Freedom Network 1.0 Architecture

Ian Goldberg, Adam Shostack
Zero-Knowledge Systems, Inc.
{ian,adam}@zeroknowledge.com

October 10, 2001

Abstract

This white paper, targeted at the technically savvy reader, offers a detailed look at the entities, protocols, and systems that make up the Freedom Network. It is intended to give the reader an in-depth understanding of how the Freedom system works, and to encourage analysis of the system.

1 Introduction

1.1 This paper

The Freedom product line is designed to be the most integrated, strongest and easiest-to-use privacy system available. This white paper gives the technical reader a deep understanding of each component, and of the system as a whole. This paper can be read on its own, or in conjunction with "Freedom 1.0 Security Issues and Analysis".

This paper is intended to explain exactly how the Freedom Network works, and to contain sufficient information to construct a Freedom compatible client or server. This paper exists in two versions, one with the protocol details and the math ("Freedom Network 1.0 Architecture and Protocols"), and one without ("Freedom Network 1.0 Architecture"). We have published the paper in this way to better serve the intended readers of each version. This is the version without the protocol details.

We start by introducing the entities that make up the Freedom network. We then explain the databases that the various entities in the network use, and then how those entities communicate, starting with AIP to AIP communication, and building from there to client-AIP communication, the telescope encryption protocols, and the way IP, UDP and TCP are handled as they traverse the Internet. We conclude by examining the application layer handling for the protocols that Freedom supports.

This paper concentrates on the protocols as they exist today. There are a number of known issues which we will be addressing over time. Those issues are not always noted here. The current version of "Freedom 1.0 Security Issues and Analysis" will always list those issues that are known to exist from a security or privacy standpoint.

1.2 Freedom overview

The Freedom network is an overlay network which runs on top of the Internet. It uses layers of encryption to allow a Freedom end-user to engage in a wide variety of pseudonymous activity by hiding the user's real IP address, email address, and other identifying information from counter-parties, eavesdroppers, and active attempts to violate the user's privacy.

Users are encouraged to create psuedonyms for each area in which they want to preserve privacy. The nyms that someone uses cannot be tied together. Thus, it is not possible to say if `superman@freedom.net` and `clarkkent@freedom.net` are the same, or different people. Superman is happy with this situation because he doesn't want his supervillian enemies to know about his life. Similarly, when `jobseeker@freedom.net` browses a resume web site, his employer can't see that Clark isn't happy with the working conditions at the Daily Planet, and wants to jump into another line of work.

Freedom protects Clark's privacy by proxying the various supported protocols, and sending those proxied packets through a private network before they are deposited on the Internet for normal service. That private network, as a system, is operated by Zero Knowledge. Individual nodes in the network are operated by Zero Knowledge and our partners, so that no single operator has comprehensive knowledge of what data is flowing through the network.

Thus, the main components of the system are pseudonyms (or nyms) and Freedom Servers. In the next section, we offer precise definitions of these and other entities.

