II Workshop on Verification and Program Transformation, Vienna, July, 17, 2014

Verification of Multi-Party Ping-Pong Protocols via Program Transformation

Antonina Nepeivoda
Program Systems Institute of RAS

Pereslavl-Zalessky

Introduction

Outline:

1. An 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. The refined algorithm modeling the multi-party ping-pong protocols in the Dolev-Yao intruder model by means of prefix grammars.

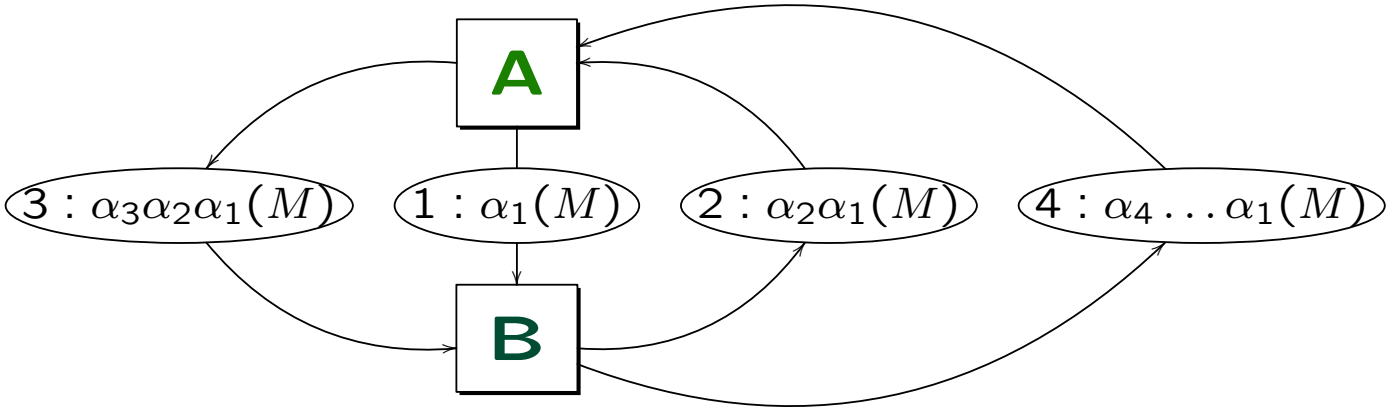
3. A simplified verification criterion for the prefix grammar protocol models.

4. Our method of program building using the prefix grammar model.

5. Discussion on some "quick and dirty" tricks that sometimes help to reduce the verification time.

A Ping-Pong Protocol for Two Principals

Let $\mathbf{A}$ and $\mathbf{B}$ send pieces of data in course, as while playing ping-pong, by an open channel. M — a secret data piece to be transmitted; Σ_A and Σ_B — sets of elementary operators available to $\mathbf{A}$ and $\mathbf{B}$ respectively.



where $\alpha_{2n+1} \in \Sigma_A^*$ and $\alpha_{2n} \in \Sigma_B^*$.

