ScreenView Cryptography White Paper

Josh Brown

Abstract

This document serves as the cryptographical white paper for ScreenView. While the specification includes this information already, this document excludes thhings that aren't necessarily cryptography related.

1 Introduction

ScreenView consists of two layers of cryptographic protocols each with two sub protocols for reliable and unreliable communication. Server Encryption Layer (SEL) provides end-to-end encryption for all communication between the server and peers. Weak Pre Shared Key, Key Authentication (WPSKKA) Protocol provides end-to-end encryption and authentication for all communication between the peers. WPSKKA can be used independently of SEL, for example in the case of direct connections.

1.1 Definitions

- Host - a peer that is sharing their screen with the Client
- Client - a peer that is viewing the Host's screen
- Peer - a generic term meaning either Host or Client
- Server - the signaling/proxy server used to establish sessions between Host and Client
- Session - a single session between a Host and Client

1.2 Algorithms

2 Server Encryption Layer (SEL)

2.1 Reliable (e.g. TCP)

SEL reliable is simply TLS 1.3 as defined in RFC8446. TLS v1.3 MUST be used. Previous versions of TLS MUST NOT be used.

All further reliable communication between the Peer and the Server occurs over TLS.

2.2 Unreliable (e.g. UDP)

Unreliable communication cannot occur before a session is established. When a session is established with the signaling server, the server generates a session-id. For each peer, the server generates a peer-id and peer-key. All these values are sent to the peer over the SEL reliable channel. session-id is sent to both peers while the peer-id and peer-key are only sent to the respective peer. session-id, peer-id, and peer-key are all 128-bit random values generated from a cryptographically secure random number generator. We can assume that session-id is globally unique throughout all sessions. We can assume that peer-id and peer-key is globally unique throughout all peers in all sessions.
Key derivation is performed as follows:

$G \coloneqq$ session-id

$H \coloneqq$ peer-id

$J \coloneqq$ peer-key

$(SU^{send}_{peer} = SU^{recv}_{serv}, SU^{recv}_{peer} = SU^{send}_{serv}) \coloneqq \mathrm{KDF}_2(\mathrm{HASH}(G \,||\, H \,||\, J), \epsilon)$

$NU^{send}_{peer} = NU^{recv}_{serv} = NU^{recv}_{peer} = NU^{send}_{serv} \coloneqq 0$

Then to send a message, P:

$$\begin{aligned}
&\text{data} \coloneqq \mathrm{AEAD}(SU^{send}_{m}, NU^{send}_{m}, P, \epsilon) \\
&\text{counter} \coloneqq NU^{send}_{m} \\
&NU^{send}_{m} \coloneqq NU^{send}_{m} + 1
\end{aligned}$$

NU_m is an 64 bit counter that does not wrap. After a transport message is sent, if NU_m equals $(2^{64} - 1)$ the TCP connection is dropped. Subsequent UDP messages are not sent.

RFC6479 is used to prevent messages from being replayed.

3 Weak Pre Shared Key (WPSKKA)

Each peer generates an ephemeral keypair. The peers then exchange their public keys over a reliable unencrypted channel.

3.1 Authentication

Next, authentication occurs. The goal of authentication is to ensure the public provided sent by each peer in the key exchange is indeed the public key of the peer.
There are multiple authentication methods.

3.1.1 SRP Static/Dynamic

SRP Dynamic and SRP Static use the SRP v6 PAKE (eventually to be replaced by OPAQUE after standardization) to authenticate the public keys. The only difference between SRP Static and SRP Dynamic is that SRP Static uses a static password chosen by the Host user, while SRP Dynamic uses a randomly generated password. The password is intentionally assumed to be weak, specifically the password is assumed to have a minimum of 20 bits of entropy when dynamic. The password is generated from a cryptographically secure random number generator. In SRP the Host acts aas the server and the Client acts as the client.
To begin, the Host generates a username and salt which are each 128-bit cryptographically secure random numbers. The Host registers the password with itself in accordance

with the SRP protocol. The Host sends the username, salt, and B value to the Client. The Client computes the SRP premaster key. The Client responds to the Host with it's A value and a MAC, where the MAC is

$$\text{HMAC}(\text{client-public-key}, \text{KDF}(\text{srp-premaster-key}))$$

The Host verifies the MAC. THe Host response with a MAC of its own public key. Both the Client and Host have been authenticated.

3.2 Key Derivation

$C_{host} = \text{DH}(E^{pub}_{client},\ E^{priv}_{host})$
$C_{client} = \text{DH}(E^{pub}_{host},\ E^{priv}_{client})$
$(ST^{send}_{host} = ST^{recv}_{client},\ ST^{recv}_{host} = ST^{send}_{client}, SU^{send}_{host} = SU^{recv}_{client},\ SU^{recv}_{host} = SU^{send}_{client}) \coloneqq \text{KDF}_4(C_{host} = C_{client},\ \epsilon)$
$NT^{send}_{host} = NT^{recv}_{client} = NT^{recv}_{host} = NT^{send}_{client} = NU^{send}_{host} = NU^{recv}_{client} = NU^{recv}_{host} = NU^{send}_{client} \coloneqq 0$

3.3 Transport Messages

3.3.1 Reliable

$$\begin{aligned}
&\text{data} \coloneqq \text{AEAD}(ST^{send}_m, NT^{send}_m, P, \epsilon) \\
&\text{counter} \coloneqq NT^{send}_m \\
&NT^{send}_m \coloneqq NT^{send}_m + 1
\end{aligned}$$

Where P is the payload to be transported.

NT_m is an 64 bit counter that MUST NOT wrap. After a transport message is sent, if NT_m equals $(2^{64}-1)$ the UDP and TCP connection MUST be dropped. Subsequent messages MUST NOT be sent.

3.3.2 Unreliable

$$\begin{aligned}
&\text{data} \coloneqq \text{AEAD}(SU^{send}_m, NU^{send}_m, P, \epsilon) \\
&\text{counter} \coloneqq NU^{send}_m \\
&NU^{send}_m \coloneqq NU^{send}_m + 1
\end{aligned}$$

Where P is the payload to be transported.

NU_m is an 64 bit counter that MUST NOT wrap . After a transport message is sent, if NU_m equals $(2^{64}-1)$ the UDP and TCP connection MUST be dropped. Subsequent messages MUST NOT be sent.

RFC6479 is used to prevent messages from being replayed.