Overview (1)

- Introduction
- Goal
- Cryptography Overview
  - Symmetric crytography
    - Modes of encryption
      - ECB: Electronic Code Book
      - OFB: Output Feed Back
      - CFB Cipher Feed Back
      - CBC: Cipher Block Chaining
      - Padding before encryption
    - Secret Key ↔ Public Key

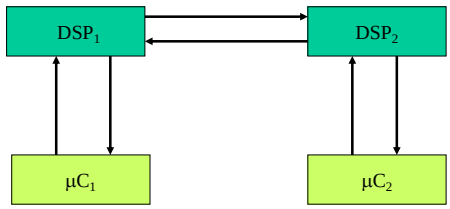
Overview (2)

- Key-Agreement
  - Diffie-Hellman Key-Agreement Protocol
  - One Way Functions
  - Modular Exponentiation
- Secret Key derivation
- Public Key Cryptography
  - Secret Key ↔ Public Key
  - RSA Public Key Algorithm
- Data Authentication
  - Hash Functions
    - MDC: Hash function without key
    - MAC: Hash function with Secret Key
    - retail MAC: MAC based on Block Cipher

Overview (3)

    - Digital Signature
      - Digital Signature with RSA
  - Station to Station Protocol
- Protocol Details: Key agreement
- Protocol Details: Data Broadcast
- Expectations
- References

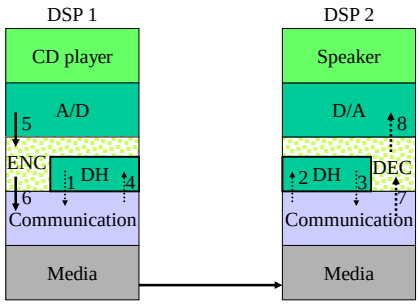
Introduction



Goal

- Setup a secure bidirectional communication channel between DSP_1 and DSP_2
- Secure: data integrity and confidentiality are guaranteed
- DSP_2 only processes information successfully checked
- Confidentiality: Encryption (AES in CBC mode)
- Integrity: retail-MAC based on the AES
- Secret keys (shared keys) negotiated through key agreement
- MAC and encryption keys are derived from the shared secret (after application of hash function SHA-1)
- Digital signatures for the authenticated key agreement use RSA

Protocol stack



Protocol Details: Key agreement

DSP 1

- **1:** Send PING
  - **get input:** x, $Priv_1$, Pub_1, Pub_2
  - **compute msg:** $\alpha^x \bmod p$
  - **send msg**
- **4:** Receive PONG
  - **get msg**
  - **check signature**
  - **compute shared secret:** $(\alpha^y)^x \bmod p$
  - **derive secret keys**

DSP 2

- **2:** Receive PING
  - **get msg** ($\alpha^x \bmod p$)
- **3:** Send PONG
  - **get input:** y, $Priv_2$, Pub_1, Pub_2
  - **compute shared secret:** $(\alpha^x)^y \bmod p$
  - **derive secret keys**
  - **compute msg:** $E_{ENC}(S_{Priv2}(\alpha^y,\alpha^x))\|\alpha^y$
  - **send msg**

Protocol Details: First Data Cell of the Data Broadcast

- First data message sent from DSP_1 to DSP_2 contains in the data field:
  - $E_{ENC}(Data\|\mathbf{S_{Priv1}(\alpha^x,\alpha^y)})$
- Next data messages will only contain $E_{ENC}(Data)$ in that field.

Protocol Details: Data Broadcast

DSP 1

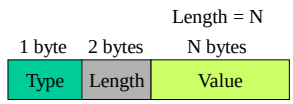
- **5:** Get data (from the A/D converter)
  - **pad the data**
  - **encrypt the padded data**
  - **compute MAC on the encrypted data**
- **6:** Send data to the communication layer

DSP 2

- **7:** Receive data from the communication layer
  - **check integrity (MAC)**
  - **decrypt data**
  - **strip padding**
- **8:** Provide data to the D/A converter

Implementation Hints: Suggested message format

- 4 types of messages
- Structure TLV (Type, Length, Value)



Key agreement: μC_1

- gives to DSP_1 the input needed for the key agreement (x, RSA key pair, RSA public key of DSP_2)
- DSP_1: compute $message_1 = S_{Priv1}(\alpha^x \textit{ mod } p)$
- DSP_1: send $message_1$
- DSP_1: wait for $message_2$ …….
- DSP_1: validate $message_2$ (public key of DSP_2)
- DSP_1: generate secret: $sec_{DH} = \alpha^{xy} \textit{ mod } p$

Key agreement: μC_2

- gives to DSP_2 the input needed for the key agreement (y, RSA key pair, RSA public key of DSP_1)
- DSP_2: wait for $message_1$…….
- DSP_2: validate $message_1$ (public key DSP_1)
- DSP_2: compute $message_2 = S_{Priv2}(\alpha^y \textit{ mod } p)$
- DSP_2: generate secret: $sec_{DH} = \alpha^{xy} \textit{ mod } p$
- DSP_2: send $message_2$

Key agreement: shared secret

- $sec_{DH} = \alpha^{xy} \ mod \ p$
- 2 secret keys are derived from the shared secret [H(•) = SHA-1]:
  - Encryption key (confidentiality protection):
    - $sec_{ENC} = H(sec_{DH} \| 0x0F) \rightarrow E_{ENC}(\bullet)$
  - MAC-Key (integrity protection):
    - $sec_{MAC} = H(sec_{DH} \| 0xF0) \rightarrow M_{MAC}(\bullet)$
    - MAC is computed on the encrypted information

Secure broadcast: μC_1

- DSP_1: produce *Padded* = Data || padding
- DSP_1: compute the ciphertext: $E(sec_{ENC})(Padded)$
- DSP_1: compute the MAC (using sec_{MAC}) on the ciphertext
- DSP_1: broadcast info to DSP_2

Secure broadcast: μC_2

- When DSP_2 receives information:
  - Retrieve the MAC-Key
  - Compute the MAC on the incoming info
  - Compare the computed MAC and the incoming MAC
  - IF both MACs have the same value:
  - DSP_2: Decrypt information (using sec_{ENC})
  - DSP_2: Validate padded information
  - IF ok, info ready for further processing

Expectations

- Ansi-C implementations of **AES** and **SHA-1** are **provided**
- IMPLEMENTATION of:
  - **Protocols**: **key agreement** and **data broadcast**
    - **Retail MAC** based on AES
    - **General modular exponentiation** (Diffie-Hellman and RSA)
    - **Padding** (adding and stripping)
    - **Encryption/Decryption** in **CBC mode**

References

- The handbook of applied cryptography (A. Menezes, P. van Oorschot, S. A. Vanstone)
- Lecture Notes on cryptography (B. Preneel)
- Google
- {Claudia.Diaz, Siddika.BernaOrs, Danny.Decock} @esat.kuleuven.ac.be,
- Offices: 01.65, 01.66, 01.67