Ping-Pong Protocols as Prefix Grammars: Modelling and Verification via Program Transformation

Antonina N. Nepeivoda[1]

[1]Program Systems Institute
Russian Academy of Sciences

Fifth International Valentin Turchin Workshop on Metacomputations
Pereslavl–Zalessky, 2016

Introduction

The short plan of the talk:

1. Brief introduction to the 2-party ping-pong model and multi-party ping-pong model and discussion on an attack definition for the multi-party ping-pong protocols.
2. Description of the refined modeling algorithm for making multi-party ping-pong protocols in the Dolev-Yao intruder model into prefix grammars.
3. Introduction to a simplified verification criterion for the prefix grammar protocol models.
4. Explanation on our method of program building using the prefix grammar model.
5. Comparison of our method of verification with the classical verification algorithm and discussion of their limits.
6. Discussion on some “quick and dirty” tricks that are sometimes helpful in the task of reducing the verification time.

Intruder Model

D. Dolev & A. Yao — the first formal model of an intruder and the first formal model of ping-pong protocols (1983).

The Dolev–Yao intruder:

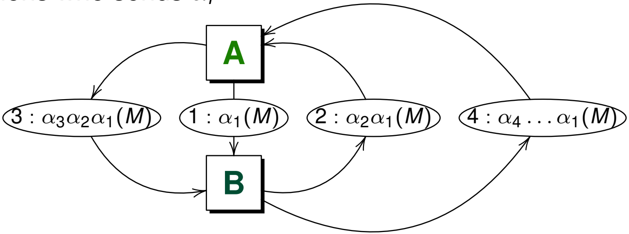
- Can intercept, modify or reuse any message in the network;
- Can disguise as any principal in the network (MIM, masquerade);
- Can initiate sessions.

- Cannot perform operations other than from a given finite set;
- Cannot guess properties of the secret operators;
- Cannot manipulate with the network itself (e.g., DDoS).

A ping-pong protocol for two principals

Principals are users who obey rules of the message exchange. Henceforth — $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ etc. *The intruder* is $\mathbf{Z}$ (one is enough).
The initial message is denoted by M (usually M is private data).
Set $\Sigma_{\mathbf{A}}$ — *the vocabulary* of user $\mathbf{A}$. Contains *operator forms* specified to users by the indices. E.g., E_x is an operator form of a public-key encryption by the key of x, D_x — decryption of E_x, a_x — prepending of the name of x, d_x — deleting a prefix equal to the name of x.

A protocol is a tuple of operator compositions α_i (*protocol words*) together with instructions who sends α_i.



Above — a generic protocol for two principals with four steps, where $\alpha_{2n+1} \in \Sigma^*_{\mathbf{A}}$ and $\alpha_{2n} \in \Sigma^*_{\mathbf{B}}$.

Dolev–Yao Intruder Model for two principals

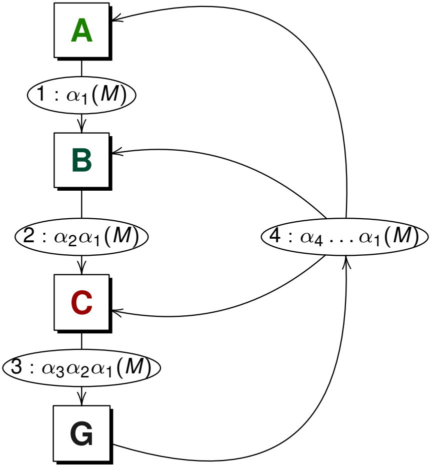
Every two-party protocol can be played (maybe partly) at most in the six instances: $\mathbf{P}[\mathbf{A},\mathbf{B}]$, $\mathbf{P}[\mathbf{A},\mathbf{Z}]$, $\mathbf{P}[\mathbf{B},\mathbf{A}]$, $\mathbf{P}[\mathbf{B},\mathbf{Z}]$, $\mathbf{P}[\mathbf{Z},\mathbf{A}]$, $\mathbf{P}[\mathbf{Z},\mathbf{B}]$; e.g. every substitution of users $\mathbf{U}_1$, $\mathbf{U}_2$ from $\{\mathbf{A},\mathbf{B},\mathbf{Z}\}$ to $\mathbf{P}[\mathbf{U}_1,\mathbf{U}_2]$ is allowed whenever $\mathbf{U}_1 \neq \mathbf{U}_2$.

