Anonymity and one-way authentication in key exchange protocols

Douglas Stebila

Queensland University of Technology

Joint work with Ian Goldberg (University of Waterloo) and Berkant Ustaoglu (Izmir Institute of Technology)

Monday September 17, 2012

*Designs, Codes and Cryptography,* online first, http://dx.doi.org/10.1007/s10623-011-9604-z

I.G. acknowledges funding from NSERC and Mprime.
Outline

Key exchange in Tor

Security goals

Security model

Protocols

Conclusions
Key exchange in Tor
 Tor circuit establishment

To establish a Tor circuit, a client Alice does the following:

1. Alice picks a Tor node $X$ and establishes an encrypted authenticated channel with $X$

2. Alice picks a second Tor node $Y$ and establishes an encrypted authenticated channel with $Y$, tunneled via $X$

3. Alice picks a third Tor node $Z$ and establishes an encrypted authenticated channel with $Z$, tunneled via $Y$

\ldots

$k$. Alice relays her communication through nodes $X$, $Y$, $Z$, \ldots, $W$, with the final exit node $W$ relaying communication to/from the destination address.
Tor circuit establishment

1

\[\text{C} \rightarrow \text{X} \rightarrow \text{Y} \rightarrow \text{Z} \]

\[\text{C} \leftarrow \text{X} \leftarrow \text{Y} \leftarrow \text{Z} \]

\(\text{\AA}\text{verlier and Syverson, PET 2007.}\)
Key exchange in Tor

Tor authentication protocol (TAP)

A trusted PKI allows Alice to determine node \( n \)'s public encryption key \( pk_n \)

1. Alice picks \( x \leftarrow \mathbb{Z}_q \)

2. Alice sends \( c \leftarrow \text{Enc}_{pk_B}(g^x) \) to Bob.

3. Bob computes \( m \leftarrow \text{Dec}_{sk_B}(c) \), range checks \( m \), picks \( y \leftarrow \mathbb{Z}_q \), and sends \( a \leftarrow g^y \) and \( b \leftarrow f(m^y) \) to Alice

4. Alice range checks \( a \) and that \( b = f(a^x) \)

5. Shared session key: \( a^x = m^y \)
Security of TAP

- Assume $\Pi$ is an IND-CPA-secure, reaction-resistant encryption scheme and CDH in $\mathcal{G}$ is hard.
- TAP is secure:\footnote{Goldberg, PET 2006.}
  - There exists no p.p.t. algorithm $M$ such that, for a random output $(pk, sk)$ of $\Pi$.KeyGen and a random exponent $x$, $M(pk, g, Enc_{pk}(g^x)) = (a, a^x)$ for some $a$ with non-negligible probability.
Security of TAP

- Assume $\Pi$ is an IND-CPA-secure, reaction-resistant encryption scheme and CDH in $\mathcal{G}$ is hard.

- TAP is secure:\footnote{Goldberg, PET 2006.}
  - There exists no p.p.t. algorithm $M$ such that, for a random output $(pk, sk)$ of $\Pi$.KeyGen and a random exponent $x$, $M(pk, g, \text{Enc}_{pk}(g^x)) = (a, a^x)$ for some $a$ with non-negligible probability.

- Non-standard security definition.
  - Customized to protocol construction.
  - Key recovery, not session key indistinguishability.
“Fourth protocol” of Øverlier and Syverson (PET 2007)

Client $\hat{A}$  

Server $\hat{B}$

long-term private key $b$,  

public key $B = g^b$

1. select $sid$

2. $x \leftarrow \mathbb{Z}_q$

3. $X \leftarrow g^x$

4. $\xrightarrow{X,sid} y \leftarrow \mathbb{Z}_q$

5. $\xleftarrow{Y,sid} k \leftarrow (BY)^x$

$\xrightarrow{Y,sid} k \leftarrow X^{b+y}$

Proposed for, but never used, in Tor circuit establishment.
Insecurity of Øverlier and Syverson’s “fourth protocol”

