.openswan.org/pipermail/users/2006-May/009487.html). To meet the NAT requirements we selected Openswan and require use of the KLIPS kernel module for IKE.

Standard use of PSK does not meet our security requirements – anyone with the PSK can masquerade as a legitimate server. Therefore we cannot embed a static PSK in the clients. Instead, Psiphon establishes a unique, one-time PSK per client session which is distributed in an out-of-band secure channel, as described in the connection protocol below.

Bootstrapping Unique Shared Secrets Using Out-of-Band Channel

When a client initiates a new session, it sends a "handshake" request to the server's web server. This request changes the server's VPN PSK to a new random value and returns the value to the client.

1. The client makes a web request to the server for new PSK. The HTTPS web request includes authentication of the server using the web server public key included in the server entry for the selected server.
2. A new PSK is generated by the server with `/dev/urandom` or equivalent.
3. Once the new credentials have been generated by the server, it modifies its `ipsec.secrets` record to enable a VPN connection with the new credentials.
4. When the client receives the server response, it configures its local VPN settings to use L2TP/IPSec/PSK according to the information received from the proxy server, and starts the VPN.
5. We rely on PSK in L2TP/IPSec as a robust, bi-directional challenge response that authenticates the server to the client.
6. The server leaves the PSK in place, but each client will request a new PSK before

establishing a VPN connection to ensure it's not at risk from a man-in-the-middle attack from an adversary that obtained the current PSK.

There's a concern that this unique sequence of HTTPS handshake request followed by a VPN-connection sequence would form a unique signature. It has been determined that the effort required to track such a succession is not a practical approach for adversaries with large-scale firewalls.

Discovery

Tor's bridge distribution and discovery strategies (https://svn.torproject.org/svn/projects/design-paper/blocking.html) were considered in the design of Psiphon, along with some new ideas.

As with Tor's bridge relay design, our goal is to achieve availability: clients can connect to a server because they have discovered enough unblocked options. We must balancing discovery against enumeration: if any given client can easily discover all servers, then an adversary can exploit this to enumerate all server IP addresses and block the entire network.

Given that the adversary has full control of the network between Psiphon clients and servers, we cannot design a system resistant to an arbitrarily powerful adversary. Instead we select strategies that we expect will require real world effort on the part of the adversary to defeat: for example, the adversary must reverse engineer some software; or masquerade as clients from a range of IP addresses; or make discovery requests at different times of the day or on certain dates; or locate all client downloads propagated through various channels. Even if the effort is marginal in each case, each strategy makes the system incrementally more resistant to blocking.

All Psiphon stategies fit the effort-for-enumeration framework. Another approach is to try to identify and isolate "bad" users by tracking users, recording which servers are disclosed to users, and flagging users associated with blocked servers. We reject this approach as it requires tracking users with some form of identifier.

Strategy #1: Propagation Channels

Each build of the client includes a unique, random client ID. Client builds are distributed through various propagation channels:

- Download links posted on general interest blogs and public social networks.
- Download links included in email newsletters and on special interest web sites.
- Promotion through other circumvention tools including Psiphon 2.
- Client builds provided to trusted individuals and passed through private social networks.

The client ID is used to assign a disjoint subset of Psiphon servers to the client. As each client must be found and downloaded separately, and with the potential of using different channels for each one of these clients, an adversary must invest effort in finding all client downloads to

complete an enumeration of the network.

Note: when using download link for propagation, be aware risk of an adversary enumerating non-public files in file hosting service: 'Exposing the Lack of Privacy in File Hosting Services' (http://www.usenix.org/events/leet11/tech/full_papers/Nikiforakis.pdf).

Strategy #2: Date Release

Tor's bridge relay design includes a "time-release" strategy. Time-release consists of discovery of different additional servers as a function of request time of day. This requires an adversary to make discovery requests at different times of the day.

Psiphon implements another time-based strategy which releases servers according to a start/ end date schedule. The start date is used to delay discovery and the end date is used to terminate discovery.

When a new client is propagated, new server entries will be discovered by that client soon after the release date. The start/end date cycle is short, so users that obtain the client early in its lifecycle will learn about servers that will no longer be discovered not long after. The adversary must work quickly to locate downloads immediately in order to enumerate all servers. To preserve server resources, the date release cycle is lengthened as the client ages.