2 Entities

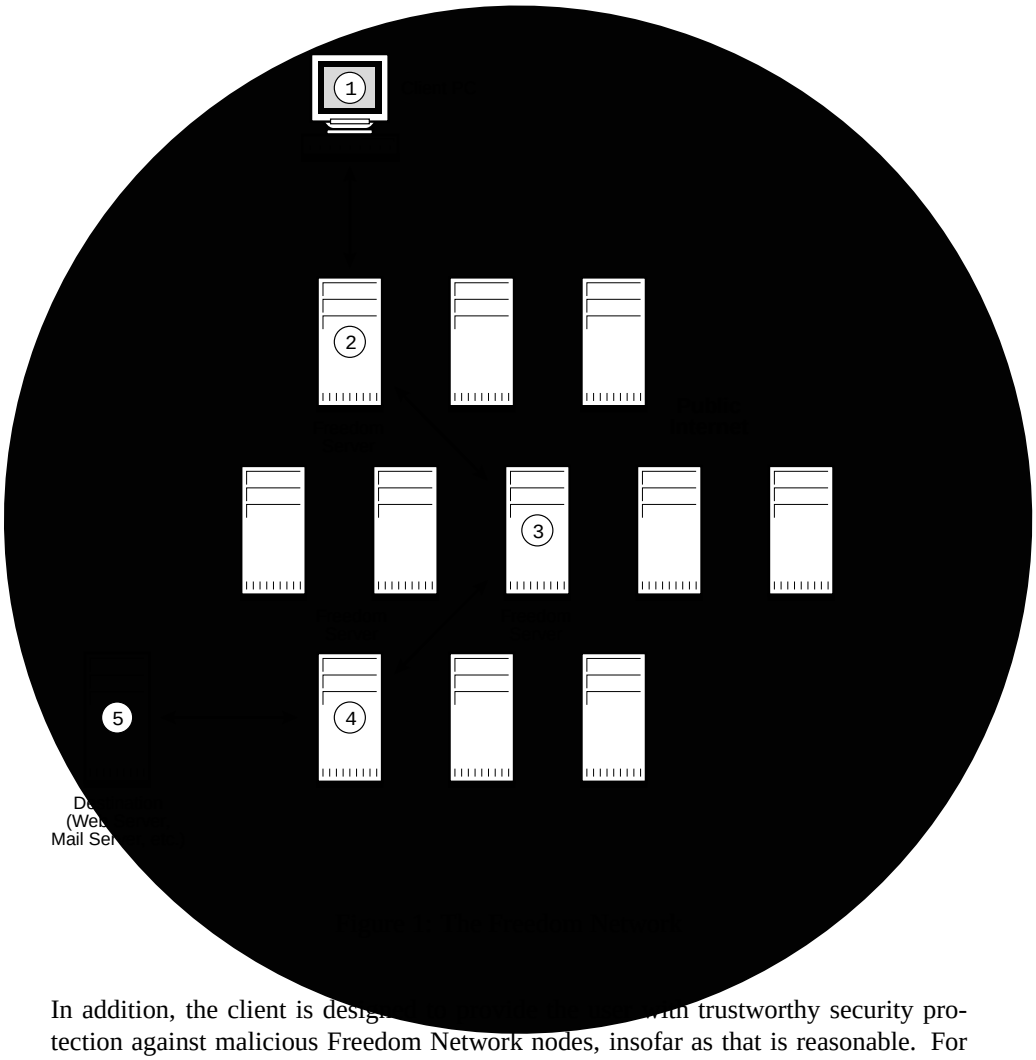
2.1 Nyms

Nyms are the identities that Freedom users assume on the Internet. A nym is defined by a unique email address at Freedom.net, and the associated digital signature key. Zero Knowledge certifies only the uniqueness of the email address. A nym has an email address, a signing key, an encryption key, and zero or more reply blocks. Reply blocks are defined in more depth in section 10, below. A nym is created when a user sends a nym creation token, a signature verification key and an encryption public key to the nym server. This process is detailed in the "Untraceable Nym Creation on the Freedom Network" white paper.

Nym signature keys are 1024 bit DSA keys. They are signed by the nym server signature key (see section 2.5, below). Nym encryption keys are 1024 bit El Gamal keys, which have been signed by the nym's signature key. (Unless otherwise noted, all signature keys are 1024 bit DSA, and all encryption keys are 1024 bit ElGamal.

2.2 Client

The client is a software package, currently provided by Zero Knowledge, which implements a variety of security activities on behalf of the user. It has, hardwired into it, a Master cryptographic key which is used to authenticate all components of the system.



In addition, the client is desig[illegible] with trustworthy security protection against malicious Freedom Network nodes, insofar as that is reasonable. For example, it is not reasonable to expect that the client can offer protection against a fully compromised network. However, much network information is delivered to the client so it can make decisions about which nodes in the network to use, rather than trusting the network to decide which nodes constitute an appropriate path. This decision is driven by a possible tension between definitions of "appropriate" used by the end user and the network operators.

The client has no private keys separate from those of the nyms which use it. It ships with the public parameters of the Freedom Master key hardcoded into it, as well as a pre-loaded cache of server public keys.

2.3 AIP

The Anonymous Internet Proxies (AIPs) are the core network privacy daemons that make up the Freedom network. They pass encapsulated network packets between them-

selves until they reach an exit node. The exit node has a "wormhole" which acts as a proxy, allowing packets to pass between the Internet and the Freedom Network.