Definition

A protocol $\mathbf{P}[\mathbf{A},\mathbf{B}]$ is *insecure* iff there exists such a sequence of intruder and principal actions (over the composition of the instances $\mathbf{P}[\mathbf{U}_1,\mathbf{U}_2]$) that the intruder can get some private data from the insecurity set INSEC after some manipulations with the initial message $\alpha_1[\mathbf{A},\mathbf{B}](M)$.

Ping-Pong Protocols for Many Principals

Now let the protocol be as follows ($\alpha_1 \in \Sigma^*_{\mathbf{A}}$, $\alpha_2 \in \Sigma^*_{\mathbf{B}}$, $\alpha_3 \in \Sigma^*_{\mathbf{C}}$, $\alpha_4 \in \Sigma^*_{\mathbf{G}}$).



A generalization of the Dolev-Yao intruder model

Every protocol for n parties can be played (maybe partly) at every instance $\langle \mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_n \rangle$, where $\mathbf{U}_i \in \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n\} \cup \{\mathbf{Z}\}$, and $\{\mathbf{Z}\}$ is a set of intruders.

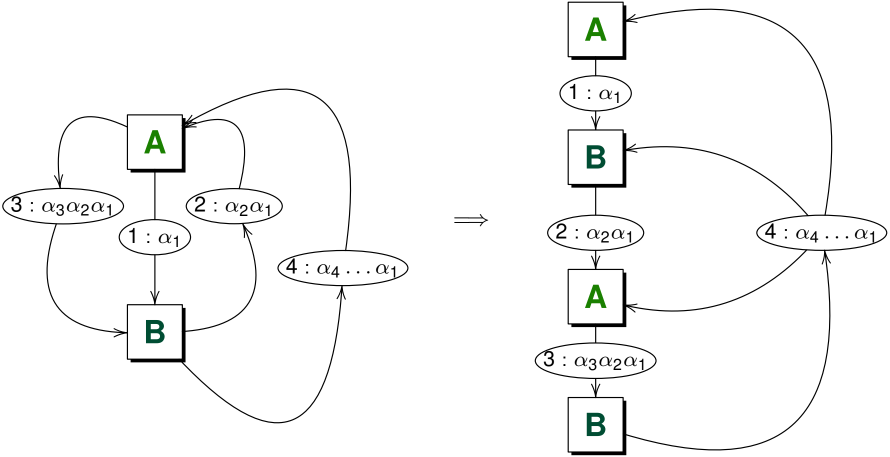
Is the condition $\mathbf{U}_i \neq \mathbf{U}_j$ satisfied for every instance $\langle \mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_n \rangle$?

If **YES (the strong attack model)**: the class of the multi-party protocol models does not include the class of the 2-party protocol models; the cardinality of the set of intruders is $O(n)$.

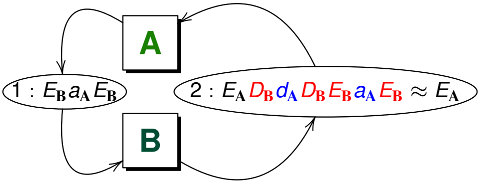
If **NO CONDITION AT ALL (the weak attack model)**: the class of the multi-party protocol models admits instances $\langle \mathbf{A}, \mathbf{A}, \ldots, \mathbf{A} \rangle$; artificial attack models; the cardinality of the set of intruders is 1.