The decaying release rate is intended to minimize the server resources allocated to either compromised propagation channels or channels that are never compromised.

Strategy #3: Client IP Address

Different servers are disclosed depending on the client's IP address. This requires the adversary to make discovery requests while masquerading as clients from various IP addresses.

Consider the client's IP address: A.B.C.D. Tor's bridge relay IP address strategy maps bridges to /24 – A.B.C. Tor discards the last octet, arguing that it's too easy for an adversary to gain control of a /24 network and send requests from all 256 addresses.

Psiphon's approach is to discard the upper octets and to hash C.D. In the /24 hash case, a casual adversary making a request from A.B.C.x would instantly get all servers assigned to their nearby network. In the x.x.C.D case, the adversary actually has to make the effort to change the source IP address to get all servers assigned to their nearby network. This is why D is included.

A.B is discarded to avoid partitioning Psiphon servers between countries or ISPs, as we expect that collusion between countries will require a certain amount of effort.

Additional Strategies

Additional discovery strategies held in reserve include:

- **Discovery Rate Release**. Start releasing a server after a scheduled start date. In a variation on Date Release, stop releasing after a certain number of clients have discovered the server instead of after a hard stop date, or use both discovery count and end date. As with Date Release, we hope to gain legitimate users on a server and then stop disclosing it after before it reaches the attention of the adversary. Discovery Rate more accurately reflects and reacts to the growing “popularity” of a server.
- **CAPTCHA Encrypted Server Entries**. Server entry lines are encrypted with random text. The encrypted line and CAPTCHA representation of the text are distributed as the regular server entries are now. Before the client can use the server entry (for the first time) the user must complete the CAPTCHA which provides the decryption key.
- **Email Auto Responder**. Run a service that responds to emails requesting servers. Respond once to each email. Allow only Google, Yahoo, and other emails that require a CAPTCHA for sign-up, as per Tor’s design). The respose is an encoded server entry string that the user pastes into an entry form in the client user interface.

Recycling Servers

With strategies such as Date Release, which stop distributing servers after some time, we may waste resources if certain unblocked servers, no longer distributed, are under used.

We will monitor usage statistics for each server. When a discovery server is initially used, we’ll start to monitor for a drop in usage. Sharp drops may indicate blocking. Gradual drops may indicate under usage. In either case, the best use of the server, once usage drops below a threshold, is to make it re-eligible for discovery with a fresh Date Release (for example) schedule. If the server is blocked, this will have no effect. If the server is under used, this may result in additional usage.

Discovery Configuration

The Build/Deploy host has a complete database of all server information.

Each Psiphon server host contains a subset database of server information for servers that can be discovered by clients related, by Client ID, to the servers it hosts.

Logging and Statistics

In order to generate statistics and usage reports, Psiphon logs web requests and VPN connect/disconnect events.

Ingress Statistics

Ingress statistics of interest include number of sessions per day per region, and range of session durations. Both of these statistics are used to deduce whether servers are blocked in certain regions.

To preserve user privacy, the public IP address for clients are never logged. Psiphon software performs GeoIP lookups against a local database to map client IP addresses to country codes and stores country codes in logs. Logging by VPN software that would normally store client IP addresses is explicitly disabled.

Logging is performed using the syslog facility. We configure syslog to direct all Psiphon logs to a separate log file to facilitate daily download.

Log entries will have the following format:

```
<Date> <Host> handshake <ServerIP> <Region> <ClientID> <SponsorID> <ClientVersion>

<Date> <Host> discovery <ServerIP> <Region> <ClientID> <SponsorID> <ClientVersion>
<DiscoveredServerIP> <UnknownToClient>

<Date> <Host> connected <ServerIP> <Region> <ClientID> <SponsorID> <ClientVersion>
<VPNAddress>

<Date> <Host> disconnected <VPNAddress>
```

For example:

```
Jun 22 15:58:44 host1 psiphonv: handshake 192.168.1.250 CA
3A885577DD84EF13 6C519A29C9B64E58 2

Jun 22 15:58:44 host1 psiphonv: discovery 192.168.1.250 CA
3A885577DD84EF13 6C519A29C9B64E58 2 192.168.1.121 0

Jun 22 15:58:50 host1 psiphonv: connected 192.168.1.250 CA
3A885577DD84EF13 6C519A29C9B64E58 2 10.10.0.2

Jun 22 15:58:53 host1 psiphonv: disconnected 10.10.0.2
```

The “handshake” and “discovery” entries are logged by the web server. Note that there may be many server IP addressed per host, so server information is included explicitly in the log line. These logs are used to determine when and in what regions servers have been discovered.