The wormhole acts much like a traditional network address translator, with additional proxy functionality to only allow well-formed packets to acceptable server ports.

An AIP has a signature key which is certified by the Monthly key after an out-of-band confirmation process with the server operator. It also has an encryption key, which is signed by its signature key.

2.4 MAIP and FMG

The Mail AIP (MAIP) is a mail transport daemon that works with reply blocks to move mail through the Freedom Network. It uses the Anonymous Mail Transfer Protocol to move messages. AMTP may be described in a forthcoming white paper, or it may be replaced, and the replacement published.

When the FMG-MAIP receives a message that is destined for a non-Freedom address, it applies some security and spam-prevention techniques, such as ensuring that the message is properly signed by the nym, and that the nym is not trying to send out too many messages at once. It then formats the message as a MIME-encapsulated email, and sends it to the destination address.

Incoming mail from the Internet, destined for a nym, is received by a Freedom Mail Gateway (FMG-MAIP), which sits behind an SMTP server. The message is encrypted and chained using each of a nym's reply blocks (more than one reply block may be used to avoid reliability problems should a Freedom server crash). This means that multiple encrypted copies of a message may be delivered to a user's (real) mailbox, but the client software hides this fact from the user, and automatically deletes duplicates.

2.5 Databases

There are a number of support databases which help make the Freedom Network a complete operational system. These databases, collectively referred to as the core services, offer network status and crypto keys and certificates.

Network Information Query and Status Servers (NIQS, NISS) These servers hold the network topology, status, ratings information and some operator data. The NIQS is a read-only server which serves up the digested information. The NISS receives the status packets from the entities on the network and stores everything in the Network Information Database (NIDB). The information that is collected is described in section 11.

Nym Server This server holds all of the nym information and keeps track of all spent tokens. The Nym server keeps a copy of the current nym keys and submits these to the key update server. It also stores a hash of each spent token, in order to prevent token double-spending.

Key Update / Key query servers Key update server receives updates from the nym server. Key query server serves up all public keys on the network. Key update is write only, key query is read only.

Token Server This server keeps the list of active (unredeemed) serial numbers, and generates tokens in exchange for them.

FMG-MAIP The FMG-MAIP has two databases, FMG-Stat and Nym-block. FMG-Stat stores recipient count information which is used for spam control. The nym-block database contains requests that have come in to not allow a nym to send messages to a specific address or domain.

2.6 Master Signature Key hierarchy

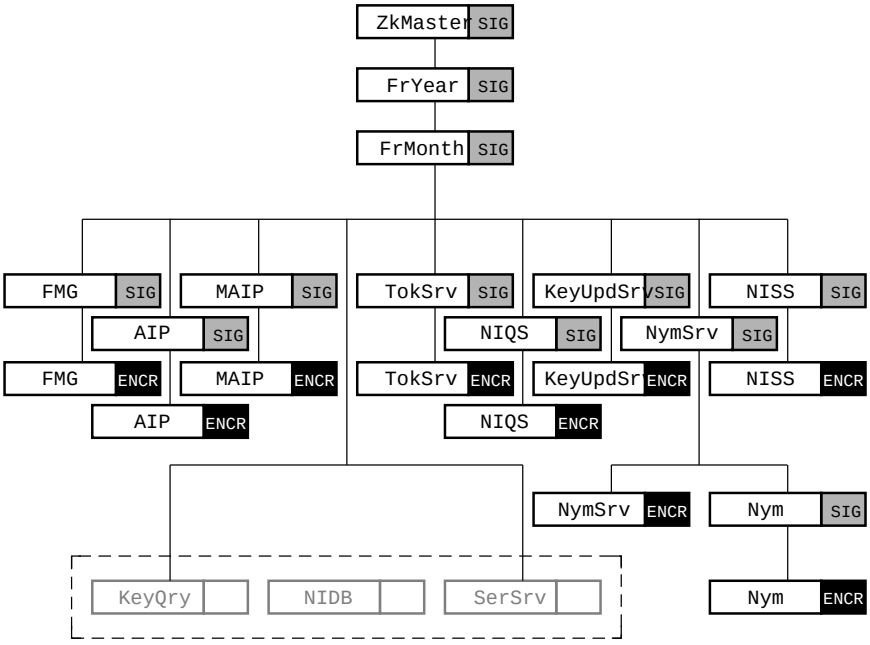


Figure 2: The key hierarchy

There is a set of keys at the top of the hierarchy which are used only for signing. These keys include the Master, Yearly, and Monthly keys. The Monthly key tends to sign a lot of other keys, including signature keys for AIPs, MAIPs, and the FMG.

