Beweisbar sichere Verschlüsselung

ITS-Wahlpflichtvorlesung

Dr. Bodo Möller
Ruhr-Universität Bochum
Horst-Görtz-Institut für IT-Sicherheit
Lehrstuhl für Kommunikationssicherheit
bmoeller@crypto.rub.de

$Id: bsv.ltx,v 1.82 2007/06/20 15:48:25 bm Exp $

Authentisierte Verschlüsselung

- *Grundanforderung* für Verschlüsselung: *CPA*- oder *OTCPA*-Sicherheit, d. h. Ciphertexte verraten nichts über die Plaintexte (außer der Länge)
- Eine weitere sinnvolle Anforderung: *Authentisierung*
- Authentisierung ist oft wichtiger als Geheimhaltung!
- Symmetrische Verschlüsselung mit Authentisierung: „*Authentisierte Verschlüsselung*“ (*authenticated encryption*)

Authentisierte Verschlüsselung (Forts.)

- Eine Form der Authentisierung in der symmetrischen Verschlüsselung: *Integrity of ciphertexts* = *INT-CTXT*-Sicherheit
- Mit INT-CTXT-Sicherheit weiß der Empfänger, dass der Ciphertext von jemandem stammt, der den Schlüssel kennt
- Formalisierung: *INT-CTXT-Angriffsspiel* auf $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ für Angreifer $\boldsymbol{A}$
  - Zufällig erzeugter Schlüssel K (während des Spielablaufs fest)
  - Ver- und Entschlüsselungsorakel $E(\cdot)$, $D(\cdot)$ liefern $\mathcal{E}_K(\cdot)$, $\mathcal{D}_K(\cdot)$
  - Anfrage $D(c)$ nicht zulässig, wenn c von $E(\cdot)$ kam
  - $\boldsymbol{A}$ *gewinnt das Spiel*, sobald $D(\cdot)$ *etwas entschlüsselt* hat (Antwort $\neq \perp$)
  - *INT-CTXT-Vorteil* $\mathrm{Adv}^{\text{INT-CTXT}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}}$ ist die *Wahrscheinlichkeit*, dass $\boldsymbol{A}$ gewinnt

Authentisierte Verschlüsselung (Forts.)

- Ähnliche Anforderung: *INT-PTXT*-Sicherheit (*integrity of plaintexts*)
- $\boldsymbol{A}$ versucht hier, einen Ciphertext zu fälschen, der einen *neuen Plaintext* ergibt
- „Neuer Plaintext“ heißt: $\boldsymbol{A}$ hat Erfolg nur mit einer Anfrage an $D(\cdot)$ mit $D(c) \neq \perp$, wobei der Plaintext $D(c)$ keine Anfrage an $E(\cdot)$ war
- Ein erfolgreicher *INT-PTXT-Angriff* ist stets auch ein erfolgreicher *INT-CTXT-Angriff*
- Ein erfolgreicher *INT-CTXT-Angriff* ist *u. U. kein* erfolgreicher *INT-PTXT-Angriff* (denn beim INT-CXT-Angriff reicht zum Erfolg auch ein Ciphertext, der beim Entschlüsseln einen *alten Plaintext* ergibt)
- *INT-CTXT-Sicherheit* ist also eine *stärkere Anforderung* für die Authentisierung als INT-PTXT-Sicherheit!

Authentisierte Verschlüsselung (Forts.)

- In der vorigen Vorlesungsstunde bewiesen:

  Mit *CPA*-Sicherheit (OTCPA-Sicherheit) und *INT-CTXT*-Sicherheit zusammen haben wir auch *CCA-Sicherheit* (bzw. OTCCA-Sicherheit)

- CCA-Sicherheit (OTCCA-Sicherheit) ist ähnlich definiert wie CPA-Sicherheit (OTCPA-Sicherheit), aber der Angreifer hat zusätzlich Zugriff auf ein Entschlüsselungsorakel (*chosen ciphertext attack*)

- Das ist eine *sehr starke Sicherheitsanforderung*, oft vielleicht unnötig stark (weil es das Entschlüsselungsorakel in der Realität so nicht gibt) – aber wir *verlangen lieber zuviel* als zu wenig

- Nun fehlt noch eine *Konstruktion* für authentisierte Verschlüsselung

- Wir fangen mit der Authentisierung für sich an ...