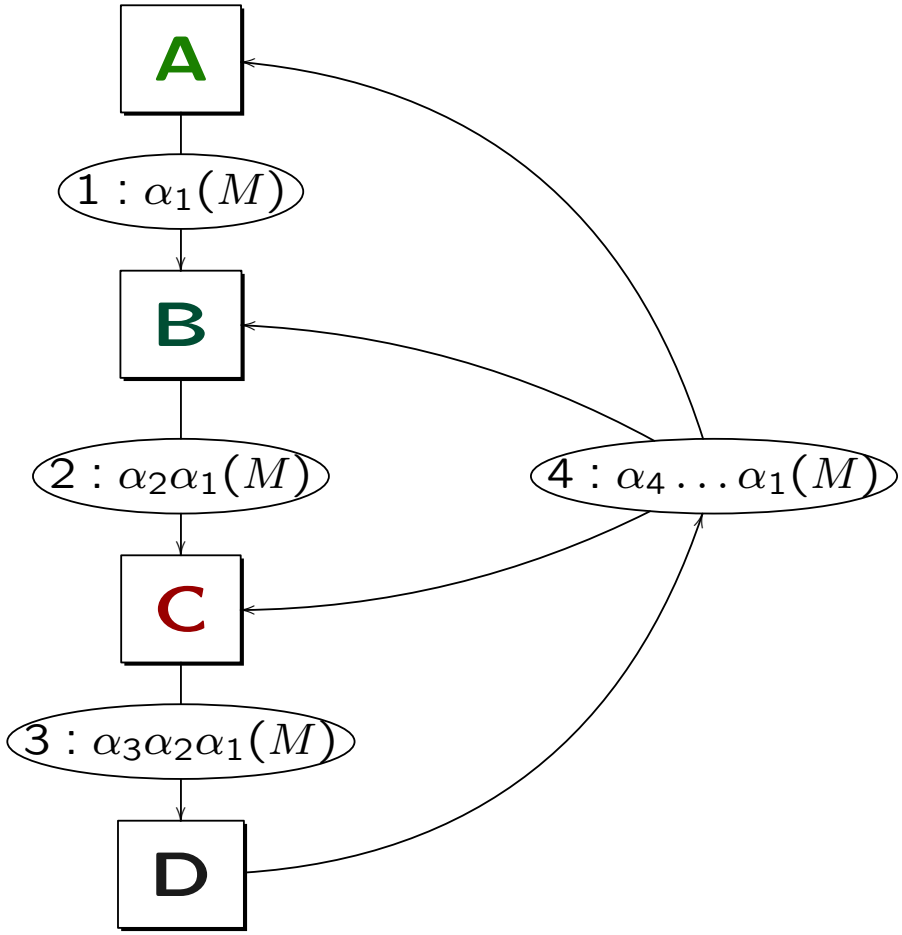
Dolev–Yao Intruder Model for Two Principals

The intruder Z can initiate multiple interactions by $\mathbf{P(A,B)}$ (playing a role of $\mathbf{A}$ or $\mathbf{B}$ in each of them or waiting for some special transmission from one principal to another).

So every protocol can be played at most in the six following instances: $\mathbf{P(A,B)}$, $\mathbf{P(B,A)}$, $\mathbf{P(A,Z)}$, $\mathbf{P(Z,A)}$, $\mathbf{P(Z,B)}$, and $\mathbf{P(B,Z)}$; e.g. every substitution of users U_1, U_2 from $\{A, B, Z\}$ to $\mathbf{P(U_1, U_2)}$ is allowed whenever $U_1 \neq U_2$.

$\mathbf{Z}$ can tackle any message x sent from $\mathbf{U_1}$ to $\mathbf{U_2}$ or vice versa and replace it by $\beta(x)$, $\beta \in \Sigma^*_{\mathbf{Z}}$. A protocol $\mathbf{P(A,B)}$ is insecure iff there exists such a sequence of actions (over the union of composition of the instances $\mathbf{P(U_1, U_2)}$ and Σ^*_Z) that Z can get M from the initial message $\alpha_1[\mathbf{A,B}](M)$.

Now let an interaction involve e.g. four legal participants, and the transmission process prescribed by the protocol be as follows ($\alpha_1 \in \Sigma_A^*$, $\alpha_2 \in \Sigma_B^*$, $\alpha_3 \in \Sigma_C^*$, $\alpha_4 \in \Sigma_D^*$).



A Generalization of the Dolev-Yao Intruder Model

Every protocol for n parties can be played (maybe partly) in every instance $\langle U_1, U_2, \ldots, U_n \rangle$, where $U_n \in \{A_1, A_2, \ldots, A_n\} \cup \mathbf{I}$, and $\mathbf{I}$ is a set of intruders.