There is a Master key, whose public parameters are encoded into all Freedom software for reference. It is a 2048 bit DSA key, using the same prime parameters as are used in PGP.

The Master Key signs a Yearly key. [1] The Yearly key is 1024 bit DSA; it is used to sign the Monthly key and client software updates. The Monthly key is used to sign a signature key for each of the FMG, NymServ, TokSrv, KeyUpdSrv, NIQS, and NISS. It is also used to sign the AIP and MAIP signature key for each AIP and MAIP. Each of

[1] The original intent was to rotate all the keys, except for the master, on a reasonable time scale. That code has not been extensively tested, and as such, we may not invoke it in this version of the fielded system.

these keys signs an encryption key for the entity. The NymSrv key also signs the Nym signature keys. The Monthly key, and those keys which it signs, are stored online.

2.7 Procedures for generating and using important keys

The Yearly and Master private keys are stored on a non-networked computer in a physically secure location. There is two-person access control for the locks controlling access to the machine, the passwords needed to log in, and the passphrases for the keys. The process for using them has strong procedural and audit controls.

2.8 Secret-sharing

The Master private key is backed up using secret sharing code by Hal Finney and updated by Ian Goldberg. The shares are 3DES encrypted and stored in the care of certain managers of the company.

3 Database queries

The databases listed in section 2.5 form the core services of the Freedom Network. Various entities in the network query these databases at various times:

NIQS: The NIQS is queried by clients, AIPs and MAIPs in order to determine the current network topology. The clients use the information in order to create routes through the network (as described in 6.1, below); AIPs use the information to determine to which other AIPs they should establish secure links.

Key query server: A client will obtain public keys of other nyms from the key query server in the event that it wishes to send encrypted email to them. In order to protect the identity of the client, the request is made over the Freedom network. In the event that the NIQS reports a new AIP being available, the client will query the Key Query server in the clear. Most servers, including at least AIPs, MAIPs, and FMG-MAIP have reasons to query this server, including authorization and authentication.

Nym server: The Nym Server is queried by the FMG and FMG-MAIP in order to get reply-blocks, check authorizations, and perform spam control.

token server: The Token Server is only queried during the nym generation process.

FMG database: The FMG database is queried by the FMG, to keep track of outgoing mail quotas, as described in section 9.2, and to apply blocking rules.

Queries to the databases are not encrypted or signed by the requestor, and generally, the responses are not encrypted or signed by the database. The NIQS and Nym Server sign their data, but that does not include negative responses, which are generated on the fly. However, the *contents* of the responses have enough information to ensure their accuracy (for example, the key query server will return signed certificates).

4 Inter-AIP link encryption

AIPs on the Freedom network talk to each other with secure links. These links are set up between specific pairs of AIPs that are chosen to minimize network latency and delay; we do not create inter-AIP links that allow you to traverse three continents, as the latency of such a link would make it unusable.

Each AIP has a number of "neighbours"; i.e. the other AIPs to which it has a secure link. As mentioned above, it is not the case that every AIP is a neighbour of every other AIP.

There is no automatic configuration of the graph of which AIPs are neighbours of which other AIPs (the "AIP graph"). The secure links are configured manually with a tool, via the NIQS. An AIP will periodically query the NIQS to get a list of the neighbours it is supposed to have, and will bring its secure links up and down accordingly.

4.1 Setup: Authenticated D-H key agreement

When two AIPs are configured to talk to each other, both will query the key server to get the key for the other AIP. Call the AIPs Alice and Bob. Alice gets Bob's public key certificate from the key query server, and validates it by climbing the key hierarchy until reaching a key that she trusts. Bob mirrors this activity to ensure the exchange is mutually authenticated. Alice and Bob then undergo an authenticated Diffie-Hellman key agreement protocol to derive the encryption key to be used on the secure link. This exchange is done once per hour; old link encryption keys are discarded to ensure perfect forward secrecy; that is, if an adversary records the encrypted traffic, and wants to force Alice or Bob to decrypt it for him, he only has until the hour is up. After Alice and Bob forget the link encryption key, there is no way to recover that traffic.

