Specification and verification of probabilistic security protocols

Konstantinos Chatzikokolakis
under the direction of Catuscia Palamidessi

Summary

- Background
  - Cryptographic security protocols
  - Expressing a property in spi-calculus
- π_{prob}-calculus
- Presentation of protocols
  - Encoding in π_{prob}
- Verification of PSE
  - Specification in π_{prob}
  - Definition of necessary relations
  - Expression of the desired property
  - Proof of correctness
- Comparison with alternative methods

Cryptographic protocols

- Use of cryptographic primitives
- $\{M\}_K$: Message M encrypted by key K
- Example

$$
\begin{aligned}
A \rightarrow S: &\quad \{K_{AB}\}_{K_{AS}} \\
S \rightarrow B: &\quad \{K_{AB}\}_{K_{SB}} \\
A \rightarrow B: &\quad \{M\}_{K_{AB}}
\end{aligned}
$$

Expressing a property in the spi-calculus

Expressing a property in the spi-calculus

- π-calculus + encryption + decryption

Expressing a property in the spi-calculus

- π-calculus + encryption + decryption
- Example :

$$
\begin{aligned}
B &= c_{SB}(\{y\}_{K_{SB}}).c_{AB}(\{w\}_y).F(w) \\
Inst(M) &= A(M) \mid S \mid B
\end{aligned}
$$

Expressing a property in the spi-calculus

- π-calculus + encryption + decryption
- Example :

$$\begin{array}{rcl} B & = & c_{SB}(\{y\}_{K_{SB}}).c_{AB}(\{w\}_y).F(w) \\ Inst(M) & = & A(M) \mid S \mid B \end{array}$$

- Specification :

$$\begin{array}{rcl} B_{spec}(M) & = & c_{SB}(\{y\}_{K_{SB}}).c_{AB}(\{w\}_y).F(M) \\ Inst_{spec}(M) & = & A(M) \mid S \mid B_{spec}(M) \end{array}$$

Expressing a property in the spi-calculus

- π-calculus + encryption + decryption
- Example :

$$\begin{aligned} B &= c_{SB}(\{y\}_{K_{SB}}).c_{AB}(\{w\}_y).F(w) \\ Inst(M) &= A(M) \mid S \mid B \end{aligned}$$

- Specification :

$$\begin{aligned} B_{spec}(M) &= c_{SB}(\{y\}_{K_{SB}}).c_{AB}(\{w\}_y).F(M) \\ Inst_{spec}(M) &= A(M) \mid S \mid B_{spec}(M) \end{aligned}$$

- Authenticity : $Inst(M) \simeq Inst_{spec}(M)$, for all M

Probabilistic protocols

Probabilistic protocols

- Use of random choices (instead of cryptography)

Probabilistic protocols

- Use of random choices (instead of cryptography)
- Many applications :
  - contract signing
  - sending certified emails
  - protecting the anonymity of communication agents

Probabilistic protocols

- Use of random choices (instead of cryptography)
- Many applications :
  - contract signing
  - sending certified emails
  - protecting the anonymity of communication agents
- Need for new techniques for their analysis

π_{prob} : a probabilistc variant of pi-calculus

π_{prob} : a probabilistc variant of pi-calculus

- Based on probabilistic asynchronous pi-calculus

π_{prob} : a probabilistc variant of pi-calculus

- Based on probabilistic asynchronous pi-calculus
- Dinstinction between probabilistic and non-deterministic behavior

π_{prob} : a probabilistc variant of pi-calculus

- Based on probabilistic asynchronous pi-calculus
- Dinstinction between probabilistic and non-deterministic behavior
- Choices are always blind

π_{prob} : a probabilistc variant of pi-calculus

- Based on probabilistic asynchronous pi-calculus
- Dinstinction between probabilistic and non-deterministic behavior
- Choices are always blind
- Syntax :

$$P ::= \ldots \mid \textstyle\sum_i p_i P_i \mid [T \text{ is } T]P \mid \text{let } \langle x, y \rangle = T \text{ in } P$$