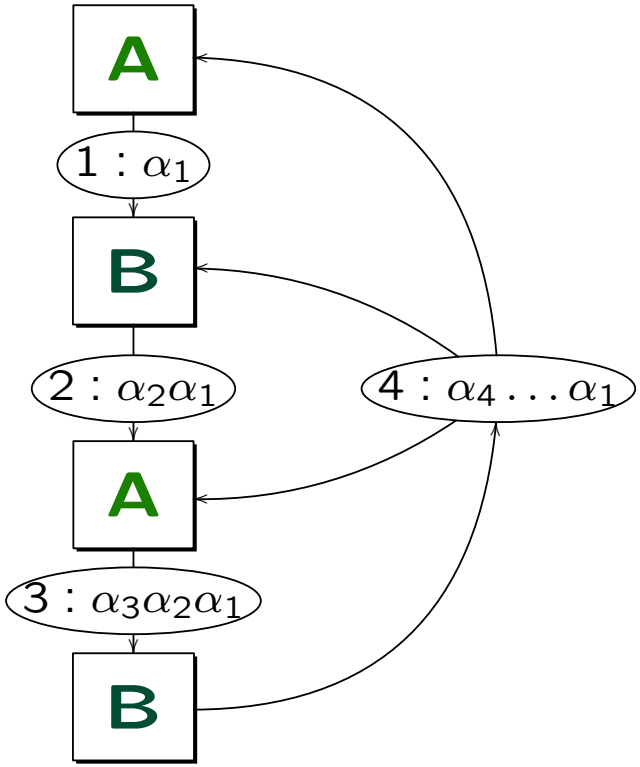
Is $U_i \neq U_j$ for every instance $\langle U_1, U_2, \ldots, U_n \rangle$?

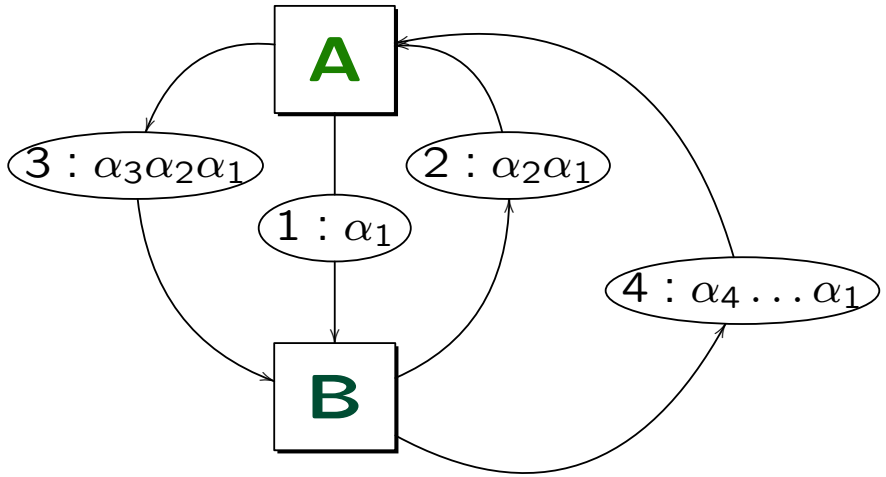
If **YES (the strong attack model)**: the class of the multi-party protocol models does not include the class of the 2-party protocol models; the cardinality of the set of intruders $\mathbf{I}$ is $O(n)$.

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

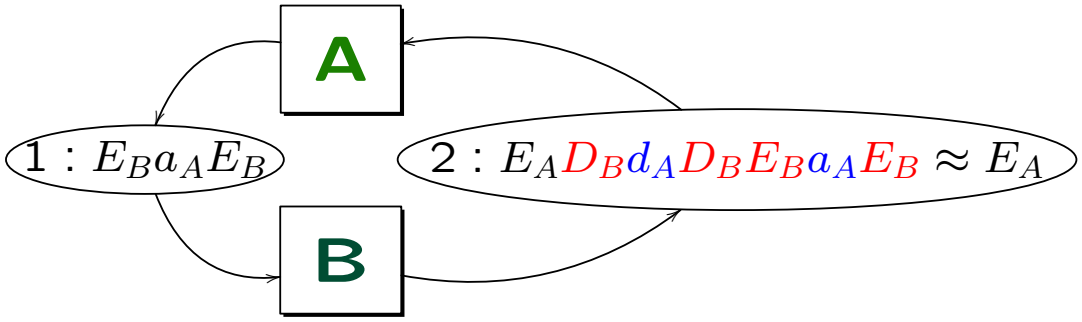
Our Restriction on the Weak Attack Model

If $\langle U_1, U_2, \ldots, U_n \rangle$ is an instance, for every $\alpha_i[\mathbf{U_1}, \ldots, \mathbf{U_n}]$ and any j, $U_i = U_j$ implies $i = j$ (i.e. no principal does transmissions to himself, except maybe the last one). The cardinality of $\mathbf{I}$ is 1; the 2-party model is embedded in the multi-party model.