4.2 Link encryption

Link encryption is applied between node-pairs in order to hide the nature and characteristics of the traffic between them. Data is sent between AIPs in UDP datagrams, typically to port 51101/udp; all but one byte of the body of the datagram is encrypted using a key derived from the Diffie-Hellman key agreement. One byte is sent in the clear, which is merely a flag indicating which key to use to decrypt the packet (this is useful during the transition from one key to the next).

The algorithm to use is specified by the AIPs as part of the key agreement protocol. The default is 128-bit Blowfish; other possibilities are 168-bit 3DES and 184-bit DESX.

5 Client-to-AIP link encryption

A client, upon starting up, will create secure links between itself and one or more AIPs. These secure links are in most ways identical to the Inter-AIP secure links described in section 4. The differences are that:

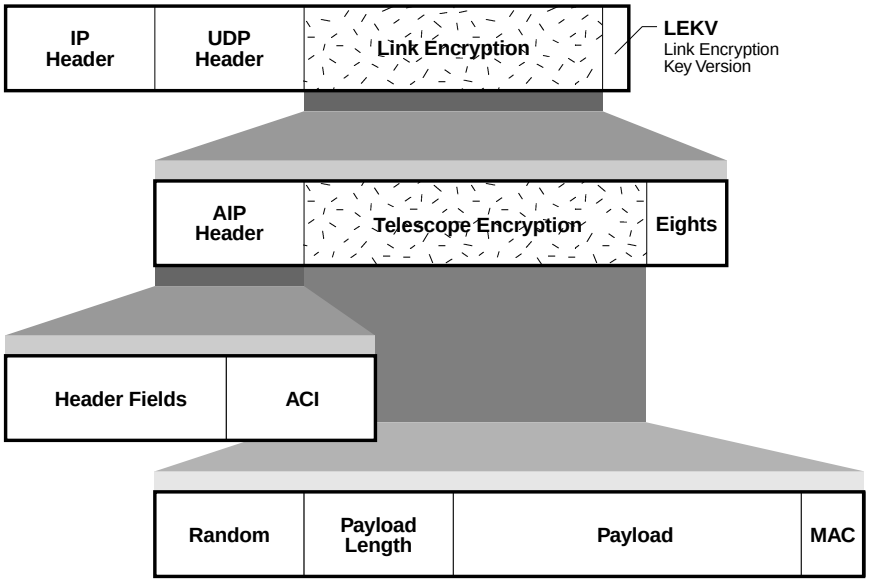


Figure 3: The encapsulation of a data packet in telescope- and link-layer encryption

- they are short-lived (they only exist while the client is running)
- they are under the control of the client (inter-AIP links are under the control of the NIDB)
- the client does *not* sign its Diffie-Hellman parameters, as it has no key it can use to do so privately (the AIP with which it is communicating *does* sign its parameters)

6 Telescope Encryption

Telescope encryption is the set of encryption layers that is designed to provide client-to-wormhole confidentiality. It is called telescope encryption because the layers can be visualized as an old fashioned telescope which collapses in on itself, leading to a set of concentric tubes. Each tube is the layer of encryption that is removed as a packet travels from the client, and added as the the packet travels to the client. In this analogy, the ends of the tube are each connected to an AIP. We use the terms “telescope” and “route” somewhat interchangeably.

There are two types of telescope encryption used in the system, authenticated and anonymous. Authenticated telescopes use a signed ROUTE CREATE request, and create a route that can be used for arbitrary destination hosts. Anonymous telescopes can only be used to connect to certain sets of defined hosts, such as the database servers. They are useful for a set of initialization purposes, when there is no appropriate certificate available with which to create routes. We discuss authenticated telescopes first.