Our restriction on the weak attack model

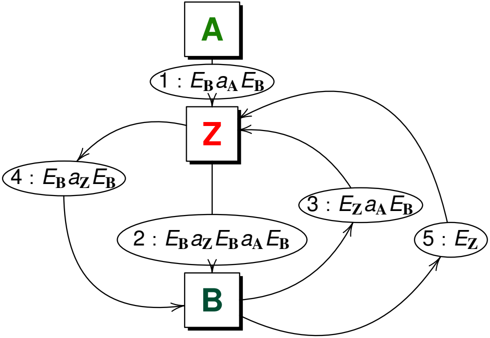
For every $\alpha_i[\mathbf{U}_1, \dots, \mathbf{U}_n]$ sent by $\mathbf{U}_k$ and any j, $\mathbf{U}_k = \mathbf{U}_j$ implies $k = j$.
The 2-party model is embedded in the multi-party model.



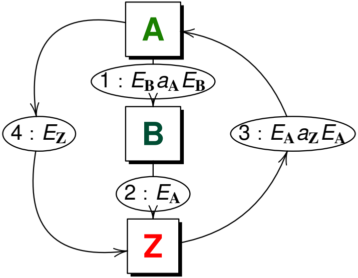
There may be several different attacks on a protocol...



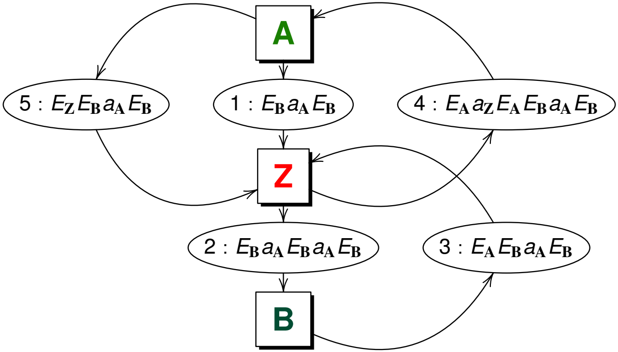
The first attack



The second attack



...and an infinite set of attacks that can be made shorter.



After the step 5, $\mathbf{Z}$ applies his/her decryption key $D_{\mathbf{Z}}$ to $E_{\mathbf{Z}} E_{\mathbf{B}} a_{\mathbf{A}} E_{\mathbf{B}}$ and the situation repeats the initial one. Then $\mathbf{Z}$ can perform any of the two attacks on the protocol and thus get a “new” attack scheme.
We are concentrated only on finding the set of the *short attacks*.

Prefix Grammars

Definition

Consider a tuple $\langle \Upsilon, \mathbf{R}, \Gamma_0 \rangle$, where Υ is an alphabet, $\Gamma_0 \in \Upsilon^+$ is an initial word and $\mathbf{R} \subset \Upsilon^* \times \Upsilon^*$ is a set of rewrite rules. If the rewrite rules are applied only to word prefixes $\dfrac{R : \Phi \longrightarrow \Psi}{\Phi\Theta \xrightarrow{R} \Psi\Theta}$ then the tuple $\langle \Sigma, \mathbf{R}, \Gamma_0 \rangle$ is called *a prefix grammar*.

Every one-step interaction of the described protocol & intruder model can be considered as a set of rules in a prefix grammar:

- an application of a protocol word $\alpha_i[\mathrm{U}_1, \ldots \mathrm{U}_n]$ can be modeled by applying $\varepsilon \to \alpha_i[\mathrm{U}_1, \ldots \mathrm{U}_n]$ and then doing all possible variants of cancellations (applications of $x_1 x_2 \ldots x_n \to \varepsilon$, e.g. $D_x E_x \to \varepsilon$).
- an action of an intruder can be modeled either by the rule $\varepsilon \to x$ (if $x \in \Sigma_{\mathbf{Z}}$) or by the rule $x_1 \ldots x_n \to \varepsilon$ (if there is some $y \in \Sigma_{\mathbf{Z}}$ s.t. $y x_1 \ldots x_n \to \varepsilon$).