Message Authentication Codes

- *Message Authentication Code* (*MAC*): Authentisierung (ohne Verschlüsselung) mit einem symmetrischen Schlüssel

- Wieder drei Algorithmen: $(\mathcal{K}, MAC, VF)$, meist nur $\mathcal{K}$ probabilistisch
  - *Schlüsselgenerierungsalgorithmus* $\mathcal{K}$

    Erzeugt Schlüssel K für MAC und VF
  - *MAC-Erzeugungsalgorithmus* $MAC_K(m)$ für $m \in \{0,1\}^*$

    Das Ergebnis ist ein Bit-String einer festen Länge l.
    Es heißt *Tag* oder einfach *MAC* (für die Nachricht m).
  - *MAC-Überprüfungsalgorithmus* (*verification*) $VF_K(m, tag)$

    Das Ergebnis ist ein Wahrheitswert $\in \{\text{true}, \text{false}\}$.

- Wir verlangen *Korrektheit*:

  $$VF_K\big(m, MAC_K(m)\big) = \text{true},$$

  wenn K von $\mathcal{K}$ erzeugt wurde

Message Authentication Codes (Forts.)

- *Sicherheit* von Message Authentication Codes $(\mathcal{K}, MAC, VF)$:
  MAC-Angriffsspiel mit Angreifer $\boldsymbol{A}$
  - K wird mit $\mathcal{K}$ erzeugt
  - $\boldsymbol{A}$ bekommt Zugriff auf ein Orakel für $MAC_K(\cdot)$ und auf ein Orakel für $VF_K(\cdot,\cdot)$
  - Anfrage $VF_K(m, tag)$ nicht zulässig, falls tag von $MAC_K(m)$ kam
  - $\boldsymbol{A}$ *gewinnt das Spiel*, wenn $VF_K(\cdot,\cdot)$ irgendwann (bei einer zulässigen Anfrage) die Antwort true gibt
  - Der *MAC-Vorteil* $\mathrm{Adv}^{\mathsf{MAC}}_{(\mathcal{K},MAC,VF),\boldsymbol{A}}$ ist die *Wahrscheinlichkeit*, dass $\boldsymbol{A}$ gewinnt
- Ähnlich wie das INT-CTXT-Angriffsspiel!
  (Aber ohne Verschlüsselung, deshalb m in der Eingabe zu $VF_K(\cdot,\cdot)$)
- Diese Sicherheitseigenschaft nennt sich auch *strong unforgeability*
  (mit "weak unforgeability" als einer Abschwächung – hier nicht behandelt)

Message Authentication Codes (Forts.)

- Wir werden *MACs als kryptographische Primitive* verwenden
- Viele MACs sind aber auch *kryptographische Konstruktionen*
  (mit Sicherheitsbeweis auf Basis geeigneter Annahmen!)
- Konstruktionen z. B.:
  - *CBC-MAC*: nur sicher für Nachrichten einer festen Länge
  - *CMAC*: baut auf CBC-MAC auf; geeignet für beliebige Längen
  - *HMAC*: MAC auf Basis eines kryptographischen Hashs (z. B. SHA-256)

  Hier keine weiteren Details zu konkreten MAC-Konstruktionen!
- Ein Ansatz: *PRF als MAC* – eine sichere Pseudo-Random Function mit Eingabemenge $\{0,1\}^*$ ist auch als MAC sicher (→ *Übungsaufgabe*)

Encrypt-then-MAC

- Sei $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein Verschlüsselungsschema,
  $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$ ein Message Authentication Code
- Wir konstruieren daraus ein Verschlüsselungsschema $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$
  mit "*Encrypt-then-MAC*":
  - $\mathcal{K}'$ setzt $K \leftarrow \mathcal{K}$ und $K_{\mathrm{M}} \leftarrow \mathcal{K}_{\mathrm{M}}$, gibt das Paar (K, K_{M}) aus
  - $\mathcal{E}'_{(K,K_{\mathrm{M}})}(m)$ setzt $c \leftarrow \mathcal{E}_K(m)$, $t \leftarrow MAC_{K_{\mathrm{M}}}(c)$, gibt $c \,\|\, t$ aus
  - $\mathcal{D}'_{(K,K_{\mathrm{M}})}(c')$ zerlegt $c' = c \,\|\, t$ (eindeutig, weil t die Länge l hat),
    setzt $m \leftarrow \mathcal{D}_K(c)$, überprüft $VF_{K_{\mathrm{M}}}(c, t)$, gibt m aus im Fall true; $\perp$ sonst
- Die *Korrektheit* ist schnell nachgeprüft:

  $$\mathcal{D}'_{(K,K_{\mathrm{M}})}\big(\mathcal{E}'_{(K,K_{\mathrm{M}})}(m)\big) = m,$$

  falls $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ korrektes Verschlüsselungsschema ist und $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$
  korrekter MAC
- *Sicherheit*?