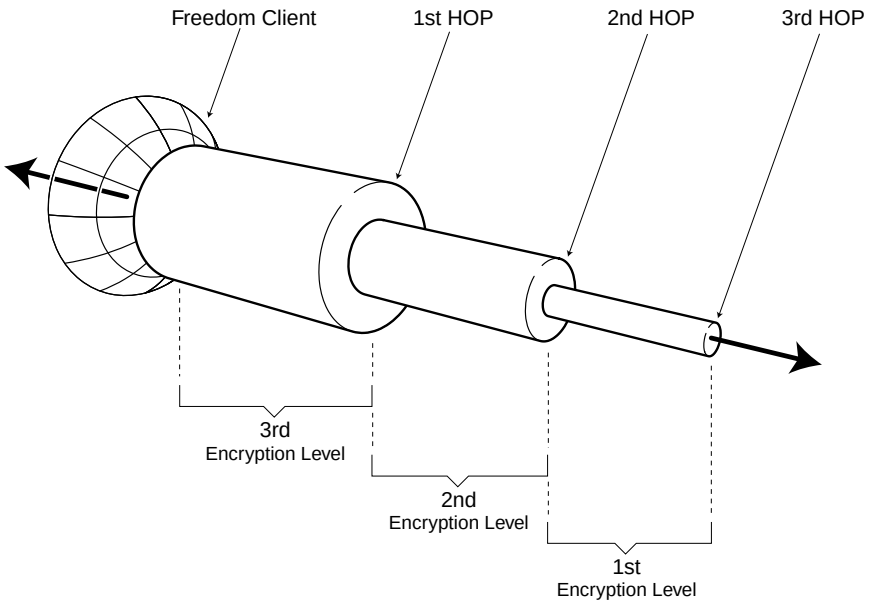


Figure 4: Telescope encryption

An authenticated telescope is one created with a ROUTE CREATE packet signed by a valid nym. Only a nym can create an authenticated telescope.

For an anonymous telescope, the signature and identity fields are left blank (all zeros). Clients and AIPs can make anonymous telescopes.

6.1 Administration

The client selects a chain of AIPs to use to build the telescope. The first AIP must be one to which the client has a secure link, as described in section 5. As well, each pair of consecutive AIPs in the chain must be a pair that has a secure link between them, as described in section 4. The client queries the NIQS to find the set of all AIP pairs with secure links between them (so as to avoid leaking information about which AIP pairs the client is interested in). The last AIP in the chain is called the “last-hop” or “exit” AIP.

6.2 Route Setup

The client constructs a ROUTE CREATE packet which contains secrets to be shared one with each AIP in the chosen chain. Nested El Gamal encryption with the AIPs’ encryption keys is used to securely transmit the secrets (and to ensure that no AIP knows more than who are the previous and next AIPs in the chain).

When the last-hop AIP receives a ROUTE CREATE packet that is correctly signed, it will return a ROUTE CREATE ACK packet, indicating success.

If any AIP receives a ROUTE CREATE packet that is for some reason malformed (for example, the signature is bad), it will send a ROUTE DESTROY packet back towards the client.

6.3 Telescope encryption

The majority of the packets on the network are DATA packets. These, like all packets on the Freedom network, are sent as the payload of a link-encrypted packet, as described in 4.2. These DATA packets are of a fixed size in order to avoid attacks based on size correlations of packets.

6.4 Teardown

If for any reason, an entity needs to tear down an existing route (the entity must be one of the AIPs involved in the route, or the client which created the route), it will send a ROUTE DESTROY packet along the route. If the entity wishing to destroy the route is an AIP in the middle of the route, it should send two ROUTE DESTROY packets: one towards the client, and one towards the Internet.

7 IP layer handling

The client sits at the NDIS level on the Win95 stack. It removes certain identifying data from a packet before encrypting and sending it. This data is the IP source address, and the IP and UDP or TCP checksum. The checksums are removed to avoid brute force attacks on the missing IP address data. The MAC in the route data (telescope) packet ensures that the data has not been corrupted in transit. The wormhole proxy adds in the appropriate source IP, changes the port, and constructs a new set of IP and TCP (or UDP) checksums.

There is a list of acceptable TCP and UDP ports which is enforced by both the client and the wormhole. That set of ports are those needed to allow the supported protocols to run over the Freedom Network. The restriction exists to minimize possibilities for abusive/hacking behavior over the network, and confirms to the principle of that which is not explicitly permitted is denied. Raw IP is explicitly not allowed through the network.