Semantics of π_{prob}-calculus

Semantics of π_{prob}-calculus

- Probabilistic automata

Semantics of π_{prob}-calculus

- Probabilistic automata
- Simplified in the absense of guarded choice

Semantics of π_{prob}-calculus

- Probabilistic automata
- Simplified in the absense of guarded choice
- Semantics :

$$\textstyle\sum_i p_i P_i \ \{\xrightarrow[p_i]{\tau} P_i\}_i$$

$$\overline{m}M.P \ \{\xrightarrow[1]{\overline{m}M} P\} \qquad\qquad m(x).P \ \{\xrightarrow[1]{m(x)} P\}$$

$$\frac{P \ \{\xrightarrow[1]{\overline{m}M} P'\} \qquad Q \ \{\xrightarrow[1]{m(x)} Q'\}}{P \mid Q \ \{\xrightarrow[1]{\tau} P' \mid Q'[M/x]\}}$$

1-out-of-2 Oblivious Transfer

1-out-of-2 Oblivious Transfer

- Primitive operation used in many protocols

1-out-of-2 Oblivious Transfer

- Primitive operation used in many protocols
- A sends two messages M_1, M_2

1-out-of-2 Oblivious Transfer

- Primitive operation used in many protocols
- A sends two messages M_1, M_2
- B receives $\langle i, M_i \rangle, i \in \{1, 2\}$, each with probability $1/2$

1-out-of-2 Oblivious Transfer

- Primitive operation used in many protocols
- A sends two messages M_1, M_2
- B receives $\langle i, M_i \rangle, i \in \{1, 2\}$, each with probability 1/2
- For A the probability (after the execution) that B received M_i remains 1/2

1-out-of-2 Oblivious Transfer

- Primitive operation used in many protocols
- A sends two messages M_1, M_2
- B receives $\langle i, M_i \rangle, i \in \{1, 2\}$, each with probability 1/2
- For A the probability (after the execution) that B received M_i remains 1/2
- Encoding in π_{prob} :

$$\begin{aligned}
\mathrm{OT}^{1}_{2}(m_1, m_2, c_{as}) &\triangleq \overline{c_{as}} m_1 . \overline{c_{as}} m_2 . 0 \\
S(c_{as}, c_{sb}) &\triangleq c_{as}(m_1) . c_{as}(m_2) . (\overline{c_{bs}} \langle 1, m_1 \rangle +_{0.5} \overline{c_{bs}} \langle 2, m_2 \rangle)
\end{aligned}$$

Partial Secrets Exchange

Partial Secrets Exchange

- Used as the core of other protocols

Partial Secrets Exchange

- Used as the core of other protocols
- Two agents. Each has n pairs of secrets

Partial Secrets Exchange

- Used as the core of other protocols
- Two agents. Each has n pairs of secrets
- Goal : exchange one pair of secrets

Partial Secrets Exchange

- Used as the core of other protocols
- Two agents. Each has n pairs of secrets
- Goal : exchange one pair of secrets
- If B receives a pair of A then A should receive a pair of B with high probability

Partial Secrets Exchange (2)

```
PSE (A, B, {a_i}_i, {b_i}_i) {

}
```

Partial Secrets Exchange (2)

```
PSE (A, B, {a_i}_i, {b_i}_i) {
    for i = 1 to n do
        OT^1_2(A, B, a_i, a_{i+n})
        OT^1_2(B, A, b_i, b_{i+n})
    next
}
```

Partial Secrets Exchange (2)

```
PSE (A, B, {a_i}_i, {b_i}_i) {
    for i = 1 to n do
        OT^1_2(A, B, a_i, a_{i+n})
        OT^1_2(B, A, b_i, b_{i+n})
    next
    for j = 1 to m do
        for i = 1 to 2n do
            A sends jth bit of a_i to B
        for i = 1 to 2n do
            B sends jth bit of b_i to B
    next
}
```