Encrypt-then-MAC (Forts.)

- Sicherheit von Encrypt-then-MAC:
  - CPA-Sicherheit (OTCPA-Sicherheit)
  - INT-CTXT-Sicherheit
- Gegeben ein CPA-sicheres (OTCPA-sicheres) Verschlüsselungsschema $(\mathcal{K}, \mathcal{E}, \mathcal{D})$
  und irgendeinen Message Authentication Code $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$, ist die
  "Encrypt-then-MAC"-Konstruktion $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ CPA-sicher (OTCPA-sicher)?
- Gegeben ein Verschlüsselungsschema $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ und
  einen sicheren Message Authentication Code $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$,
  ist die "Encrypt-then-MAC"-Konstruktion $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$
  INT-CTXT-sicher?

Encrypt-then-MAC (Forts.)

- Sei $\boldsymbol{A}$ ein LoR-(OT)CPA-Angreifer auf das "Encrypt-then-MAC"-Verschlüsselungsschema $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$.

  Wir beschreiben einen LoR-(OT)CPA-Angreifer $\boldsymbol{B}$ auf das zugrundeliegende Verschlüsselungsschema $(\mathcal{K}, \mathcal{E}, \mathcal{D})$:

  - $\boldsymbol{B}$ erzeugt selbst einen MAC-Schlüssel: $K_{\mathrm{M}} \leftarrow \mathcal{K}_{\mathrm{M}}$
  - $\boldsymbol{B}$ bearbeitet alle Verschlüsselungsorakel-Anfragen von $\boldsymbol{A}$ mit Hilfe seines eigenen Verschlüsselungsorakels und mit $MAC_{K_{MAC}}(\cdot)$
  - $\boldsymbol{B}$ übernimmt das Ausgabebit von $\boldsymbol{A}$

- So kann $\boldsymbol{B}$ den Angreifer $\boldsymbol{A}$ exakt im LoR-(OT)CPA-Angriffsspiel auf $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ laufen lassen:

  $\boldsymbol{A}$ bekommt dann Linksverschlüsselungen, wenn $\boldsymbol{B}$ Linksverschlüsselungen bekommt; analog für Rechtsverschlüsselungen.

  Der Vorteil stimmt somit überein:

  $$\mathrm{Adv}^{\text{LoR-(OT)CPA}}_{(\mathcal{K}',\mathcal{E}',\mathcal{D}'),\boldsymbol{A}} = \mathrm{Adv}^{\text{LoR-(OT)CPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{B}}$$

- *Also:* $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ ist (OT)CPA-sicher, falls $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ (OT)CPA-sicher ist

Encrypt-then-MAC (Forts.)

- Sei $\boldsymbol{A}$ nun ein INT-CTXT-Angreifer auf das "Encrypt-then-MAC"-Verschlüsselungsschema $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$.

  Wir beschreiben einen MAC-Angreifer $\boldsymbol{B}$ auf den zugrundeliegenden Message Authentication Code $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$:

  - $\boldsymbol{B}$ erzeugt selbst einen Verschlüsselungs-Schlüssel: $K \leftarrow \mathcal{K}$
  - $\boldsymbol{B}$ bearbeitet alle Verschlüsselungsorakel-Anfragen von $\boldsymbol{A}$ mit $\mathcal{E}_K(\cdot)$ und mit Hilfe seines eigenen MAC-Orakels;

    $\boldsymbol{B}$ bearbeitet alle Entschlüsselungsorakel-Anfragen von $\boldsymbol{A}$ mit Hilfe seines eigenen Verifizierungsorakels und mit $\mathcal{D}_K(\cdot)$

- So kann $\boldsymbol{B}$ den Angreifer $\boldsymbol{A}$ exakt im INT-CTXT-Angriffsspiel auf $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ laufen lassen;

  gewinnt $\boldsymbol{A}$ mit einer Fälschung im INT-CTXT-Angriffspiel, so gewinnt auch $\boldsymbol{B}$ im MAC-Angriffsspiel

- Für den Vorteil gilt also:

  $$\mathrm{Adv}^{\text{INT-CTXT}}_{(\mathcal{K}',\mathcal{E}',\mathcal{D}'),\boldsymbol{A}} \leq \mathrm{Adv}^{\text{MAC}}_{(\mathcal{K}_{\mathrm{M}},MAC,VF),\boldsymbol{B}}$$

- *Also:* $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ ist INT-CTXT-sicher, falls $(\mathcal{K}_{\mathrm{M}}, MAC, VF)$ sicher ist