The size of a resulting model, 1

Given *n* parties, every protocol word $\alpha_i[\mathbf{U}_1, \dots \mathbf{U}_n]$ must generate at least one rewrite rule for every instance $[\mathbf{U}_1, \dots, \mathbf{U}_n]$ that can appear in the restricted attack model $\Rightarrow$ the size of the rule set grows exponentially in *n*. Thus, every extra grammar rule can cause practical non-applicability of the verification algorithm.

It is reasonable to reduce the number of considerable variants of cancellations by doing all cancellations as early as it is possible.

The size of a resulting model, 2

Rewrite rules $R_l \to R_r$, s.t. R_r contains an operator e, e is not present in INSEC, e has no left inverse, are redundant (since e is either erased immediately or never erased).

Example

d_x has no left inverses, so it is reasonable to apply protocol word $E_x D_y d_x D_y$ only to words with the prefix $E_y a_x$. The rules

$$\varepsilon \to E_x D_y d_x D_y$$

$$E_y \to E_x D_y d_x$$

are redundant.

Instances of protocol words $\alpha_i[\mathbf{U}_1, \ldots, \mathbf{U}_n]$ s.t. $\alpha_i[\mathbf{U}_1, \ldots, \mathbf{U}_n] \in \Sigma_{\mathbf{Z}}^*$ are also redundant.

A simplified criterion of a short attack, 1

Definition

A prefix grammar **G** is *annotated* if every right-hand side of a rule of **G** is either prefixed by or a prefix of another right-hand side or shares no letter with it.

Simple idea: to use colors to annotate right-hand sides of the rules.

Definition

Let **G** be an annotated prefix grammar. Let us say that Γ is *lhs-redundant* iff for some a the number of occurrences of a in Γ is greater than the number of different prefixes preceding a in the left-hand sides of rewrite rules of the grammar **G**. For every $a \in \Upsilon$ the number of different prefixes preceding a in the left-hand sides is called *an erasing limit of a* (denoted by $EL(a)$).

Example

$\mathbf{G_{2EXP}} = \langle \{a, b, c, A, B, C, i\}, \mathbf{R_{2EXP}}, i \rangle$.

The set of rewrite rules $\mathbf{R_{2EXP}}$ is:

$R^{[1]} : i \to aA$	$R^{[5]} : AA \to \varepsilon$	$R^{[9]} : Ba \to bB$
$R^{[2]} : \varepsilon \to aA$	$R^{[6]} : BB \to \varepsilon$	$R^{[10]} : Cb \to cC$
$R^{[3]} : \varepsilon \to bB$	$R^{[7]} : CC \to \varepsilon$	
$R^{[4]} : \varepsilon \to cC$	$R^{[8]} : c \to \varepsilon$	

The grammar is annotated. The erasing limit $EL(i) = 1$; also $EL(a) = EL(b) = EL(c) = 1$, so the words $aAaA$ and $cCcC$ are redundant. The word cCC is not lhs-redundant since $EL(C) = 2$.

A simplified criterion of a short attack, 2

Theorem

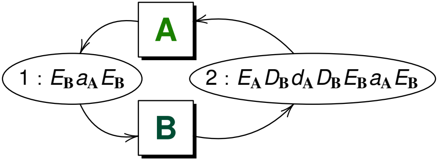
Let **G** *be a finite annotated prefix grammar. Every infinite trace generated by* **G** *either contains some* Γ *and* Δ *such that* $\Gamma = \Delta$*, or contains an lhs-redundant word.*

Theorem

Let **G** *be an arbitrary finite annotated prefix grammar. All short attack models generated by* **G** *contain no* Γ *and* Δ *such that* $\Gamma = \Delta$ *or* Γ *is lhs-redundant.*

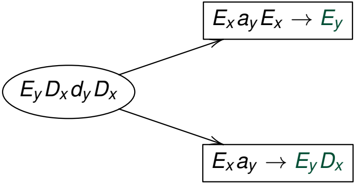
No time annotation is needed for this case, but the annotating procedure produces more distinct rewrite rules.

