

Formal verification with the SVO logic

Security Protocols (bmevihim132)

Dr. Levente Buttyán
associate professor
BME Hálózati Rendszerek és Szolgáltatások Tanszék
Lab of Cryptography and System Security (CrySyS)
buttyan@hit.bme.hu, buttyan@crysys.hu

Motivation for formal methods

- many protocols have been proposed, but most of them are flawed
  - how can these flaws be explained?

- flaws are subtle and hard to find
  - how can flaws be uncovered?

- in order to avoid flaws, we need more understanding
  - what does the protocol really achieve?
  - does the protocol need more assumptions than another one?
  - does the protocol do anything unnecessary that could be left out without weakening it?
    - e.g., does the protocol encrypt something that could be sent in clear?

→ formal methods may help answering these questions

Many different approaches

- logics
  - general purpose: HOL/Isabelle
  - special: BAN (Burrows-Abadi-Needham), GNY (Gong-Needham-Yahalom), **SvO (Syverson-van Oorschot)**, …
- process algebrae
  - general purpose: CSP/FDR
  - special: spi-calculus, applied pi (ProVerif)
- strand spaces (Athena)
- model checking (Murphi, …)
- specialized expert systems (NRL protocol analyzer, Interrogator)

Basic idea of belief logics

- modal logic based tools allow us to describe the beliefs of parties in the protocol and to study the evolution of these beliefs as a consequence of communication

- examples:
  - if you have sent Bob a number that you have never used for this purpose before, and if you subsequently receive from Bob something that depends on knowing that number, then you should believe that Bob's message originated recently (at least after yours)
  - if you believe that only you and Bob know K, then you should believe that anything you receive protected with K comes originally from Bob
  - if you believe that K is Bob's public key, then you should believe that any signed message that you can successfully verify with K comes originally from Bob
  - …

SVO language

- set of primitive terms $\mathcal{T}$:
  - constant symbols representing principals (P, Q, …), keys (K_), numbers (N_), …
  - $\$_1$, $\$_2$, … – unrecognized received messages (or fragments)

- language of messages $\mathcal{M}_{\mathcal{T}}$:
  - any primitive term is a message
  - any logical formula is a message (see definition of the language of formulae $\mathcal{F}_{\mathcal{T}}$)
  - if X_1, X_2, …, X_n are messages and F is a function, then $F(X_1, X_2, \ldots, X_n)$ is a message
    - examples:
      - tuple: $(X_1, X_2, \ldots, X_n)$
      - encryption: $\{X\}_K$

SVO language

- language of formulae $\mathcal{F}_{\mathcal{T}}$:
  - if φ and ψ are formulae then $!\varphi$ and $(\varphi \,\&\, \psi)$ are formulae
  - if P and Q are principals and K and K^+ are a keys, then $P \leftarrow K \rightarrow Q$ and $K^+ \rightarrow P$ are formulae
    - **$P \leftarrow K \rightarrow Q$** means that K is a symmetric key shared by P and Q, and therefore, only P and Q (and parties they trust) know K, and only P and Q will ever use K for encryption (or MAC computation)
    - **$K^+ \rightarrow P$** means that K^+ is the public key of P, and therefore, only P knows the corresponding private key K^-, and only P can produce signed messages that verify correctly with K^+
    - $K^+ \rightarrow_{\varepsilon} P$, $K^+ \rightarrow_{\sigma} P$, $K^+ \rightarrow_{\gamma} P$ denote encryption, signature verification, and key agreement keys, respectively
  - SV(S, K^+, M) is a formula if S and M are messages and K^+ is a public key
    - **SV(S, K^+, M)** means that S is a correct signature on M produced with K^-
    - a similar predicate could be defined for MAC verification

SVO language

- P sees X, P received X, P says X, P said X, X is fresh are formulae if P is a principal and X is a message
  - **P sees X** means that P has (access to) X
  - **P received X** means that P received a message that contains X (of course, P received X implies P sees X, but the reverse is not true in general)
  - **P said X** means that P once sent a message that contained X
  - **X is fresh** means that X has never been sent before the current protocol run
  - **P says X** means that P said X in the current protocol run (i.e., P said X and X is fresh)