<table>
<thead>
<tr>
<th>Client $\hat{A}$</th>
<th>Attacker $\hat{M}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. select $sid$</td>
<td>Bob’s public key $B = g^b$</td>
</tr>
<tr>
<td>2. $x \leftarrow \mathbb{Z}_q$</td>
<td></td>
</tr>
<tr>
<td>3. $X \leftarrow g^x$</td>
<td></td>
</tr>
<tr>
<td>4. $Y' \leftarrow B^{-1} g^r = g^{r-b}$</td>
<td></td>
</tr>
<tr>
<td>5. $k \leftarrow (BY')^x = g^{(b+r-b)x} = g^{rx}$</td>
<td></td>
</tr>
</tbody>
</table>

$$k = g^{rx}$$
Security goals
One-way authenticated key exchange

- Key agreement security models (BR93, CK01, eCK, ...) typically two-way (mutually) authenticated
One-way authenticated key exchange

- Key agreement security models (BR93, CK01, eCK, ... ) typically **two-way (mutually) authenticated**
- Many real-world protocols only **one-way authenticated**:
  - Tor; vast majority of TLS usage
One-way authenticated key exchange

- Key agreement security models (BR93, CK01, eCK, …) typically two-way (mutually) authenticated
- Many real-world protocols only one-way authenticated:
  - Tor; vast majority of TLS usage

One-way ≠ one-flow:

- **One-flow** AKE establishes a session key with a single message from the client to the server.
- **One-way** AKE gives server-to-client authentication but not client-to-server authentication
One-way authenticated key exchange

- Key agreement security models (BR93, CK01, eCK, ...) typically two-way (mutually) authenticated
- Many real-world protocols only one-way authenticated:
  - Tor; vast majority of TLS usage

One-way ≠ one-flow:

- One-flow AKE establishes a session key with a single message from the client to the server.
- One-way AKE gives server-to-client authentication but not client-to-server authentication

One-way AKE as either:

- Restriction of standard two-way AKE to one-way setting
- Extension of public-key encryption to include forward secrecy
Secrecy without authentication?

What motivation does a party not receiving authentication promises have for using secrecy?

- A server provides the same level/type of service to each unauthenticated client:
  - Medical advice to anonymous patients the same whether request came encrypted or not.
  - Search engine responses the same whether request came over HTTP or HTTPS.

But...

- Doctors required to preserve patient–doctor confidentiality even with unauthenticated patients.
- ISPs may eavesdrop on search engine queries/responses for marketing purposes.
Secrecy without authentication?

What motivation does a party not receiving authentication promises have for using secrecy?

- A server provides the same level/type of service to each unauthenticated client:
  - Medical advice to anonymous patients the same whether request came encrypted or not.
  - Search engine responses the same whether request came over HTTP or HTTPS.
Secrecy without authentication?

What motivation does a party not receiving authentication promises have for using secrecy?

- A server provides the same level/type of service to each unauthenticated client:
  - Medical advice to anonymous patients the same whether request came encrypted or not.
  - Search engine responses the same whether request came over HTTP or HTTPS.

secrecy ≤ authentication
Secrecy without authentication?

What motivation does a party not receiving authentication promises have for using secrecy?

- A server provides the same level/type of service to each unauthenticated client:
  - Medical advice to anonymous patients the same whether request came encrypted or not.
  - Search engine responses the same whether request came over HTTP or HTTPS.

  $\text{secrecy} \leq \text{authentication}$

But...

- Doctors required to preserve patient–doctor confidentiality even with unauthenticated patients $\implies \text{exclusivity}$.
- ISPs may eavesdrop on search engine queries/responses for marketing purposes.
Anonymity properties

- **Anonymity**: party is not identifiable (within a set of parties)

---

Anonymity properties

- **Anonymity**: party is not identifiable (within a set of parties)
- **Unlinkability**: cannot determine if two items of interest (e.g., sessions) are related
Anonymity properties

- **Anonymity**: party is not identifiable (within a set of parties)
- **Unlinkability**: cannot determine if two items of interest (e.g., sessions) are related
- **Undetectability**: cannot determine if something exists or not