A prefix grammar from a protocol: an example, 1



$\mathrm{INSEC} = \{\varepsilon\}$. The useful protocol words are: $\alpha_2[\mathbf{B},\mathbf{A}] = E_{\mathbf{A}} D_{\mathbf{B}} d_{\mathbf{A}} D_{\mathbf{B}}$; $\alpha_2[\mathbf{A},\mathbf{B}] = E_{\mathbf{B}} D_{\mathbf{A}} d_{\mathbf{B}} D_{\mathbf{A}}$; $\alpha_2[\mathbf{A},\mathbf{Z}] = E_{\mathbf{Z}} D_{\mathbf{A}} d_{\mathbf{Z}} D_{\mathbf{A}}$; $\alpha_2[\mathbf{B},\mathbf{Z}] = E_{\mathbf{Z}} D_{\mathbf{B}} d_{\mathbf{Z}} D_{\mathbf{B}}$.

Each generates the two rewrite rules (with the similarly colored right-hand sides!):



A prefix grammar from a protocol: an example, 2

The intruder alphabet is $\{E_{\mathbf{A}}, E_{\mathbf{B}}, E_{\mathbf{Z}}, a_{\mathbf{A}}, a_{\mathbf{B}}, a_{\mathbf{Z}}, D_{\mathbf{Z}}, d_{\mathbf{A}}, d_{\mathbf{B}}, d_{\mathbf{Z}}\}$. Thus, the additional rules are:

$\varepsilon \to E_{\mathbf{A}}$	$a_{\mathbf{A}} \to \varepsilon$	$D_{\mathbf{A}} \to \varepsilon$
$\varepsilon \to E_{\mathbf{B}}$	$a_{\mathbf{B}} \to \varepsilon$	$D_{\mathbf{B}} \to \varepsilon$
$\varepsilon \to E_{\mathbf{Z}}$	$a_{\mathbf{Z}} \to \varepsilon$	
$\varepsilon \to a_{\mathbf{A}}$	$E_{\mathbf{Z}} \to \varepsilon$	
$\varepsilon \to a_{\mathbf{B}}$	$D_{\mathbf{Z}} \to \varepsilon$	
$\varepsilon \to a_{\mathbf{Z}}$		

The initial word is $E_{\mathbf{B}} a_{\mathbf{A}} E_{\mathbf{B}}$. E_x in the right-hand sides may be colored by any color except green.

The color does not matter for the left-hand sides of the rules. The rules $D_x E_x \to \varepsilon$ are not useful, since in these cases the rule $E_x d_y E_x \to E_y$ is to be used instead of $E_x d_y \to E_y D_x$. $E_{\mathbf{Z}} D_{\mathbf{Z}} \to \varepsilon$ is not useful — it is a composition of the two other rules.

From the grammar to a program, 1

- Assign the erasing limit for every letter and assign a counter of the letter in the current word. If the counter exceeds the erasing limit, then stop — no short attack exists that can contain the current word.
- Determine a set of the final states of the program — they are the words from INSEC.
- If the program transformation technique uses generalization, it must be made unavailable. The only needs of the verification process is the unfolding and looping back to the same configuration.

From the grammar to a program, 2

Only one function is in the model program

`F((History),(Array_of_Counters), Secure_Word) = An Attack Found;`

or (if `Counter_Ai`$>$ *EL*(`Ai`))

`F((History),(Counter_A1, ... Counter_Ai, ... Counter_AN), Current_Word) = Stop;`

or

`F((R_i, History),(Array_of_Counters), Current_Word) = F((History), (Upd_Array_of_Counters), New_Word);`

`R_i` is the name (or the number) of the grammar rule that is applied to the `Current_Word`.

The initial call of `F` is `F((History_Param),(Array_Const),Init_Word_Const)`, where `History_Param` is an undetermined parameter and `Array_Const` and `Init_Word_Const` are the (determined) initial counters and the initial word respectively.