The “connected” entry is logged by the web server. There is an explicit request from the client to the web server specifically to create this log entry. Clients send this request once a VPN connection has been established. The client supplies the VPN IP address assign to it. The VPN IP address is used as a “session ID” when calculating session duration as the value is unique for the duration of the session and the same value is included in a matching “disconnected” log entry.

The disconnected entry is logged by an ip-down script hooked into pppd. This script is invoked by the system when a VPN connection terminates.

Note: the connected request was conceived as a workaround for technical limitations. Specifically, it's not possible to use an ip-up script to log session start as ip-up isn't provided with any of Client ID, region, or server IP address which are required fields to aggregate session duration against; and, we can't use the "handshake" entries as session start since the VPN session hasn't been established and in fact the VPN protcol could be blocked even when the handshake succeeds.

Egress Statistics

Presently, we do not log egress traffic or statistics. Statistics of interest on this side include, for example, top web sites visited through Psiphon by client region, and top traffic type (e.g., web vs. torrent).

We plan to use tools such as Snort or Netflow to obtain egress statistics. Our privacy policy will be followed and users will not be linked to traffic. In order the aggregate statistics by region or server IP address, we will consider packet marking in iptables to attach a regional identifier to egress traffic logs. Note that such mark will not leave the host server.

Centralized Statistics

The daily statistics from all VPN servers are downloaded to a central system. All the data is then merged into a single database in order to generate usage reports and other statistics.

Sponsorship

Psiphon offers sponsored clients. A sponsor receives custom client builds with a sponsor-specific graphic banner. The banner and a Sponsor ID are embedded in the client builds.

These clients may embed distinct Client IDs for dedicated propagation channels or may share Client IDs for shared propagation channels.

When the client connects and when the user clicks on the banner, the client launches the default web browser and opens web sites specified by the sponsor. The web sites are configured dynamically and delivered to the client by the server as part of the handshake. The client sends its embedded Sponsor ID as part of the handshake request. The server determines the client's region by GeoIP location and returns the home page (or pages) for the Sponsor ID and region.

For instance, a client branded for a customer A, with presence in Vietnam, Iran and China. Home pages for the customer's Vietnamese, Persian and Chinese sites will be served according to the location from which the client is run.

