BW's eCTF 2025 Satellite TV Design

BWCyberSec

Introduction

This document describes BW's implementation of the Satellite TV encoder and decoder for eCTF 2025. This system aims to build a platform to allow the secure transmission of TV frames, without allowing for interference or interception.

Language Choice

We chose to use Rust for our implementation of the decoder. We find that Rust's memory safety guarantees are valuable for maintaining the security of our design when parsing untrusted input being sent to the decoder. It is important to note that Rust is not a panacea, and that care has still been taken to ensure that our design is secure from other vulnerabilities.

Protocol

All subscription updates and frames sent to the device are encrypted using the XChaCha20-Poly1305 algorithm, and are signed using the Ed25519 algorithm. All encrypted messages have the following header, prefixed before the encrypted packet payload.

Field	Size (in bits)
Nonce	192
MAC Tag	128
Signature	512

On receiving a message that is unable to be decrypted successfully, the decoder will pause until the total time since the command was sent hits 5 seconds.

Update Subscription

The entire Update Subscription message is encrypted, the encrypted payload is structured as follows:

Field	Size (in bits)
Channel ID	32
Start Timestamp	64
End Timestamp	64
Channel Key	256

There is one "decoder" encryption key baked into the decoder, which is derived from the Decoder ID. This key is used for encryption for all Update Subscription messages sent to the decoder.

The decoder will respond with an empty body on successfully registering a subscription.

Decode Frame

The encrypted payload for the Decode Frame packet is prefixed with the 32-bit channel ID. This will be used to determine which channel key will be used to decrypt the rest of the message.

The Decoder will respond with the decrypted frame.

Secrets

BW's implementation of the Encoder and Decoder make use of several cryptographic secrets, as described below.

Keys

- Deployment Key - this is a global key for the deployment. This key will never be sent out to individual decoders, and is used as a master key for deriving decoder keys
  - Decoder Key - this key is derived from the master key and the decoder ID, this key will be baked into the decoder at build time
- Channel Keys - These are global keys, one created for each channel. These will be shared with the decoder, but ONLY when encrypted using a decoder key. This ensures that a pirate subscription cannot be decrypted by a given decoder.
  - Channel 0 key - This is a special case. This key is baked into each decoder alongside the Decoder Key, so that decoders can always decode emergency communications.
- Signing Keypair - This keypair is used for creating cryptographic signatures of all encrypted payloads. This ensures that, even if a channel key or decoder key were to leak, the decoder can ensure that a message came from the genuine encoder.
- Flash Key - this key is generated randomly each time a decoder is compiled. It is used to encrypt the subscription data stored on the flash of the decoder.

Security Requirements

1. An attacker should not be able to decode TV frames without a Decoder that has a valid, active subscription to that channel.
2. The Decoder should only decode valid TV frames generated by the Satellite System the Decoder was provisioned for.

Our implementation meets these requirements by using authenticated encryption and message signatures. This ensures that it is impossible to forge a frame or subscription without the encryption key and signing key. We additionally have channel subscriptions include a different encryption key, meaning that it is impossible to decode a frame without having a subscription, and impossible to decode a subscription without it being destined for your encoder.

3. The Decoder should only decode frames with strictly monotonically increasing timestamps.

The decoder keeps track of the timestamp of the last frame received. It compares all incoming frames to this value, and rejects any which are not larger than the last frames timestamp.

Storage

Storing the subscription updates to flash is done using the Postcard serialization format. Storage uses the last page of available flash. In this page, the first word of the flash is set to a known value (0x4d696b75) to determine whether or not this is the first boot, or if the saving process was previously interrupted. The storage can store up to 8 subscriptions, each for a unique channel.

The storage is encrypted using Chacha20-Poly1305, using a nonce generated using the hardware TRNG, and a flash key, generated at compile time.

The layout of the flash is as follows.

Field	Size (in bits)
Magic	32
Length	32
Nonce	192
MAC Tag	128
Data	(variable)

If the decoder determines that the saving process was interrupted and the storage is corrupted, it will reset and wipe the storage.