---

Anonymity properties

- **Anonymity**: party is not identifiable (within a set of parties)
- **Unlinkability**: cannot determine if two items of interest (e.g., sessions) are related
- **Undetectability**: cannot determine if something exists or not

Related properties:

- **Identity hiding**: identity of a party never communicated in the clear but eventually made known to peer

---

Anonymity properties

- **Anonymity**: party is not identifiable (within a set of parties)
- **Unlinkability**: cannot determine if two items of interest (e.g., sessions) are related
- **Undetectability**: cannot determine if something exists or not

Related properties:

- **Identity hiding**: identity of a party never communicated in the clear but eventually made known to peer
- **Deniability**: identity of a party not necessarily kept secret, but party’s participation in a session cannot be conclusively proven

---

Security model
Session execution

- Parties have long-term (static) and session-specific (ephemeral) key pairs and certificates associated to long-term keys
- Parties assign a locally unique session identifier $\Psi$ to each session
- Parties output a tuple $(sk, pid, \vec{v})$ for each session, where
  - $sk$ is a session key
  - $pid$ is a party identifier or the anonymous symbol $\star$
  - $\vec{v} = (\vec{v}_1, \vec{v}_2, \ldots)$ is a vector of vectors of public values
Adversary powers

- Send$^P(params, pid) \rightarrow (\Psi, msg)$:
  Activate party $P$ to start a new key exchange session.
Adversary powers

- Send\textsuperscript{P}(params, pid) \rightarrow (\Psi, msg):
  Activate party P to start a new key exchange session.
- Send\textsuperscript{P}(\Psi, msg) \rightarrow msg':
  Send a message to party P.
Security model

Adversary powers

- **Send**\(^P\)(params, pid) → (Ψ, msg):
  Activate party \(P\) to start a new key exchange session.

- **Send**\(^P\)(Ψ, msg) → msg′:
  Send a message to party \(P\).

- **RevealNext**\(^P\) → \(X\):
  Learn the next public key value \(X\) that will be used by \(P\).
Adversary powers

- **Send**\(^P\)(\textit{params, pid}) \rightarrow (\Psi, msg):
  Activate party \(P\) to start a new key exchange session.

- **Send**\(^P\)(\Psi, msg) \rightarrow msg':
  Send a message to party \(P\).

- **RevealNext**\(^P\) \rightarrow X:
  Learn the next public key value \(X\) that will be used by \(P\).

- **Partner**\(^P\)(X) \rightarrow x:
  Learn the secret value \(x\) for party \(P\)'s key pair \((x, X)\).
Adversary powers

- **Send**\(^P\)(\(params, pid\) → (\(\Psi, msg\)):
  Activate party \(P\) to start a new key exchange session.

- **Send**\(^P\)(\(\Psi, msg\) → \(msg'\)):
  Send a message to party \(P\).

- **RevealNext**\(^P\) → \(X\):
  Learn the next public key value \(X\) that will be used by \(P\).

- **Partner**\(^P\)(\(X\)) → \(x\):
  Learn the secret value \(x\) for party \(P\)'s key pair \((x, X)\).

- **SessionKeyReveal**\(^P\)(\(\Psi\)) → \(sk\)
Adversary powers

- \(\text{Send}^P(\text{params}, \text{pid}) \rightarrow (\Psi, \text{msg})\): Activate party \(P\) to start a new key exchange session.
- \(\text{Send}^P(\Psi, \text{msg}) \rightarrow \text{msg}'\): Send a message to party \(P\).
- \(\text{RevealNext}^P \rightarrow X\): Learn the next public key value \(X\) that will be used by \(P\).
- \(\text{Partner}^P(X) \rightarrow x\): Learn the secret value \(x\) for party \(P\)'s key pair \((x, X)\).
- \(\text{SessionKeyReveal}^P(\Psi) \rightarrow sk\)
- \(\text{EstablishCertificate}\)
One-way AKE security

