Linaro Connect

MADRID 2024 | MAY 14-17 2024

So, you want to encrypt?

David Brown

Our Problem

- Data at rest.
- Recipient and Sender known, some communication before ok, beyond our scope here.
    - Examples: IoT messages, firmware images.
- Sender sends messages, no response.
- Need to ensure authenticity, integrity, privacy.

Our Tools

- Symmetric Cipher
- Hash Function
- Key agreement
- Digital signature

Tools: Symmetric Cipher

Given a secret key, and a message M, encodes the message E(M) such that D(E(M)) recovers the original message, but requires that the function D also have the secret key.

Tools: Hash Function

Given a message M, H(M) returns a fixed sized “summary” of that message. Properties include

- Having H(M) reveals no information about M
- Any change to M, results in a completely different H(M)

Tools: Key Agreement

For the two recipients: A, and B, each has a private key d and Q (d_A, Q_A, d_B, and q_B). Given knowledge of each other’s public key, and their own private key, the two parties can agree on a secret between them x, that cannot be determined by an entity that doesn’t know both public keys and one private key.

Ephemeral key exchange. A random element is introduced that allows a unique key x_i to be generated for each session. Care must be taken as these keys are not authenticated.

Tools: Digital Signatures

Given a private and public key d and Q, sig=Sign(d, M) will give a digital signature, a fixed-sized message. A recipient can use Check(d, sig, M) to validate that the message, M, is identical to the message used to sign.

Generally, the message is not signed but H(M), or similar.

Tools: Authenticated Encryption

A new addition to encryption is authenticated encryption. Given a private key k, and a message M, We have (ciphertext, tag) = E(k, M), where the ciphertext is the encrypted version of the message, and tag is a small authentication tag that confirms that the message was not tampered with. Decryption will check the tag.

Resolves complexity beyond “encrypt then mac” vs “mac then encrypt”, by getting the benefits of both in a single operation. Also generally faster than either encrypt then mac or mac then encrypt.

Choices

This gives us lots of choices. For each tool, there are multiple options available, and within the options, often more choices to make.

It helps to choose standards for these, e.g., AES-GCM-128, SHA-256, ECDHE and ECDSA with the p-256. But, it is also important to be able to be flexible enough to use other options if vulnerabilities are found in any of the components.

Implementation will result in even more choices.

Encoding: the hidden protocol

All of the tools are defined in a mathematical sense. Some result in raw data, such as the symmetric cipher, but the rest generally result in a small set of large numbers.

How this is encoded is also important. Lots of examples from interactive protocols, e.g. TLS 1.0, 1.1, 1.2, and 1.3. 1.0 and 1.1 are deprecated, as vulnerabilities are discovered.

Same issues apply to data at rest. Lesson: don’t invent this ourselves!

The Protocol: COSE

- COSE: CBOR Object Signing and Encryption, RFC 8949
- CBOR: Concise Binary Object Represenation, RFC 9052
- Public-defined formats, lots of scrutiny
- Still in infancy compared to TLS
- Most use of COSE has been about signing, encryption is young
- SUIT (software update for IoT) RFC 9019, and RFC 9124 built around COSE and CBOR

Choices (again)

- COSE is useful, but again has numerous choices
- Pick algorithms and parameters we picked before
- Specify a “profile”
- Balance between generalized decoder, and one specific to a given set of choices. For IoT firmware, may need to be specialized
- Some choices, e.g., Firmware encryption RFC draft specifics lots of details about many of the choices, for the sake of security

It’s Still Hard

- These choices result in some confidence of security
- Given the track record of other protocols, vulnerabilities and weaknesses will almost certainly be found

Linaro Connect
MADRID 2024 | MAY 12-17 2024

Thank you