Encoding in π_{prob}-calculus

Encoding in π_{prob}-calculus

$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}_{\frac{1}{2}}(\langle a_{i1}, \ldots, a_{ij}\rangle, \langle a_{(i+n)1}, \ldots, a_{(i+n)j}\rangle, c_{as_i}) \mid$$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}\tfrac{1}{2}(\langle a_{i1}, \ldots, a_{ij}\rangle, \langle a_{(i+n)1}, \ldots, a_{(i+n)j}\rangle, c_{as_i}) \mid$$

$$c_{sa_1}(\langle k_1, r_1\rangle).\mathrm{let}\ \langle r_{11}, \ldots, r_{1j}\rangle = r_1 \ \mathrm{in}\ \ldots c_{sa_n}(\langle k_n, r_n\rangle).\mathrm{let}\ \langle \ldots \rangle = r_n \ \mathrm{in}$$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}_{\frac{1}{2}}(\langle a_{i1},\ldots,a_{ij}\rangle,\langle a_{(i+n)1},\ldots,a_{(i+n)j}\rangle, c_{as_i}) \mid$$

$$c_{sa_1}(\langle k_1, r_1\rangle).\mathrm{let}\ \langle r_{11},\ldots,r_{1j}\rangle = r_1\ \mathrm{in}\ \ldots c_{sa_n}(\langle k_n, r_n\rangle).\mathrm{let}\ \langle\ldots\rangle = r_n\ \mathrm{in}$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1}\star \mid$$

$$\prod_{j=1}^{m} s_j(x).$$

$$s_{m+1}(x).\overline{c_p}ok)$$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}_{\frac{1}{2}}(\langle a_{i1},\ldots,a_{ij}\rangle,\langle a_{(i+n)1},\ldots,a_{(i+n)j}\rangle,c_{as_i}) \mid$$

$$c_{sa_1}(\langle k_1,r_1\rangle).\mathrm{let}\ \langle r_{11},\ldots,r_{1j}\rangle = r_1\ \mathrm{in}\ \ldots c_{sa_n}(\langle k_n,r_n\rangle).\mathrm{let}\ \langle\ldots\rangle = r_n\ \mathrm{in}$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1}\star \mid$$

$$\prod_{j=1}^{m} s_j(x).\ \overline{c_p}a_{1j}.\ \ldots\overline{c_p}a_{(2n)j}.c_p(d_{1j}).\ \ldots c_p(d_{(2n)j}).$$

$$s_{m+1}(x).\overline{c_p}ok)$$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n, j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}_{\frac{1}{2}}(\langle a_{i1}, \ldots, a_{ij} \rangle, \langle a_{(i+n)1}, \ldots, a_{(i+n)j} \rangle, c_{as_i}) \mid$$

$$c_{sa_1}(\langle k_1, r_1 \rangle).\mathrm{let}\ \langle r_{11}, \ldots, r_{1j} \rangle = r_1\ \mathrm{in}\ \ldots c_{sa_n}(\langle k_n, r_n \rangle).\mathrm{let}\ \langle \ldots \rangle = r_n\ \mathrm{in}$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1} \star \mid$$

$$\prod_{j=1}^{m} s_j(x).\ \overline{c_p} a_{1j}.\ \ldots \overline{c_p} a_{(2n)j}.c_p(d_{1j}).\ \ldots c_p(d_{(2n)j}).$$

$$\nu t_1 \ldots \nu t_{n+1}(\overline{t_1} \star \mid \prod_{i=1}^{n} t_i(x).Test(i,j) \mid t_{n+1}(x).\overline{s_{j+1}} \star) \mid$$

$$s_{m+1}(x).\overline{c_p} ok)$$

Encoding in π_{prob}-calculus

$$A(\{a_{ij}\}_{i=1..2n,j=1..m}) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}_{\frac{1}{2}}(\langle a_{i1},\ldots,a_{ij}\rangle, \langle a_{(i+n)1},\ldots,a_{(i+n)j}\rangle, c_{as_i}) \mid$$