Equivalency for the Classical Case

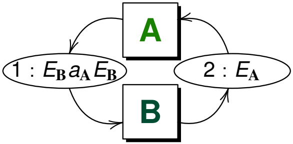
Definition

An automata model Aut_P for a protocol $P[x_1, ..., x_n]$ is a finite automaton defined as follows.

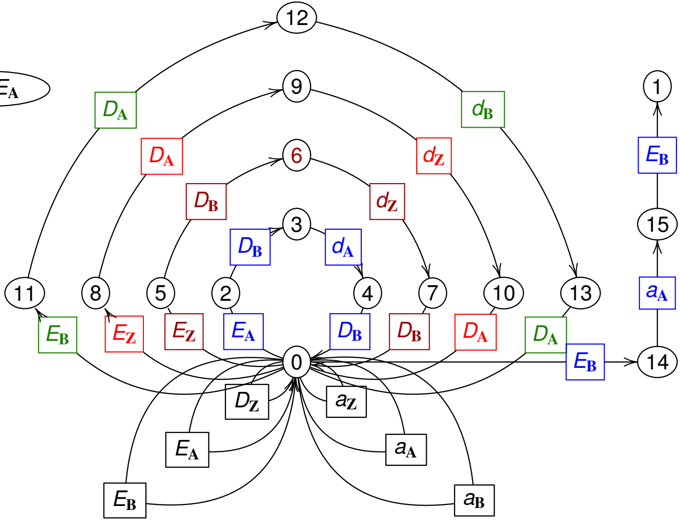
1. State 0 is the unique initial state and state 1 is the unique final state of Aut_P. The input alphabet is the union of all users' alphabets.
2. There is a directed path from 0 to 1 whose edges are labelled by operators of word $\alpha_1[\mathbf{U}_1, \ldots, \mathbf{U}_n]$. Between every two consequent edges, a non-final state is introduced.
3. For every input letter $\sigma \in \Sigma_{\mathbf{I}}$ there is a self-loop from 0 to 0, labelled by σ.
4. For every semiproper instance $\alpha_i[\mathbf{U}_{k_1}, \ldots, \mathbf{U}_{k_n}]$ of $\alpha_i[x_1, \ldots, x_n] \in P[x_1, \ldots, x_n]$, there is a directed loop from 0 to 0 whose edges are labelled by the operators of $\alpha_i[\mathbf{U}_{k_1}, \ldots, \mathbf{U}_{k_n}]$. Between every two consequent edges, a non-final state is introduced.
5. There are no other states and edges in Aut_P.

Example: Automaton Model for $\mathbf{P}_{\mathrm{Double}}[\mathbf{A},\mathbf{B}]$

Protocol



Automaton Model



Instances

$\mathbf{P}_{\mathrm{Double}}[\mathbf{A},\mathbf{B}]$

$\mathbf{P}_{\mathrm{Double}}[\mathbf{B},\mathbf{A}]$

$\mathbf{P}_{\mathrm{Double}}[\mathbf{A},\mathbf{Z}]$

$\mathbf{P}_{\mathrm{Double}}[\mathbf{Z},\mathbf{B}]$

Consistency

A collapsing path — a path containing of the edges whose labels, given in composition, are equal to ε.

Theorem

The attack corresponding to the shortest collapsing path from state 0 *to state* 1 *in the automaton model is always found by the verification prefix grammar model.*

...but the algorithm for the automata model cannot deal with:

- INSEC containing anything except ε;
- cancellation rules except $xy \to \varepsilon$;
- "universal keys" like U, where $UE_{\mathbf{A}} = \varepsilon$ and $E_{\mathbf{B}}U = \varepsilon$.

Moreover, its result is 1 bit about security, not revealing the attacks if they exist.