- P believes φ and P controls φ are formulae if P is a principal and φ is a formula
  - **P believes φ** means that P acts as if φ was true
  - **P controls φ** means that P is a trusted authority on φ (i.e., if P says φ is true, then φ is indeed true)

SVO axioms

- all tautologies of classical propositional logic
- believing:

  A1 (P believes φ) & (P believes ($\varphi \Rightarrow \psi$)) $\Rightarrow$ P believes ψ

  A2 P believes φ $\Rightarrow$ P believes (P believes φ)

- source association:

  A3 (P$\leftarrow$K$\rightarrow$Q) & (R received $\{X^Q\}_K$) $\Rightarrow$(Q said X) & (Q sees K)

  A4 (K$\rightarrow_\sigma$Q) & (R received S) & SV(S, K, X) $\Rightarrow$(Q said X)

- key agreement:

  A5 ($K_p \rightarrow_\gamma$ P) & ($K_q \rightarrow_\gamma$ Q) $\Rightarrow$(P$\leftarrow$KA(K_p,K_q)$\rightarrow$Q)

  A6 $\varphi \Leftrightarrow \varphi$[KA(K, K')/KA(K', K)]

  where KA is the key agreement function

SVO axioms

- receiving:

  A7 (P received $(X_1, \ldots, X_n)) \Rightarrow$(P received X_i)

  A8 (P received $\{X\}_K$) & (P sees K) $\Rightarrow$(P received X)

  A9 (P received $\{X\}_{K+}$) & (P sees K^-) $\Rightarrow$(P received X)

- seeing:

  A10 (P received X) $\Rightarrow$(P sees X)

  A11 (P sees $(X_1, \ldots, X_n)) \Rightarrow$(P sees X_i)

  A12 (P sees X_1) & … & (P sees X_n) $\Rightarrow$(P sees $F(X_1, \ldots, X_n)$)

- comprehending:

  A13 (P believes (P sees F(X))) $\Rightarrow$(P believes (P sees X))

  where F is a one-to-one function such that either F or F^{-1} is computable in practice by P

SVO axioms

- saying:
  A14 (P said (X_1, …, X_n)) $\Rightarrow$(P said X_i)
  A15 (P says (X_1, …, X_n)) $\Rightarrow$(P says X_i)
  A14’ (P said X) $\Rightarrow$(P sees X)
  A15’ (P says X) $\Rightarrow$(P said X)

- jurisdiction:
  A16 (P controls φ) & (P says φ) $\Rightarrow\varphi$

- freshness:
  A17 (X_i is fresh) $\Rightarrow$((X_1, …, X_n) is fresh)
  A18 ((X_1, …, X_n) is fresh) $\Rightarrow$(F(X_1, …, X_n) is fresh)

- nonce verification:
  A19 (X is fresh) & (P said X) $\Rightarrow$(P says X)

SVO inference rules

- Modus Ponens (MP):

$$\frac{\varphi \qquad \varphi \Rightarrow \psi}{\psi}$$

- Necessitation (N):

$$\frac{\sim \varphi}{\sim \text{P believes } \varphi}$$

where S ~ φ means that φ can be derived from the set of formulae S and the axioms, and ~ φ means that φ can be derived only from the axioms (i.e., φ is a theorem)