All packets are fragmented to 252 bytes before sending. The wormhole convinces the client that the path MTU is 252 bytes, so the client's standard IP stack will deal with fragmentation for us. The wormhole actually framgents inbound data to this size, and the client's normal stack does the reconstruction. The packet size is intended as a compromise, and will probably be replaced by a different compromise involving multiple packet sizes in the future.

The 252 byte fragment is telescope encrypted for each Freedom Server along the path, link encrypted for the first hop, and then placed in a UDP packet for sending to port 51101.

The data travels over the network, being link decrypted and telescope unwrapped at each point. The data is routed to the next hop by use of an Anonymous Circuit ID (ACI)

mapping table; data coming in over a given ACI is encrypted or decrypted with a key that maps to that ACI, and then link encrypted and sent on its way. If the ACI indicates that packets from that host are sent to the wormhole, then they are. The wormhole will map the ACI into a local TCP (or UDP) port, and map its source port to the source port on which the client is expecting to receive a response. The wormhole will then insert its IP address into the packet, calculate IP and TCP header checksums, and insert the packet onto the Internet.

When a TCP response comes back from an Internet server to the wormhole, the wormhole will break the stream into chunks, add a cryptographic authenticator to the each, and then pass them to the AIP for encryption. The AIP only applys a sinlge layer of encryption. This may be counter-intuitive, if you expect the AIP to add layers, and see them stripped off as the packet travels through the network. This is not done, as the AIP must not know the set of keys it would need to add all the layers of encryption.

The information is passed along the route indicated by the ACIs until it reaches the client, where the software removes the several[2] layers of encryption, inserts the appropriate source address and port back in, recalculates the checksum, and pops the packet back into the IP stack.

8 Nym Creation

The nym creation process is described in detail in the "Untraceable Nym Creation on the Freedom Network" white paper. The process of paying for a nym is intentionally separated from the process of creating a nym, so that we can build a wall which payment identity information does not cross.

9 Application layer handling

9.1 Client-side application proxies

Client-side application proxies are designed to remove identifying information from application streams before it reaches the Internet. This is in contrast to the AIP-side packet-filters, which aim to prevent hacking attempts by the client, and assume that the data stream has already been sanitized. The location and assignment of responsibilities is designed to reduce the need for trust in the system.

Outgoing text in a variety of protocols is scanned by the text scanner. The text scanner looks for strings that match those entered into the client's "Word Scanning" dialog box in a case-sensitive manner. Any matches will result in a warning dialog box being displayed to the user. The data is not sent until the user has approved the sending.

9.1.1 DNS

DNS packets are intercepted and sent over the Freedom network (otherwise, a collaboration between your local DNS server and the wormhole could reveal a lot). There are

[2] This is 4 layers of encryption for a three-hop route; the three telescope and a link layer.

no modifications made to the request.

9.1.2 SSL

SSL packets arrive at the Freedom client already encrypted by the web browser. As such, there is nothing we can do to reliably remove or edit what data they send.

9.1.3 HTTP

HTTP GET and POST messages are sent through the text scanner. Each nym is allocated a seperate cookie jar. The "Referer:" header is left intact.

9.1.4 SMTP/AMTP

Outbound SMTP messages are intercepted by the client-side SMTP proxy. The proxy sanitizes the headers of the message, including replacing the user's real email address with that of a nym. It then checks if all the recipients are Freedom users. If so, it fetches their keys from the Nym Server, and encrypts the multiple times, once for each nym. If there are non-nym recipients, a cleartext copy of the message is kept. The encrypted (and plaintext, if needed) messages are delivered over an anonymous connection to the FMG-MAIP, which applies its processing logic (signature processing, spam and abuse control), and send them either through the MAIP cloud for nym recipients or SMTP to a non-Freedom recipient.

9.1.5 POP3

The POP3 proxy intercepts POP3 requests to the user's pop server. The proxy will collect all mail after authenticating, and keep the user's mail client waiting. The POP3 proxy will correlate messages to prevent the mail client from seeing the redundant versions of messages created by Reply Blocks. As such, the mail client (Eg, Netscape Mail or Outlook) authenticates itself to a local POP3 proxy, that proxy authenticates itself to the POP server, the proxy downloads the mail, decrypts it, and only then will the mail client see a number of messages for reading.