- Test\((P, \Psi) \rightarrow sk:\)
  1. Stop if \(\Psi.sk = \bot\) or \(\Psi.pid = \odot\).
  2. Choose \(b \leftarrow \{0, 1\}\)
  3. If \(b = 1\): return \(\Psi.sk\)
  4. If \(b = 0\): return random key of same length
One-way AKE security

- **Test**($P, \Psi$) $\rightarrow sk$:
  1. Stop if $\Psi.sk = \bot$ or $\Psi.pid = \oplus$.
  2. Choose $b \leftarrow \{0, 1\}$
  3. If $b = 1$: return $\Psi.sk$
  4. If $b = 0$: return random key of same length

- **$\Psi$ is one-way-AKE-fresh** if both:
  1. for every $\vec{v}_j$ in $\Psi.\vec{v}$, there is at least one element $X \in \vec{v}_j$ where adversary is not a partner to $X$
  2. no SessionKeyReveal$^P(\Psi')$ at $P = \Psi.pid$ where $\Psi'.\vec{v} = \Psi.\vec{v}$
One-way AKE security

- Test$(P, \Psi) \rightarrow sk$:
  1. Stop if $\Psi.sk = \perp$ or $\Psi.pid = \odot$.
  2. Choose $b \leftarrow \{0, 1\}$
  3. If $b = 1$: return $\Psi.sk$
  4. If $b = 0$: return random key of same length

- $\Psi$ is one-way-AKE-fresh if both:
  1. for every $\vec{v}_j$ in $\Psi.\vec{v}$, there is at least one element $X \in \vec{v}_j$ where adversary is not a partner to $X$
  2. no SessionKeyReveal$^P(\Psi')$ at $P = \Psi.pid$ where $\Psi'.\vec{v} = \Psi.\vec{v}$

- A protocol is one-way-AKE-secure if for all p.p.t. $M$ the advantage that $M$ guesses $b$ in a fresh session is negligible.
One-way AKE security

- Test($P, \Psi$) $\rightarrow sk$:
  1. Stop if $\Psi.sk = \bot$ or $\Psi.pid = \star$.
  2. Choose $b \leftarrow \{0, 1\}$
  3. If $b = 1$: return $\Psi.sk$
  4. If $b = 0$: return random key of same length

- $\Psi$ is **one-way-AKE-fresh** if both:
  1. for every $\vec{v}_j$ in $\Psi.\vec{v}$, there is at least one element $X \in \vec{v}_j$ where adversary is not a partner to $X$
  2. no SessionKeyReveal$^P(\Psi')$ at $P = \Psi.pid$ where $\Psi'.\vec{v} = \Psi.\vec{v}$

- A protocol is **one-way-AKE-secure** if for all p.p.t. $M$ the advantage that $M$ guesses $b$ in a fresh session is negligible.

- Forward secrecy?
One-way anonymity

Guess which of two parties is participating in the key exchange.

\[ A \]

Start(\(i, j, \text{params, pid}\))

\[ \text{Challenger } C \]

\(i^* \leftarrow \{i, j\}\)

Send(\(\text{params, pid}\))

\[ \text{Party } P_{i^*} \]

Send

\(\text{Send}\)

guess \(i^*\)
One-way anonymity

Guess which of two parties is participating in the key exchange.

Goal: Guess $i^*$ with non-negligible advantage.
One-way anonymity

Guess which of two parties is participating in the key exchange.

- **Goal:** Guess \( i^* \) with non-negligible advantage.
- Can issue RevealNext, Partner, and SessionKeyReveal to challenger
- Can’t issue queries related to challenge session to original parties
Unlinkability

Determine whether two items of interest are related or not.

\[ \text{Challenger } C \]
\[ b^* \leftarrow \{0, 1\} \]

\[ \text{Start}(i, j, \text{params}_1, \text{params}_2, \text{pid}) \]
\[ \text{Send}(\text{params}_1, \text{pid}) \]
\[ \text{Send}(\text{params}_2, \text{pid}) \]

\[ \text{if } b^* = 0 \]
\[ \text{Send}(\text{params}_2, \text{pid}) \]
\[ \text{Send}(\text{params}_1, \text{pid}) \]

\[ \text{if } b^* = 1 \]
\[ \text{Send}(\text{params}_1, \text{pid}) \]
\[ \text{Send}(\text{params}_2, \text{pid}) \]
Unlinkability

Determine whether two items of interest are related or not.