Verifying Needham–Schroeder Protocol

Protocol $\mathbf{P}_{\mathrm{NS}}[\mathbf{A},\mathbf{B}]$, $\mathrm{INSEC} = \{N_{\mathbf{B}}\}$ ($O_x N_x = \varepsilon$, but not vice versa):

- $\alpha_1[\mathbf{A},\mathbf{B}] = (\mathbf{A}, E_{\mathbf{B}}\, a_{\mathbf{A}}\, N_{\mathbf{A}})$
- $\alpha_2[\mathbf{A},\mathbf{B}] = (\mathbf{B}, E_{\mathbf{A}}\, N_{\mathbf{A}}\, N_{\mathbf{B}}\, O_{\mathbf{A}}\, d_{\mathbf{A}}\, D_{\mathbf{B}})$
- $\alpha_3[\mathbf{A},\mathbf{B}] = (\mathbf{A}, E_{\mathbf{B}}\, O_{\mathbf{A}}\, D_{\mathbf{A}})$

Obstacles

- $\mathbf{B}$ "knowing" $N_{\mathbf{A}}$ in advance — not a real problem. $O_{\mathbf{A}}$ has only a right inverse, so it "spoils" the message any time when applied not to $N_{\mathbf{A}}$.
- $N_{\mathbf{A}}$ "for everyone". The real flaw — since nonces are generated for a concrete interaction. Can be partly fixed by introducing $N_{x \to y}$ — specified both by the sender and the recipient $\to$ growth of the model size.

Noteworthy: Lowe's symbolic model checking verification also considered only a bounded number of nonces!

Verifying Shamir 3-pass Protocol

Protocol $\mathbf{P}_{\mathrm{S3}}[\mathbf{A}, \mathbf{B}]$, $\mathrm{INSEC} = \{\varepsilon\}$ ($S_x S_x = \varepsilon$):

- $\alpha_1[\mathbf{A}, \mathbf{B}] = (\mathbf{A}, S_{\mathbf{A}})$
- $\alpha_2[\mathbf{A}, \mathbf{B}] = (\mathbf{B}, S_{\mathbf{B}})$
- $\alpha_3[\mathbf{A}, \mathbf{B}] = (\mathbf{A}, S_{\mathbf{A}})$

Obstacles

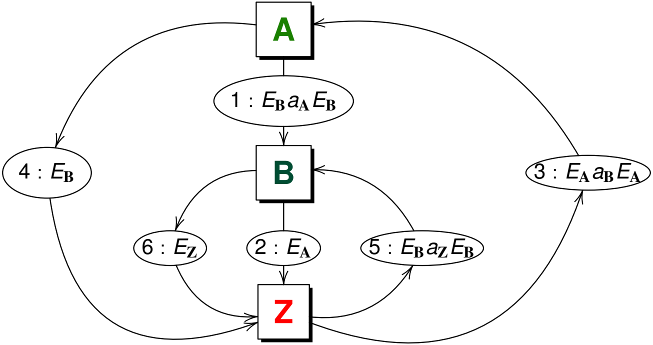
- MIM attacks where $\mathbf{A}$ explicitly decrypts her own encryption. May exist if $\mathbf{A}$ is a robot or applies encryption automatically.
- More serious: commutativity. Adding rewrite rules $XY \to YX$ to the model does not help since their application depth is not bounded.

Limits of the Suggested Model

- The set of the operator forms is finite, the operators are unary. E.g., the number of nonces in the Needham–Shroeder protocol can be only finite.
- A restricted notion of the privacy (based on the set INSEC). E.g., the MIM attack on the Diffie–Hellman protocol: the intruder does not receive secret data, but makes the principals to receive false data instead.
- No operator equations besides the cancellation rules. E.g., problems with commutative operations as $\mathtt{XOR}$ or multiplication.