Note that the connection from the POP3 proxy to the user's POP3 server is *not* made through the Freedom Network, but rather is a regular TCP connection over the Internet. This is because the expected POP mailbox is the users standard POP mailbox, and we don't want to associate a nym with that mailbox. (All inbound mail is encrypted so that there is no way for an observer to distinguish to which nym it is addressed.)

9.1.6 NNTP

The body of outgoing news postings is passed through the text scanner. The message is encapsulated by the client news proxy in a mail message, and sent (un)encrypted to a mail2news gateway. Reading Usenet news is accomplished via a web-based news reading site.

9.1.7 Telnet/ssh

Any telnet/ssh based protocol can be passed over the Freedom network to acceptable target ports. The information is run through the text scanner. Uses for this include private contributions to source trees via CVS over SSH, access to MUDs, and other telnet based information services.

9.2 AIP-side application proxies

The server side packet filter in the wormhole is used to offer a level of protection for the Internet from abuse by our customers, by ensuring that outbound packets only reach certain target ports. Our goal is to make it difficult to use the Freedom network for hacking activity, however, since a vulnerability that exists on a target machine may be exploited without the Freedom Network, we simply attempt to be a good neighbor. Ultimately, defending your hosts and networks is your responsibility.

The FMG-MAIP is responsible for outbound spam control for the Freedom network. It does this by placing programmable limits on the volume of mail a given user may send. The limit is initially set based on the class of customer they are: Paid users have a higher quota than trial users. Either type of user's mail quota may be changed by the Zero Knowledge Abuse Center [3] in response to complaints or other issues. The exact quotas are not published so that spammers can not send a number of messages just under the quota. For the same reasons, quotas may vary slightly on a per user basis. The FMG-MAIP also implements the abuse blocking controls, where people can request that they not recieve email from given nyms.

10 Incoming mail handling

Incoming mail for Nyms is received by a Freedom Mail Gateway. The gateway ensures that the mail is for a valid nym which is able to receive mail. (There may be valid nyms which do not have any reply blocks, or for other reasons are unable to receive mail.) If the mail is accepted, a copy of the message is encrypted with each reply block that the nym has created. Each copy of the message will be routed through a series of MAIPs using the AMTP protocol. Each inter-MAIP connection is done over a non-anonymous TCP connection to port 51112.

A reply block contains one or more layers of key, next-hop, message-block tuples. The key is used to encrypt the message before sending the message to the next-hop, which may be a MAIP or a POP mailbox.

Eventually, a number of (encrypted) copies of the message will arrive in the user's POP box (one for each reply block he has set up, unless some AIPs are down). As described above, the POP3 proxy will automatically decrypt the Freedom messages and remove the duplicate copies before passing them on to the user's POP3 client.

[3] `http://www.freedom.net/support/abuse.html`

11 Collecting Network Information

Periodically, the Network Information Status Server (NISS) collects various data about the availability and quality of service of the Freedom Network. The exact details of what information is gathered will be given in a future version of this white paper.

12 Conclusion

Every effort has been made to make Freedom the most integrated, strongest and easiest-to-use privacy system available, and we believe we have achieved this goal. No system is completely infallable, however, but this white paper, in conjunction with "Freedom 1.0 Security Issues and Analysis", will show the reader the extent to which the system is secure under ordinary, and even extraordinary circumstances. We maintain a policy of full disclosure of the system's workings and weaknesses in an effort to be up-front and honest to the community of Freedom users and interested parties.

A Change History

November 29, 1999 Made the following changes:

2.3 AIP keys are signed by the monthly key, not nym server key

2.4 Corrected identity of actor to be FMG-MAIP in para 2

2.5 FMG-MAIP added 2nd database

3.x Added users of various databases

3.key-query added times when clients make cleartext, non-anon queries

3.nym-server expanded nym server, seperated token server to match reality

3.fmg-maip added block rules

4.2 fixed port for data packets

7 the last paragraph of section 7 was substantially cleaned up for clarity in what software performs which actions

9.1 noted that the user can prevent data xfer in dialog box.

9.1.3 added note about cookie jars

9.1.4 mail encryption was wrong, pop handling was insufficiently explained

9.1.6 expanded nntp handling

9.2 fmg-maip abuse control

A Fixed Y2K bug in Change History section

November 23, '99 Released Initial Version.