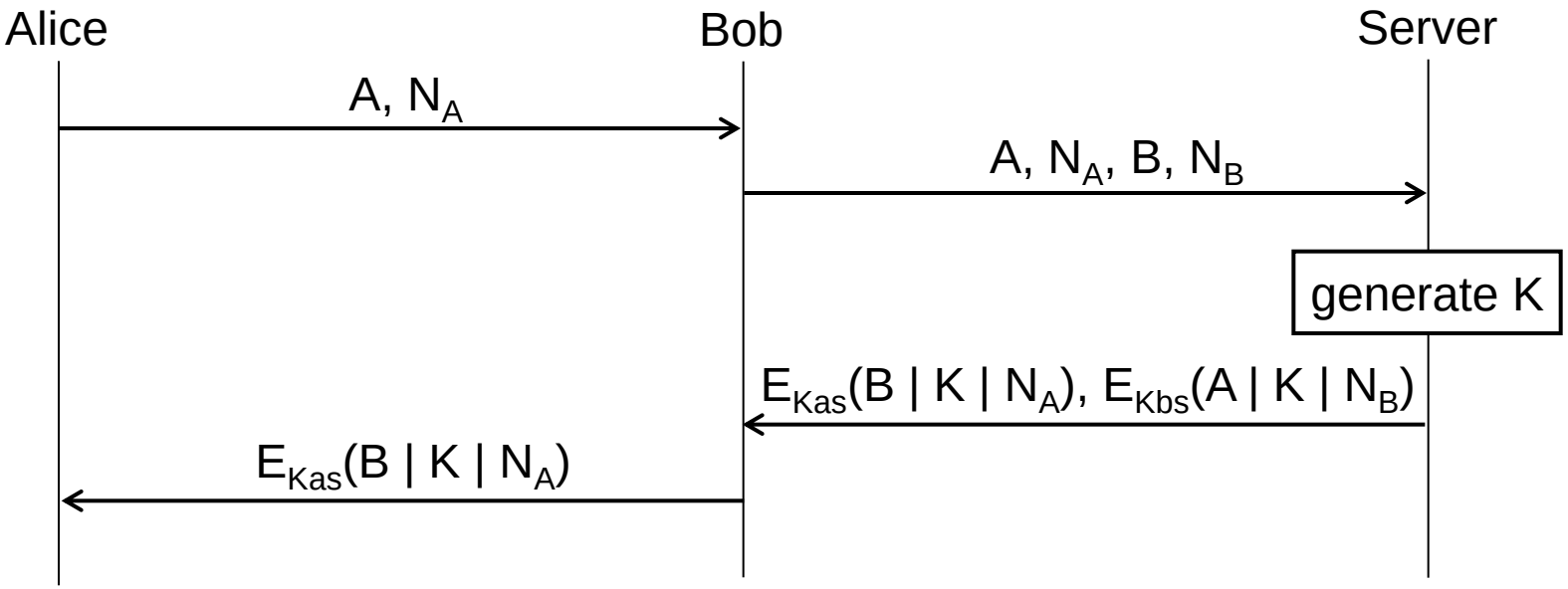
Summary of assumptions

- cryptography and communication
  - cryptographic primitives are secure and perfect (unbreakable)
- time
  - time is divided into past and present; the present epoch begins at the start of the particular run of the protocol under consideration
- belief
  - all beliefs held in the present are stable for the entirety of the protocol run
  - beliefs held in the past are not necessarily carried forward into the present
- honesty
  - protocol participants are honest principals
  - honest principals can indicate in encrypted messages the origin of the message ($\{X^Q\}_K$)
  - SVO may not detect attacks mounted by legitimate but malicious (dishonest) principals

Analysis methodology

- identify and formalize premises
  - initial assumptions on beliefs in freshness of nonces, goodness of long term keys, trust in servers, …
  - reception of protocol messages (e.g., A received $\{T_s, B, K_{ab}\}_{Kas}$)
  - premises reflecting what is comprehended from received messages (e.g., A believes (A received $\{T_s, B, \$\}_{Kas}$))
  - premises reflecting the interpretation of received messages (e.g., A believes (A received $\{T_s, B, \$\}_{Kas}$ $\Rightarrow$ A received $\{T_s, B, A \leftarrow K_{ab} \rightarrow B, K_{ab}$ is fresh$\}_{Kas}$)
- identify and formalize the goals
- try to derive the goals from the premises and the axioms using the inference rules
  - in case of success, the protocol is secure in the SVO model
  - in case of failure, you may be able to determine a specific flaw and a specific attack based on that flaw

Example


Alice
Bob
Server
A, N_A
A, N_A, B, N_B
generate K
E_{Kas}(B | K | N_A), E_{Kbs}(A | K | N_B)
E_{Kas}(B | K | N_A)

Initial assumptions

- nonces:
  - (I1) A believes (N_a is fresh)
  - (I2) B believes (N_b is fresh)
- pre-established channels:
  - (I3) A believes ($A \leftarrow K_{as} \rightarrow S$)
  - (I4) S believes ($A \leftarrow K_{as} \rightarrow S$)
  - (I5) B believes ($B \leftarrow K_{bs} \rightarrow S$)
  - (I6) S believes ($B \leftarrow K_{bs} \rightarrow S$)
- trust:
  - (I7) A believes (S controls ($A \leftarrow K \rightarrow B$))
  - (I8) B believes (S controls ($A \leftarrow K \rightarrow B$))
  - (I9) A believes (S controls (K is fresh))
  - (I10) B believes (S controls (K is fresh))