Quick and dirty tricks to make the verification faster, 1

Using symmetry. If all the alphabets can be presented as $\{OP^1_{i_1}, \ldots, OP^n_{i_n}\}$, where only i_k are uniformly changed, then reaching the word $\alpha_1[\mathbf{U}_{k_1}, \mathbf{U}_{k_2}, \ldots, \mathbf{U}_{k_m}]$ where $\mathbf{U}_{k_i} \neq \mathbf{Z}$ are arranged in the same way as in $\alpha_1[\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_m]$, then a short attack on $\alpha_1[\mathbf{U}_{k_1}, \mathbf{U}_{k_2}, \ldots, \mathbf{U}_{k_m}]$ will repeat short attacks on $\alpha_1[\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_m]$ up to the users' arrangement.



Quick and dirty tricks to make the verification faster, 2

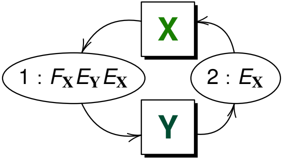
Using inversion.

In most protocols, an intruder is allowed more to append than to erase. If all the rules are transformed from $R_l \to R_r$ to $R_r \to R_l$, the final states become the initial states; and the initial state becomes (a unique) final state, sometimes the verification process takes significantly less time.

Small Example

Let $E_{\mathbf{X}}F_{\mathbf{X}}$ be an open key encryption with both individual key $E_{\mathbf{X}}$ known by anyone and registered site key $F_{\mathbf{X}}$ known only by administrator $\mathbf{B}$.
Let $F_x G_x \neq \varepsilon$, $G_x F_x = \varepsilon$.
The protocol used by site visitors is

$$\mathbf{P_{LA}}[\mathbf{X}, \mathbf{Y}] = ((\mathbf{X}, F_{\mathbf{X}}E_{\mathbf{Y}}), (\mathbf{Y}, E_{\mathbf{X}}D_{\mathbf{Y}}G_{\mathbf{X}})).$$

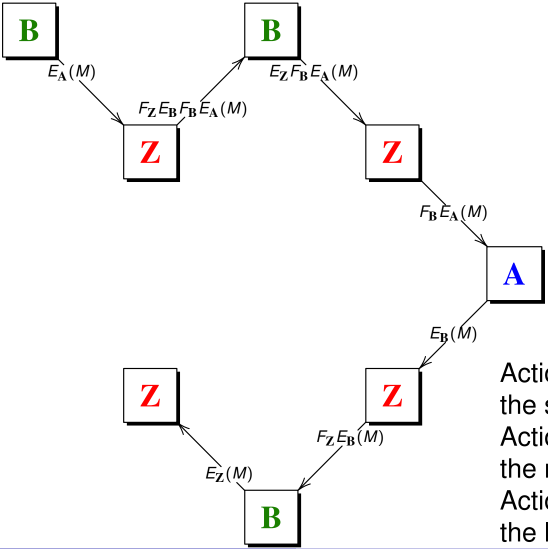


Besides $\mathbf{B}$ there is a programmer $\mathbf{A}$, $G_{\mathbf{B}} \in \Sigma_{\mathbf{A}}$ who also can use $\mathbf{P_{LA}}$ to confirm identity of $\mathbf{B}$. $\mathbf{A}$ participates in protocol plays only with $\mathbf{B}$.

Is $E_{\mathbf{A}}$ secure?

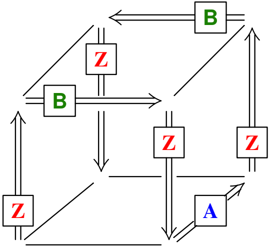
The Attack on $\mathbf{P_{LA}}$ and the Hanoi Puzzle

The attack scheme



The puzzle solution scheme

(for 3 disks)



Actions of $\mathbf{Z}$ — replacements of the smallest disk.
Actions of $\mathbf{B}$ — replacements of the middle disk.
Actions of $\mathbf{A}$ — replacements of the largest disk.

Conclusion

- CAN be applicable.
- Without a restriction to a special generalization / termination technique;
- Widely applicable (the class of verified protocols is wider than the classical ping-pong protocols)

- Computational complexity grows fast in the general case, and special cases require special efforts;
- Almost no “necessary and sufficient” conditions — only for very restricted (annotated) models.

Thank You