Auto-update

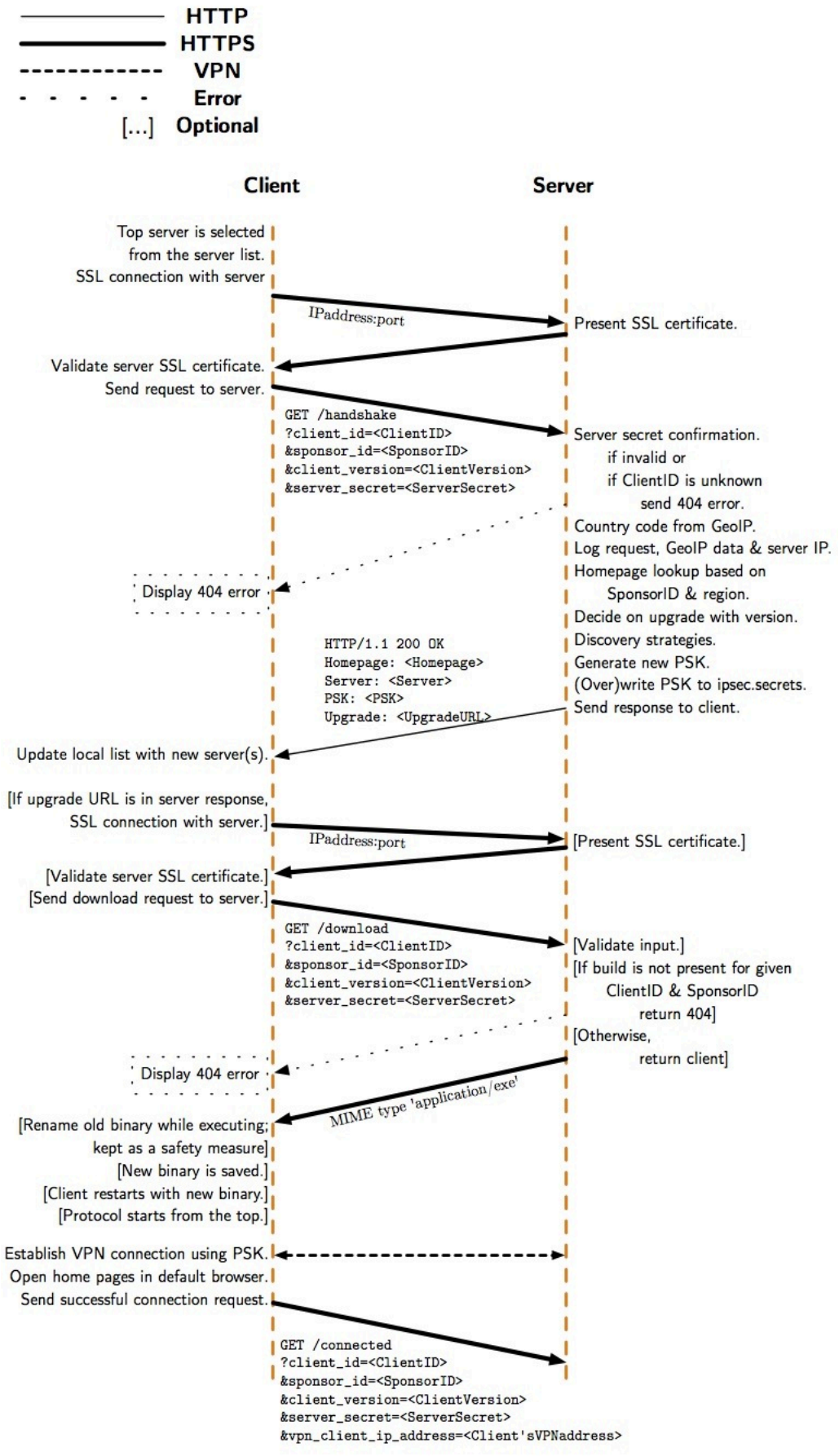
The Psiphon client has the ability to upgrade itself. Upgrades are distributed through the

servers. When it connects to a server, the client submits its current version number as part of the handshake request. When a new version of the client is available, download information for the new binary is returned in the server response. The client makes a second download request to fetch the new version. Once the new binary has been downloaded, the old client is renamed while it is still executing – it is kept for fall-back purposes – and its execution is stopped. The new executable is then run.

Client/Server Connection Protocol

The connection protocol between the Psiphon client and server is used to establish the VPN credentials, discover new servers, deliver default home pages, and perform client upgrades when required.

The step-by-step sequence is as follows.


HTTP
HTTPS
VPN
Error
[...] Optional
Client
Server
Top server is selected from the server list.
SSL connection with server
IPaddress:port
Present SSL certificate.
Validate server SSL certificate.
Send request to server.
GET /handshake
?client_id=<ClientID>
&sponsor_id=<SponsorID>
&client_version=<ClientVersion>
&server_secret=<ServerSecret>
Server secret confirmation.
if invalid or
if ClientID is unknown
send 404 error.
Country code from GeoIP.
Log request, GeoIP data & server IP.
Homepage lookup based on
SponsorID & region.
Decide on upgrade with version.
Discovery strategies.
Generate new PSK.
(Over)write PSK to ipsec.secrets.
Send response to client.
Display 404 error
HTTP/1.1 200 OK
Homepage: <Homepage>
Server: <Server>
PSK: <PSK>
Upgrade: <UpgradeURL>
Update local list with new server(s).
[If upgrade URL is in server response,
SSL connection with server.]
IPaddress:port
[Present SSL certificate.]
[Validate server SSL certificate.]
[Send download request to server.]
GET /download
?client_id=<ClientID>
&sponsor_id=<SponsorID>
&client_version=<ClientVersion>
&server_secret=<ServerSecret>
[Validate input.]
[If build is not present for given
ClientID & SponsorID
return 404]
[Otherwise,
return client]
Display 404 error
MIME type 'application/exe'
[Rename old binary while executing;
kept as a safety measure]
[New binary is saved.]
[Client restarts with new binary.]
[Protocol starts from the top.]
Establish VPN connection using PSK.
Open home pages in default browser.
Send successful connection request.
GET /connected
?client_id=<ClientID>
&sponsor_id=<SponsorID>
&client_version=<ClientVersion>
&server_secret=<ServerSecret>
&vpn_client_ip_address=<Client'sVPNaddress>