Message reception, comprehension, interpretation

- reception:
  - (R1) B received $\{ A, K, N_b \}_{Kbs}$
  - (R2) A received $\{ B, K, N_a \}_{Kas}$

- comprehension:
  - (C1) B believes (B received $\{ A, \$_1, N_b \}_{Kbs}$)
  - (C2) A believes (A received $\{ B, \$_2, N_a \}_{Kas}$)

- interpretation:
  - (P1) B believes (
    B received $\{ A, \$_1, N_b \}_{Kbs} \Rightarrow$
    B received $\{ A, A \leftarrow K \rightarrow B, K \text{ is fresh}, N_b \}_{Kbs}$)
  - (P2) A believes (
    A received $\{ B, \$_2, N_a \}_{Kas} \Rightarrow$
    A received $\{ B, A \leftarrow K \rightarrow B, K \text{ is fresh}, N_a \}_{Kas}$)

Derivations

- C1, P1, A1 (MP):
  (D1) B believes (B received { A, A←K→B, K is fresh, N_b}$_{Kbs}$)
- A3 (N):
  (D2) B believes ((B←K_{bs}→S) & (B received {X^S}$_{Kbs}$) ⇒(S said X))
- I5, D1, D2, A1 (MP):
  (D3) B believes (S said (A, A←K→B, K is fresh, N_b))
- A17 (N):
  (D4) B believes ((N_b is fresh) ⇒((A, A←K→B, K is fresh, N_b) is fresh))
- I2, D4, A1 (MP):
  (D5) B believes ((A, A←K→B, K is fresh, N_b) is fresh)

Derivations

- A19 (N):

  (D6) B believes (((A, A←K→B, K is fresh, N_b) is fresh) &
  (S said (A, A←K→B, K is fresh, N_b)) ⇒
  (S says (A, A←K→B, K is fresh, N_b)))

- D3, D5, D6, A1 (MP):

  (D7) B believes (S says (A, A←K→B, K is fresh, N_b))

- A15 (N):

  (D8) B believes ((S says (A, A←K→B, K is fresh, N_b)) ⇒
  (S says (A←K→B)))

  (D8') B believes ((S says (A, A←K→B, K is fresh, N_b)) ⇒
  (S says (K is fresh)))

- D7, D8/D8', A1 (MP):

  (D9) B believes (S says (A←K→B))

  (D9') B believes (S says (K is fresh))

Derivations

- A16 (N):

  (D10) B believes ((S controls (A←K→B)) & (S says (A←K→B)) ⇒ (A←K→B))

  (D10') B believes ((S controls (K is fresh)) & (S says (K is fresh)) ⇒ (K is fresh))

- I8, D9, D10, A1 (MP):

  (D11) B believes (A←K→B)

- I10, D9', D10', A1 (MP):

  (D12) B believes (K is fresh)

- similar derivations work for

  A believes (A←K→B)

  A believes (K is fresh)

Semantics of the SVO logic

- SVO defines a model of computation where
  - protocol participants and the adversary are represented by state machines
  - principals can send and receive messages, generate terms, and perform computation
  - local states consists of the local history of actions (sent and received events) and available transformations
  - the local state of the adversary consists of a global history, a set of transformations available to the adversary, and a message buffer of sent but not yet delivered messages

- the truth condition of each logical construct (e.g., P said X, X is fresh, …) is precisely defined w.r.t. this model of computation
  - e.g., P believes φ holds in a global state s, if φ is true in all states s' such that s' is indistinguishable from s by P (possible worlds semantics)

- the SVO logic is proven to be **sound**: everything one can derive from a set of formula S (using the axioms and the inference rules) is indeed true in any state that satisfies S

Recommended reading

- P. Syverson, P. van Oorschot, A Unified Cryptographic Protocol Logic, NRL Technical Report, November 1996.
  - syntax
  - semantics
  - more examples