Goal: Guess $b^*$ with non-negligible advantage.
Unlinkability

Determine whether two items of interest are related or not.

- **Goal:** Guess $b^*$ with non-negligible advantage.
- Can issue RevealNext1, RevealNext2, Partner1, Partner2, SessionKeyReveal1, and SessionKeyReveal2 to challenger
- Can’t issue queries related to challenge session to original parties
Security model

One-way anonymity

Unlinkability
Security model

One-way anonymity = Unlinkability

A
Start(i, j, params, pid)
Send
guess i*

Challenger C
i* \in \{i, j\}
Send(params, pid)

Party P_i

A
Start(i, j, params1, params2, pid)
Send
Send1
Send2

guess b*

Challenger C
b* \in \{0, 1\}

if b* = 0

if b* = 1

Party P_i

Send
Party P_j

Send

Send

Send

Send
Equivalence of anonymity and unlinkability

One-way anonymity $\iff$ unlinkability:

- Adversary starts unlinkability game with parties $P_i$ and $P_j$.
Equivalence of anonymity and unlinkability

One-way anonymity $\iff$ unlinkability:

- Adversary starts unlinkability game with parties $P_i$ and $P_j$
- Simulator creates two sessions using anonymity challenger:
  1. One session with $P_i$
  2. One session with anonymity challenger for $P_i$ and $P_j$
Equivalence of anonymity and unlinkability

One-way anonymity $\iff$ unlinkability:

- Adversary starts unlinkability game with parties $P_i$ and $P_j$
- Simulator creates two sessions using anonymity challenger:
  1. One session with $P_i$
  2. One session with anonymity challenger for $P_i$ and $P_j$
- If anonymity challenger uses $P_i$: unlinkability simulator uses $P_i$ and $P_i$
- If anonymity challenger uses $P_j$: unlinkability simulator uses $P_i$ and $P_j$
Equivalence of anonymity and unlinkability

One-way anonymity $\iff$ unlinkability:

- Adversary starts unlinkability game with parties $P_i$ and $P_j$
- Simulator creates two sessions using anonymity challenger:
  1. One session with $P_i$
  2. One session with anonymity challenger for $P_i$ and $P_j$
- If anonymity challenger uses $P_i$: unlinkability simulator uses $P_i$ and $P_i$
- If anonymity challenger uses $P_j$: unlinkability simulator uses $P_i$ and $P_j$
- Unlinkability adversary guesses $b$

  $\implies$ one-way anonymity simulator guesses $\begin{cases}     i, & \text{if } b = 0 \\     j, & \text{if } b = 1 \end{cases}$
Equivalence of anonymity and unlinkability

Unlinkability $\iff$ one-way anonymity:

- Adversary starts one-way anonymity game with parties $P_i$ and $P_j$
- Simulator uses unlinkability challenger for $P_i$ and $P_j$:
  1. Adversary’s queries are relayed to unlinkability challenger’s second party
- If unlinkability challenger uses $P_i$: anonymity simulator uses $P_i$
- If unlinkability challenger uses $P_j$: anonymity simulator uses $P_j$
- Anonymity adversary guesses $i'$
  $\implies$ unlinkability simulator guesses $\begin{cases} 1, & \text{if } i' = i \\ 0, & \text{if } i' = j \end{cases}$
Protocols
One-way-authenticated TLS

Session key security

- Mutually authenticated:
  - Jonsson and Kaliski (CRYPTO 2002): RSA encryption security
  - Morrissey, Smart, Warinschi (ASIACRYPT 2008): truncated TLS
  - Gajek et al. (ProvSec 2008): UC security of TLS_DHE
  - Jager et al. (CRYPTO 2012): mutual ACCE security of TLS_DHE
One-way-authenticated TLS