$$c_{sa_1}(\langle k_1, r_1\rangle).\mathrm{let}\ \langle r_{11},\ldots,r_{1j}\rangle = r_1\ \mathrm{in}\ \ldots c_{sa_n}(\langle k_n, r_n\rangle).\mathrm{let}\ \langle\ldots\rangle = r_n\ \mathrm{in}$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1}\star \mid$$

$$\prod_{j=1}^{m} s_j(x).\ \overline{c_p}a_{1j}.\ \ldots \overline{c_p}a_{(2n)j}.c_p(d_{1j}).\ \ldots c_p(d_{(2n)j}).$$

$$\nu t_1 \ldots \nu t_{n+1}(\overline{t_1}\star \mid \prod_{i=1}^{n} t_i(x).Test(i,j) \mid t_{n+1}(x).\overline{s_{j+1}}\star) \mid$$

$$s_{m+1}(x).\overline{c_p}ok)$$

$$Test(i,j) \triangleq ([k_i\ \mathrm{is}\ 1][r_{ij}\ \mathrm{is}\ d_{ij}]\overline{t_{i+1}}\star \mid [k_i\ \mathrm{is}\ 2][r_{ij}\ \mathrm{is}\ d_{(i+n)j}]\overline{t_{i+1}}\star)$$

Verification of PSE

Verification of PSE

- Property to verify :

  If B receives one pair of A then with high probability A should be able to receive one of B's pairs.

Verification of PSE

- Property to verify :

  If B receives one pair of A then with high probability A should be able to receive one of B's pairs.

- Differences with respect to cryptographic protocols
  - The adversary is B himself
  - A behaves differently when B tries to cheat

Verification of PSE

- Property to verify :

  If B receives one pair of A then with high probability A should be able to receive one of B's pairs.

- Differences with respect to cryptographic protocols
  - The adversary is B himself
  - A behaves differently when B tries to cheat
- The specification of A depends on the behavior of B

Verification of PSE

- Property to verify :

  If B receives one pair of A then with high probability A should be able to receive one of B's pairs.

- Differences with respect to cryptographic protocols
  - The adversary is B himself
  - A behaves differently when B tries to cheat

- The specification of A depends on the behavior of B

- Express B's "intention to cheat" by a function h :

$$h(i) = j_i \qquad \text{if } B \text{ is going to cheat on the } j_i\text{-th bit of } i\text{-th pair}$$

Specification of PSE

Specification of PSE

$$A_{spec}(\{a_{ij}\}_{i=1..2n,j=1..m}, h) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}\,^1_2(\langle a_{i1}, \ldots, a_{ij}\rangle, \langle a_{(i+n)1}, \ldots, a_{(i+n)j}\rangle, c_{as_i}) \mid$$

$$c_{sa_1}(x) \ldots c_{sa_n}(x).$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1}\star \mid$$

$$\prod_{j=1}^{m} s_j(x).\overline{c_p}a_{1j}. \ \ldots \overline{c_p}a_{(2n)j}.c_p(x). \ \ldots c_p(x).$$

$$\nu t_1 \ldots \nu t_{n+1}(\overline{t_1}\star \mid \prod_{i=1}^{n} t_i(x).Test_{spec}(i,j,h) \mid t_{n+1}(x).\overline{s_{j+1}}\star) \mid$$

$$s_{m+1}(x).\overline{c_p}ok)$$

Specification of PSE

$$A_{spec}(\{a_{ij}\}_{i=1..2n,j=1..m}, h) \triangleq$$

$$\prod_{i=1}^{n} \mathrm{OT}^{\,1}_{\,2}(\langle a_{i1}, \ldots, a_{ij}\rangle, \langle a_{(i+n)1}, \ldots, a_{(i+n)j}\rangle, c_{as_i}) \mid$$

$$c_{sa_1}(x) \ldots c_{sa_n}(x).$$

$$\nu s_1 \ldots \nu s_{m+1}(\overline{s_1}\star \mid$$

$$\prod_{j=1}^{m} s_j(x).\overline{c_p}a_{1j}. \ \ldots \overline{c_p}a_{(2n)j}.c_p(x). \ \ldots c_p(x).$$

$$\nu t_1 \ldots \nu t_{n+1}(\overline{t_1}\star \mid \prod_{i=1}^{n} t_i(x).Test_{spec}(i,j,h) \mid t_{n+1}(x).\overline{s_{j+1}}\star) \mid$$

$$s_{m+1}(x).\overline{c_p}ok)$$

$$Test_{spec}(i,j,h) \triangleq \begin{cases} \overline{t_{i+1}}\star +_{0.5} 0 & \text{if } h(i) = j \\ \overline{t_{i+1}}\star & \text{otherwise} \end{cases}$$

