

A New Standard in Secure Networking

Robert Denier

February 19, 2005

Abstract

A simple yet elegant design is presented to secure a network. Through the use of cryptography each possible communications path is encrypted under a unique key with those keys negotiated through elliptic curve based public key cryptography. Digital signatures are used to verify the origin and integrity of negotiation packets. Key changing is handled automatically. No special control layer is needed beyond that which ordinary Ethernet networks already implement. The use of public key cryptography insures that it is impossible for one station to impersonate another or for any station to eavesdrop on the unicast communications of another pair of stations. Traffic analysis is made extremely difficult for everyone, including insiders, through the use of randomly generated path variables and common packet design. Access control is possible on a per station basis, with each station able to control which other nodes it will accept communications from. The future addition of a userspace program for key management will allow central management and control of systems.

1 License

This document is licensed under a Creative Commons License. The specific license terms can be found at <http://creativecommons.org/licenses/by-nc-nd/2.0/>. This work is for non commercial use only. You may not alter or produce derivative works of this documentation without the author's express permission.

2 Introduction

Today's military systems and to an extent business systems require a degree of network security that is lacking in current standards. Stealth Encryption System (SES) is a network encryption design and implementation to solve the fundamental problem of securing a network at the hardware level. It uses the best cryptographic methods and techniques to implement a simple yet elegant system that insures the integrity, security, and secrecy of network transmissions. Stations are secure both from data interception from outside users and insiders. Impersonation of authorized stations is impossible due to the cryptographic signing performed on key negotiation packets. Periodic key cycling insures that

cryptographic integrity is maintained. Secrecy is maintained with respect to stations not directly involved in the communication. This secrecy includes the contents of the message as well as the true identity of the sender and the recipient of the message.

3 Compatibility

This system is compatible with existing networks that use 48 bit hardware addresses. This has been modified slightly in this design, in that the traditional hardware address is randomized at driver startup. Consider the following figure.

Destination	Source	Length	Path
-------------	--------	--------	------

The first three parts of the header are to maintain compatibility with existing networks. The path field is a codeword representing an entire path from source to destination. In general, every path is unique and the associated codeword is valid for a limited time. In the case where the most significant bit is set it represents a broadcast frame. Path codewords are created with a random number generator and as such are not separable into source and destination parts. The source and destination hardware addresses have been effectively replaced by the path field. To fully implement all the features in this design will require new hardware designed around this path based approach.

The cost of compatibility with existing networks is to reduce the systems traffic analysis resistance. Assigning any kind of fixed address that is transmitted in the clear with every packet automatically yields a great deal of information about the traffic on the network. Randomizing this address whenever the system starts helps, but even this is of limited effectiveness if there is any way for an insider to link this address to something like an internet address or machine name. Since an insider must be able to talk to the machines it is interested in, it can easily get this address and thus analyze the traffic flow patterns on the network. Fortunately SES does not rely on these fixed addresses. Its current use of them is purely a limit of the available hardware. The remaining discussion will be under the assumption that these hardware addresses are not present in the packet header transmitted.

4 Complex Networks

The use of a path variable is well suited to simple wireless networks since no routing is required and everything can be assumed to be sharing the same information channel. Fully implementing this system on more complex networks, while maintaining the anonymous aspect as much as possible, will require extensions to the system so routing paths can be negotiated in a secure and confidential manner. This short section proposes a simple extension to the system that addresses this problem. At this time no code has been developed to test this extension since new hardware would be required for this extension to work in a meaningful way.

4.1 Routing

To simplify this discussion it will be assumed that each station knows, or can find out, the basic network topology. If this is the case, then it is a trivial matter for the system to plan out a path to reach the destination. Further assume that each router on the network is setup for the SES system and has a unique public key. For the sake of discussion assume the path is as follows.

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \quad (1)$$

A will represent the source station, while E will represent the destination station. B,C and D will represent routers along the way. A's goal is to send out a route to the intermediate routers such that the following is true.

1. Packets sent from A reach E through the path specified.
2. No single router knows more than where the packet came from last and where it goes next.

The procedure for the generation of a routing packet is as follows.