1. **Handshake Message**
    a. Establish SSL connection and validate SSL certificate
        - The client selects the first server entry from the server list and establishes an SSL connection to the IP address and port specified in the entry.
        - The client validates that the SSL certificate presented by the server is an exact match to the self-signed certificate embedded in the server entry.
        - No SSL client authentication is performed by the server.
    b. The client sends the “handshake” HTTP GET request.
        - The request must include the Client ID, version number, and Sponsor ID embedded in the client and the server web secret from the server entry.
        - The client also includes a list of known server IPs. This is used by the server to to flag known/unknown servers when logging discovery events.
        - Example:

```
GET /handshake?

client_id=3A885577DD84EF13&sponsor_id=8BB28C1A8E8A9ED9&clie
nt_version=2&server_secret=FEDCBA9876543210
C&known_server=192.168.1.250&known_server=192.168.1.121
```

    c. Server handles the client request.
        - The clients inputs are parsed and validated. If the Client ID isn’t known to the server host or the web secret is incorrect, a generic 404 error is returned to the client.
        - The client’s IP address is mapped to a region, an ISO 3166-1 country code, using a local GeoIP database.
        - The request is logged, including all inputs, region, and server IP address.
        - The <SponsorID, Region> pair is mapped to the table of home pages and the correct home pages are selected.
        - The client version is compared against the latest known version; if there is a newer version then the download link will be returned to the client.
        - The discovery strategies are invoked to select a subset (possibly empty) of servers to reveal to client.
        - The VPN connection is bootstrapped by generating a new pseudorandom pre-shared key (PSK) with /dev/urandom or equivalent and writing the value to /etc/ipsec.secrets.
            - Any existing value is overwritten. Note that there’s a race condition – when two clients connect at once, only one of the PSKs will be in the ipsec.secrets file and one client will get a connection error. We estimate it is not highly likely that many users will connect at once. This issue occurs only at the start of a session, not with every request. Additionally, the Psiphon client has a built-in retry and failover mechanism.
    d. Server sends response.
        - Standard HTTP response header: HTTP/1.1 200 OK

- The response body is text in the following format (example):

```
Homepage: http://news.google.com
Server: 3139...3d3d
PSK: 4ed6...05fd
```

- PSK line: the one-time PSK for the client
- HomePage: zero or more lines containing URLs for the client to load in a web browser.
- Server: zero or more lines containing hex-encoded server entries. This is the discovery information.
- Upgrade: zero or one lines containing the download ID for a new version of the client.

e. Client handles the server response.
- First, the client extracts discovery server entries and updates its local store. In the case where servers are already stored locally, the new entry is still taken if any of the data differs. New server entries are added to the bottom of the client's local list.
- When the "Upgrade" line is present, the client performs the download request and applies the upgrade, restarting itself.
- The client establishes a VPN connection using PSK.
- After the VPN connection is established, the client opens all home pages in default browser.

2. **Download Message**. When an upgrade is indicated in the handshake server response, the client makes another request to download the new client executable to upgrade itself.
   a. Establish SSL connection and validate SSL certificate, as above.
   b. The client sends the "download" HTTP GET request:

```
GET /download?

client_id=3A885577DD84EF13&sponsor_id=8BB28C1A8E8A9ED9&client_version=2&server_secret=FEDCBA9876543210
```

   c. The server handles the client request.
      - Input validation is as above.
      - The server always returns the latest version for the specified Client ID and Sponsor ID. If the client build isn't present, it returns 404.
   d. The client handles the server response.
      - The response body is MIME type 'application/exe'.
      - The old binary is renamed while it is still executing. The old file is left in place for manual recovery/roll-back.
      - The new binary, the server response, is saved in place of the previous executable file.