Correctness of the specification

Correctness of the specification

- Suppose B is willing to reveal at most l bits

Correctness of the specification

- Suppose B is willing to reveal at most l bits
- B's behavior will be modeled by h st : $h(i) \leq l + 1$ for all $i \in \{1..n\}$

Correctness of the specification

- Suppose B is willing to reveal at most l bits
- B's behavior will be modeled by h st : $h(i) \leq l + 1$ for all $i \in \{1..n\}$
- The specification is deterministic, except from $Test_{spec}(i, j, h)$ for $h(i) = j$, which stalls with probability $1/2$

Correctness of the specification

- Suppose B is willing to reveal at most l bits
- B's behavior will be modeled by h st : $h(i) \leq l + 1$ for all $i \in \{1..n\}$
- The specification is deterministic, except from $Test_{spec}(i, j, h)$ for $h(i) = j$, which stalls with probability $1/2$
- There are n such tests in the first $l + 1$ steps

$$pb(\text{reveal } l + 2 \text{ pairs}) = 2^{-n}$$

Relations between processes

Relations between processes

- Testing equivalence
  - A test O is a process capable of performing an action ω
  - P passes O if $P \mid O$ can perform ω
  - Two processes are equivalent if they can pass the same tests

Relations between processes

- Testing equivalence
  - A test O is a process capable of performing an action ω
  - P passes O if $P \mid O$ can perform ω
  - Two processes are equivalent if they can pass the same tests
- Several ways of extending to probabilistic processes

Relations between processes

- Testing equivalence
  - A test O is a process capable of performing an action ω
  - P passes O if $P \mid O$ can perform ω
  - Two processes are equivalent if they can pass the same tests
- Several ways of extending to probabilistic processes
- We are not only interested in the ability to pass a test, but in the exact probability of passing it

Relations between processes

- Testing equivalence
  - A test O is a process capable of performing an action ω
  - P passes O if $P \mid O$ can perform ω
  - Two processes are equivalent if they can pass the same tests
- Several ways of extending to probabilistic processes
- We are not only interested in the ability to pass a test, but in the exact probability of passing it
- Define a preorder instead of an equivalence

Testing preorders

Testing preorders

- may-testing

Testing preorders

- may-testing
  - $pb(sexec(\nu(P \mid O), \zeta) =$ prob. of passing O under a scheduler ζ

Testing preorders

- may-testing
  - $pb(sexec(\nu(P \mid O), \zeta) =$ prob. of passing O under a scheduler ζ
  - $P\lceil O\rceil = \sup\{pb(sexec(\nu(P \mid O), \zeta)) \mid \zeta \text{ is a scheduler}\}$

Testing preorders

- may-testing
  - $pb(sexec(\nu(P \mid O), \zeta) =$ prob. of passing O under a scheduler ζ
  - $P\lceil O \rceil = \sup\{pb(sexec(\nu(P \mid O), \zeta)) \mid \zeta \text{ is a scheduler}\}$
  - $P \sqsubseteq Q$ iff for all tests $O : P\lceil O \rceil \leq Q\lceil O \rceil$

Testing preorders

- may-testing
  - $pb(sexec(\nu(P \mid O), \zeta) =$ prob. of passing O under a scheduler ζ
  - $P\lceil O\rceil = \sup\{pb(sexec(\nu(P \mid O), \zeta)) \mid \zeta \text{ is a scheduler}\}$
  - $P \sqsubseteq Q$ iff for all tests $O : P\lceil O\rceil \leq Q\lceil O\rceil$
- Conditional may-testing

$$P \sqsubseteq^{\phi} Q \quad \text{iff for all tests } O : \phi(O) \Rightarrow P\lceil O\rceil \leq Q\lceil O\rceil$$