1. Begin at the last router in the path. In this case D
2. Generate instructions for it to forward packets to the destination and then encrypt the instructions along with any previous payload using the routers public key.
3. Move up one router. Now the destination is the previous router. Repeat step 2. Continue this until the first router in the sequence has instructions.

A routing packet is sent from the source to the first router. There public key decryption removes the a layer of encryption and sets the route. The leftover payload is sent to the next router and the process is repeated until the route is established. Since each passage through a router removes a layer of encryption it is not possible to form any kind of routing loop although it would be possible for an attacker to form a really long route that went through the same routers many times. The simplest solution for this is to drop packets with ridiculous lengths under the assumption that they are attempting to tie up network resources.

It should be noted that, to an extent, traffic analysis resistance is somewhat in opposition to normal means to control and limit certain types of traffic on a network. This is not a problem. It is merely something to be aware of when considering the actual implementation on a given network. For example, suppose it was necessary to filter at one or more routers using either the source or destination address. The only change required is to add this information to the packets before encrypting them under the respective router's public key. The routers could then be set up to filter as desired.

5 Design

There are two modes of operation. They are referred to as open and closed mode. Open mode only requires stations to have access to the shared key used for broadcast transmissions. Closed mode makes the additional requirement that each station have a list of the public keys and related information for stations allowed to communicate with it. There are two types of packets. Normal packets are those packets that are primarily designed to carry information between stations. Control packets, on the other hand, are primarily designed to update the network and allow continued secure operation. Both kinds of packets use the same outer layer encryption with varying keys depending on the packet type. Control packets use an additional layer of public key encryption to protect the secrecy of their payload whenever possible. For unicast packets, only the destination can directly determine which kind of packet it is although it might be possible to guess packet types by looking at the overall packet length. This is a small consideration since changing their length would be trivial. There are three control packets currently defined. They are modified versions of Address Resolution Protocol(ARP) request and reply packets.

5.1 Keys

In some designs the initialization vector is changed periodically and sent in the clear along with the encrypted packet. Doing so increases the entropy of the resulting packets at the expense of making the packet longer. Unfortunately placing this in the clear also gives an attacker half the information needed to decrypt the data. The better solution is to add entropy directly to the payload before encryption since that way at least the attacker doesn't have access to the entropy bytes.

The keys for private key encryption are ordinary 128 bit Advance Encryption Standard(AES) keys. Every time a new key is randomly generated, the associated initialization vector is also randomly generated. Since both parts are sent securely, the effective cardinality for the key space is 2^{256} possibilities. This is considerably different than sending a changing initialization vector in the clear, since this way it is kept just as secret as the key. The only cost of doing this is a slightly longer key negotiation packet. It should be noted that this does not imply equivalent security to a 256 bit block cipher, although if you needed the security of a 256 bit cipher it would be trivial to make those changes. Whenever a key is referred to in future it will be assumed that this means referring to both the normal AES key and the initialization vector since they are both changed as a unit.

5.2 Initialization Vector Modification

The first 32 bits of the initialization vector is combined with the path variable using the multiply operation. For packets sent with a negotiated key this has a negligible impact on security since the initialization vector, key, and path are

all randomly generated at the same time. It does, however, help insure overall packet integrity since any change of the path variable will cause decryption to fail. Things are a little different for packets sent with the common broadcast key. The common broadcast key cannot be changed unless it is changed for all systems. On the other hand, the path variable on those packets will change on every packet. Since this key will be in use for a long time compared to other keys, the added entropy from incorporating the path variable is a welcome increase in security especially since it comes at the cost of one multiply operation with no change in packet length.

5.3 Normal Packets

Normal packets have the simplest design. Packets are encrypted with the Advanced Encryption Standard(AES) with cipher block chaining. They are encrypted with either the common broadcast key or a key negotiated with public key encryption. The entire packet is encrypted with the exception of the header. The path in the header is associated with a specific key. A 24 bit sequence number is included to prevent the potential of replay attacks. At least two bytes of random data are added in the first encrypted block to generate entropy in the output. The total amount of random data is such that the encrypted payload is a multiple of the cipher blocksize. A 32 bit checksum is included to insure packet integrity and to prevent the possibility of the receiving station decrypting any packet constructed of parts of other packets. The inclusion in the first block of random data, the sequence number, and the overall packet checksum insures data integrity as well as preventing the potential of codebook type lookups. Without the sequence number the entropy from the random data should lead to at least 65 thousand encrypted variations for the same encrypted data. With the sequence number added in the total number of encrypted variations is estimated to be $2^{16+24} = 10^{12}$ for the same plaintext. This effectively makes codebook type lookups impossible since the system will have changed encryption keys long before 10^{12} packets have been sent.