Session key security

- Mutually authenticated:
  - Jonsson and Kaliski (CRYPTO 2002): RSA encryption security
  - Morrissey, Smart, Warinschi (ASIACRYPT 2008): truncated TLS
  - Gajek et al. (ProvSec 2008): UC security of TLS_DHE
  - Jager et al. (CRYPTO 2012): mutual ACCE security of TLS_DHE

- One-way authenticated:
  - Morrissey, Smart, Warinschi (ASIACRYPT 2008): truncated TLS
  - Gajek et al. (ProvSec 2008): UC security of TLS_DHE
  - TLS_RSA and TLS_DHE could be proven secure in our model, although neither with forward secrecy
One-way-authenticated TLS

Anonymity

Lots of values in TLS could leak identifying information:

- **ClientHello**: supported TLS versions, cipher suites, algorithms, extensions
- **ClientHello.client_random.gmt_unix_time**: current time in seconds
- **ServerHello.session_id**: many clients abort if they receive a session identifier that already exists in its cache
**Proposed protocol:** ntor

<table>
<thead>
<tr>
<th>Client $\hat{A}$</th>
<th>Server $\hat{B}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>long-term private key $b$, public key $B = g^b$</td>
<td></td>
</tr>
</tbody>
</table>

1. $x \leftarrow Z_q$
2. $X \leftarrow g^x$
3. $\Psi_a \leftarrow H_{sid}(X)$

1. $y \leftarrow Z_q$
2. $Y \leftarrow g^y$
3. $\Psi_b \leftarrow H_{sid}(Y)$
4. $(sk', sk) \leftarrow H(X^y, X^b, \hat{B}, X, Y)$
5. $t_b \leftarrow H_{mac}(sk', \hat{B}, Y, X)$
6. $(sk', sk) \leftarrow H(Y^x, B^x, \hat{B}, X, Y)$
7. verify $t_b$
8. output $(sk, \hat{B}, \vec{v} = (X, (Y, B)))$
9. output $(sk, \bigstar, \vec{v} = (X, (Y, B)))$
Analysis of ntor

► **One-way AKE security**: If $H$ and $H_{mac}$ are random oracles and $H_{sid}$ is collision-resistant, and the gap Diffie–Hellman assumption holds.

► **One-way anonymity**: Unconditionally.

---

## Analysis of `ntor`

- **One-way AKE security**: If $H$ and $H_{mac}$ are random oracles and $H_{sid}$ is collision-resistant, and the gap Diffie–Hellman assumption holds.

- **One-way anonymity**: Unconditionally.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Efficiency (client)</th>
<th>Efficiency (server)</th>
<th>authentication</th>
<th>security</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Off-line</td>
<td>On-line</td>
<td>Off-line</td>
<td>On-line</td>
</tr>
<tr>
<td>DH</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Signed-DH</td>
<td>1</td>
<td>1+sigver</td>
<td>1</td>
<td>1+sign</td>
</tr>
<tr>
<td>ØS</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MQV</td>
<td>1</td>
<td>1.17 (1.5)</td>
<td>1</td>
<td>1.17 (1.5)</td>
</tr>
<tr>
<td>UM</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td><code>ntor</code></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1.33</td>
</tr>
<tr>
<td>Ace(^4)</td>
<td>2</td>
<td>1.08 (1.17)</td>
<td>1</td>
<td>1.08 (1.17)</td>
</tr>
</tbody>
</table>

Conclusions
Summary

- Insecurity of previously proposed protocol of Øverlier and Syverson
- **Security definitions** for
  - one-way AKE
  - anonymity
  - unlinkability
- **Equivalence** of anonymity and unlinkability
- **New protocol** ntor with security arguments
Open questions

- Most appropriate protocol for deployment?
- Impact of weak randomness on anonymity?
- Equivalence or inequivalence of anonymity and unlinkability in other settings?
- Pseudonymity in AKE: is it just mutual AKE with throw-away credentials?
- One-way AKE as public-key encryption with forward secrecy?