Testing preorders

- may-testing
  - $pb(sexec(\nu(P \mid O), \zeta) =$ prob. of passing O under a scheduler ζ
  - $P\lceil O\rceil = \sup\{pb(sexec(\nu(P \mid O), \zeta)) \mid \zeta \text{ is a scheduler}\}$
  - $P \sqsubseteq Q$ iff for all tests $O : P\lceil O\rceil \leq Q\lceil O\rceil$
- Conditional may-testing

$$P \sqsubseteq^{\phi} Q \quad \text{iff for all tests } O : \phi(O) \Rightarrow P\lceil O\rceil \leq Q\lceil O\rceil$$

- $\sqsubseteq$ is a *precongruence*

Expressing the correctness of PSE

Expressing the correctness of PSE

- An instance of the protocol :

$$I \triangleq A(\{a_{ij}\}_{i=1..2n,j=1..m}, \{b_i\}_{i=1..2n}) \mid \prod_{i=1}^{n}(S(c_{as_i}, c_{sb_i}) \mid S(c_{bs_i}, c_{sa_i}))$$

Expressing the correctness of PSE

- An instance of the protocol :

$$I \triangleq A(\{a_{ij}\}_{i=1..2n,j=1..m}, \{b_i\}_{i=1..2n}) \mid \prod_{i=1}^{n}(S(c_{as_i}, c_{sb_i}) \mid S(c_{bs_i}, c_{sa_i}))$$

- PSE is correct if :

$$\forall h : I \sqsubseteq^{\phi_h} I_{spec}(h)$$

where $\phi_h(O) = true$ iff $\forall i \in \{1..n\}$:

O dont contain both $b_{ih(i)}$ and $b_{(i+n)h(i)}$

Proof of the correctness of PSE (sketch)

Proof of the correctness of PSE (sketch)

- I and I_{spec} differ only in the definition of $Test$.

Proof of the correctness of PSE (sketch)

- I and I_{spec} differ only in the definition of $Test$.
- For $j \neq h(i)$

$$Test(i,j) \sqsubseteq^{\phi_h} Test_{spec}(i,j,h)$$

  so we can replace the first by the second.

Proof of the correctness of PSE (sketch)

- I and I_{spec} differ only in the definition of $Test$.
- For $j \neq h(i)$

$$Test(i,j) \sqsubseteq^{\phi_h} Test_{spec}(i,j,h)$$

  so we can replace the first by the second.
- For $j = h(i)$
  - We define the relation $\sqsubseteq_p$ between pairs of processes :

$$(P_1, P_2) \sqsubseteq_p (Q_1, Q_2) \quad \text{iff} \quad P_1 +_p P_2 \sqsubseteq Q_1 +_p Q_2$$

  - We show that $\sqsubseteq_p$ is a precongruence
  - We construct an appropriate context

Comparison with alternative methods

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the "normal" execution of the protocol

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the “normal” execution of the protocol
  - B can only stop executing the protocol

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the "normal" execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the “normal” execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage
  - Advantage : automated method

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the “normal” execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage
  - Advantage : automated method
- Advantages of process-calculi approach

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the “normal” execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage
  - Advantage : automated method
- Advantages of process-calculi approach
  - No need to explicitly define the adversary

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the "normal" execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage
  - Advantage : automated method
- Advantages of process-calculi approach
  - No need to explicitly define the adversary
  - Possible analysis of an infinite number of sessions

Comparison with alternative methods

- Probabilistic model checking (Norman & Shmatikov)
  - Consider only the “normal” execution of the protocol
  - B can only stop executing the protocol
  - With probability 1 B can obtain a one-bit advantage
  - Advantage : automated method
- Advantages of process-calculi approach
  - No need to explicitly define the adversary
  - Possible analysis of an infinite number of sessions
  - Great number of theoretical tools from the literature