5.4 Control Packets

Control packets originate from ARP request and response frames generated by the network layer. They are used to setup encryption keys between a pair of stations. Public key encryption is used for the main payload, if the sender knows the target's public key. All control packets are digitally signed with the Elliptic Curve Digital Signature Algorithm(ECDSA) for sender validation and message integrity purposes. Finally, control packet data is wrapped in the encryption used for normal packets. This outer layer encryption uses the key for broadcast communications or a key specified by the destination.

5.4.1 Unencrypted Request Packets

Unencrypted request packets are only available in open mode. In closed mode they cannot be used and will be discarded if received. While the payload is sent without the benefit of public key encryption, the entire packet is still digitally signed and encrypted with the outer layer encryption using the broadcast key. Therefore the identity of the sender and the integrity of the information sent can be verified. The payload for these packets includes the original ARP payload, a 32 bit randomly generated identifier, and the public key of the sender. The identifier is included so the receiving station can add it to the response packet to insure that response packets received are based on request packets sent and thus prevent replay attacks.

Open mode, in general, makes no guarantees about the identity of the member's of the network since it does not in general know which stations are allowed on the network beforehand. All that is required to negotiate a connection is knowledge of the common broadcast key. Unencrypted request packets will occur at most once after the system is started since afterwards the public key will be stored in memory allowing the use of the encrypted version. Open mode operation serves two purposes. First it exists as a flexible alternative to closed mode operation at the expense of not being able to verify the sender's information. Second it can be used as a way for the system to build the set of information needed for closed mode operation. All that is required is for one station to communicate with every station on the network briefly for the information to be gathered. It can then be saved to a file and used as a basis for closed mode operation.

5.4.2 Encrypted Request Packets

Encrypted request packets are used whenever the sending station knows the public key of the target node. They are identical to unencrypted request packets with the following exceptions. First, a randomly generated key and initialization vector is added to the payload. Second the entire payload is encrypted under the public key of the target with the target stations public key prepended to the front of the encrypted block. This allows the receiving station to differentiate packets meant for itself. While in principle it would be straightforward to attempt the decryption and find out which packets are meant for itself by which ones the decryption succeeds on, in practice public key decryption operations are very time consuming and should be minimized.

5.4.3 Response Packets

The payload of the modified response packet is encrypted under the public key of the target station. It includes the original ARP response payload as well as the paths, initialization vectors and keys for both the forward and reverse transmission paths. This allows secure communication for normal packets with the target station. Response packets use the previously sent identifier as the path variable to send the packet. This insures the target station can confirm

the packet is in response to its originating request packet and not an attempt at a replay attack. Response packets, along with all control packets are digitally signed to insure message integrity and sender identity.

If the response packet is associated with an encrypted request packet then the outer layer encryption will be encrypted using the key and initialization vector sent in that request packet. This prevents any station, other than the destination station, from being able to directly determine what kind of packet has been sent or even to be able to remove the outer layer encryption. The only way an insider could even guess the packet type would be to check the length, and as mentioned previously it would require little effort to change that. If it is associated with an unencrypted ARP request packet then the common broadcast key will be used for the outer layer encryption. This does not affect the security of the system since the public key encryption guarantees only the target node will be able to decrypt the main payload. It would, however, allow an insider to easily identify the packet type since an insider could remove the outer layer encryption.