    - The client application restarts itself. When it starts again, it will begin again with the handshake, step 1.
3. **Connected Message**. The client makes another request which is required by the server to log that the client has successfully established a VPN connection. This request does not perform an action. This request follows the same sequence above, with the addition of a vpn_ip_address input value which indicates, to the server, which VPN client is making this request:

```
GET /connected?

client_id=3A885577DD84EF13&sponsor_id=8BB28C1A8E8A9ED9&client_versi
on=2&server_secret=FEDCBA9876543210&vpn_client_ip_address=<Client's
assigned VPN address, e.g. 10.0.0.2>
```

Risks

In this section we summarize attacks against the Psiphon system. Psiphon components, operating assumptions and typical usage are described in earlier sections. While we identify attacks that seek to block users' access to Psiphon services, we also consider attacks and conditions that may expose Psiphon users to risks against which we cannot defend.

Discovery & Blocking

There are a variety of methods that can be used to identify Psiphon servers and block access to them. Once a server IP address is discovered, an attacker may monitor connections to that IP address and locate Psiphon users. Expected modes of attack include:

Network scanning and/or monitoring: An attacker may scan IP address ranges for web and/or VPN servers that match particular patterns in URL paths, HTML responses, and SSL certificates. They may also monitor traffic flows to identify and block access to IP addresses matching known patterns.

Intercepting communication with users: Email, instant messaging, and SMS may used to communicate client download links to Psiphon users. If communication with a Psiphon user is intercepted, an attacker may be able to obtain clients and discover related server IP addresses or block notifications about new client downloads.

Infiltrating social networks: An attacker may infiltrate a social network and receive a private client download link.

Compromising legitimate Psiphon users' computers: A user may become infected through a virus or trojan that is used to discover and report client usage and server IP addresses to censors.

Exit blocking: A content provider can identify the Psiphon server as a proxy used by out-

of-region or malicious users (e.g., defacing), and consequently block access to all requests originating from the Psiphon server.

In order to reduce the effectiveness of scanning attacks, the Psiphon servers are configured to return generic data and error messages to non-clients; and to use generic authentication information including self-signed web server certificates.

Psiphon cannot defend against traffic analysis attacks. It is possible for an attacker to distinguish Psiphon traffic from normal Internet traffic.

Psiphon recommends that users keep their systems up-to-date with security updates for their operating system and application software. Psiphon users are also advised to install and maintain regularly-updated anti-virus software.

Vulnerabilities / Denial of Service Attacks

An attacker may attempt to compromise Psiphon servers through the exploitation of application software and / or operating system vulnerabilities. If a server is compromised, the attacker may capture all the traffic passing through the server, obtain historical log information, and observe information that isn't logged such as client IP addresses. Points of vulnerability include:

Server OS: An attacker may try to scan and exploit software and / or operating system vulnerabilities on the Psiphon server, for example, by exploiting known vulnerabilities in specific version of the server software running on these servers. An attacker may also try to brute force the administrator password.

Web Server: An attacker may try to compromise the web application using malformed input in an attempt to inject malicious code.

VPN Software: An attacker may try to scan and exploit VPN software. For example, by exploiting known vulnerabilities in a specific version of the software.

Denial of Service Attack: An attacker harnesses an automated script or "botnet" to overload a Psiphon server, resulting in a denial of service for legitimate users.

Since Psiphon servers are hosted by 3rd party providers, we rely on the administrators of these services to ensure that the systems are patched and secured. This may be a faulty assumption.

Psiphon server adminstrators ensure that the operating system and server software are patched with the latest security patches.

Attacking Users

An attacker may attempt to launch attacks against specific users in order to determine the

identity of the user or the content that the user is accessing through Psiphon.

Fake Psiphon: Attacker emulates Psiphon functionality, and is able to lure users and harvest user identifies and Internet traffic.

Traffic Analysis: Timing/correlation attacks against VPN traffic to known Psiphon servers.

Computer Forensics: The Psiphon client leaves traces on the users computer or smart phone -- for example, the discovered server entries.

Psiphon is not a security and anonymity tool and thus cannot defend users from directed attacks.