5.5 Layered Structure and Complexity

In principle some of the steps used in authentication could be omitted while maintaining the security of the system. For instance, in closed mode a secure key is already available to send the response packet, so there is really no need to encrypt the payload with public key encryption as well. In practice, it is better to include some redundancy. It is possible that some weakness will eventually be found in the public key encryption used in the system. Perhaps occasionally it will become possible to decrypt a packet even without recovering the original private key used for the public key encryption. This would yield the key used on the outer layer encryption of the response packet. Fortunately, since a second layer of public key encryption is around the core payload in the response packet, the system will probably remain secure. This small amount of redundancy helps to insure the secrecy of negotiated keys. One additional benefit to this approach is a separate response packet is not needed for encrypted response packets. This makes system implementation and analysis simpler.

Some other design elements in the system are not in of themselves critical for system security. For instance, sending a randomly generated initialization vector along with every new key is not required to insure the integrity of the encryption. It does, however, help to increase the effective security of the system. Other things that fall in this category are the random bytes added to the payload before encryption, the use of the path variable to modify the initialization vectors, and separate keys for both forward and reverse paths. Each addition, while not in of themselves required for a secure system, enhances the security above where it would be without it.

5.6 Public Key Encryption

Public key cryptography is much slower than private key encryption. Elliptic curve based public key encryption is used in this system. Elliptic curve cryptosystems are much faster than their RSA counterparts. According to [1], “For example, signing speeds range from 20 to 300 times faster on mainstream platforms.” The author’s current elliptic code cryptography implementation is based partly on sample code included with Implementing Elliptic Curve Cryptography by Michael Rosing[2]. Further improvements are required to get this libraries performance comparable to the performance discussed in [1].

Previously, when it was said something was encrypted under public key encryption, what it really meant is a randomly generated private key and initialization vector was encrypted under public key encryption. The actual payload is then encrypted with AES encryption using that key information. This approach allows any sized payload to be encrypted under public key encryption rather than being limited to the 320 bytes available in the specific approach used in this system. Additional entropy is added to the public key encryption operation by filling unused bytes with random data. Since 288 bytes are used between the key, the initialization vector, and the path a total of $2^{320-288} = 2^{32}$ outcomes are possible for a single set of key information.

5.7 Caching

Since public key encryption operations are very time consuming, it is important not to perform them more often than required. The existing infrastructure for handling ARP packets in Linux will continue to send new requests at regular intervals. This is appropriate for the modified ARP request packets as well since wireless stations may lose connection or otherwise miss the previous requests. By sending them on a regular basis, automatic resumption of network connectivity is achieved when the connectivity improves. Both request and response packets are cached to prevent the need to recompute things until it is time for a key to be updated. At that time, a new request packet is sent, that is followed by a response packet with new keys. Furthermore, by not encrypting the same key information repeatedly, an attacker is denied material that might potentially aid in cryptanalysis.

6 Future Work

There are several areas which need further work. The first area is the development of hardware capable of a pure path based approach so that the full system can be utilized. The second area is programming and algorithmic optimizations. Currently all the processing is done in an interrupt handler. This may cause brief periods of unresponsiveness during the handling of control frames. Finally a way needs to be developed to handle the distribution of public key and related information. This can be accomplished with the creation of a userspace

program to run on normal stations, with a supervisory program run on a given station to coordinate everything.

7 Availability

The author has developed an implementation that works as a filter on packets sent to the Orinoco[3] series of wireless drivers for Linux. The implementation is contained in two Linux kernel modules that are implemented in pure C. The system was developed under the 2.6 series of the Linux kernel. The system was tested using Netgear MA311 802.11b PCI cards as well as a Linksys WPC11 card running on a laptop. Since the implementation works as a simple filter on sent and received packets it could easily be ported to other network hardware. Traffic analysis resistance will be limited until network hardware is available to fully take advantage of the path based design.

References

- [1] Certicom, “The elliptic curve cryptosystem for smart cards,” May 1998. URL: http://www.comms.scitech.susx.ac.uk/fft/crypto/ECC_SC.pdf; accessed May 2, 2004.
- [2] M. Rosing, *Implementing Elliptic Curve Cryptography*. Manning Publications, 1999.
- [3] D. Gibson, P. Roskin, and et. al, “Linux orinoco wireless drivers.” URL: <http://savannah.nongnu.org/cvs/?group=orinoco>; accessed May 7, 2004.