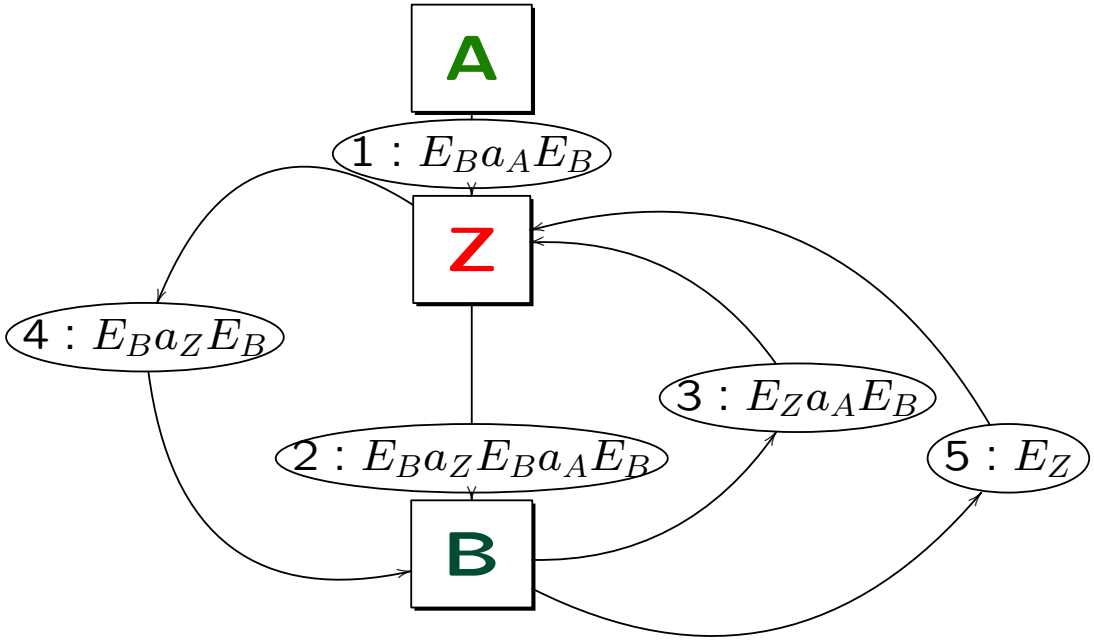




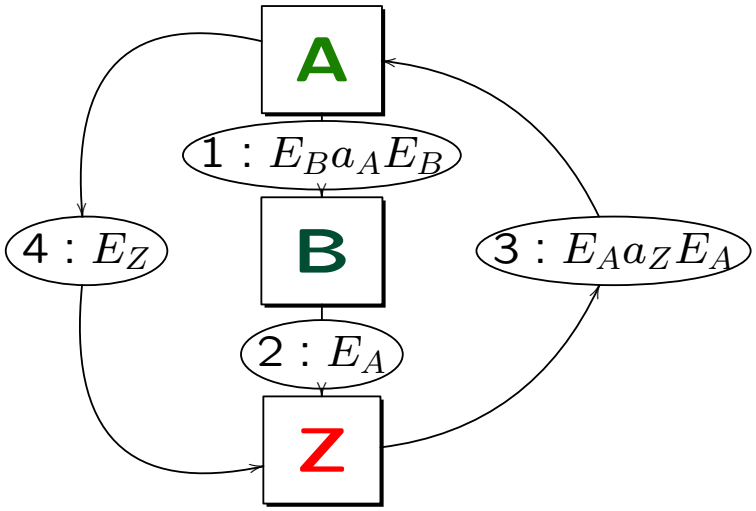
There may be several different attacks on a protocol



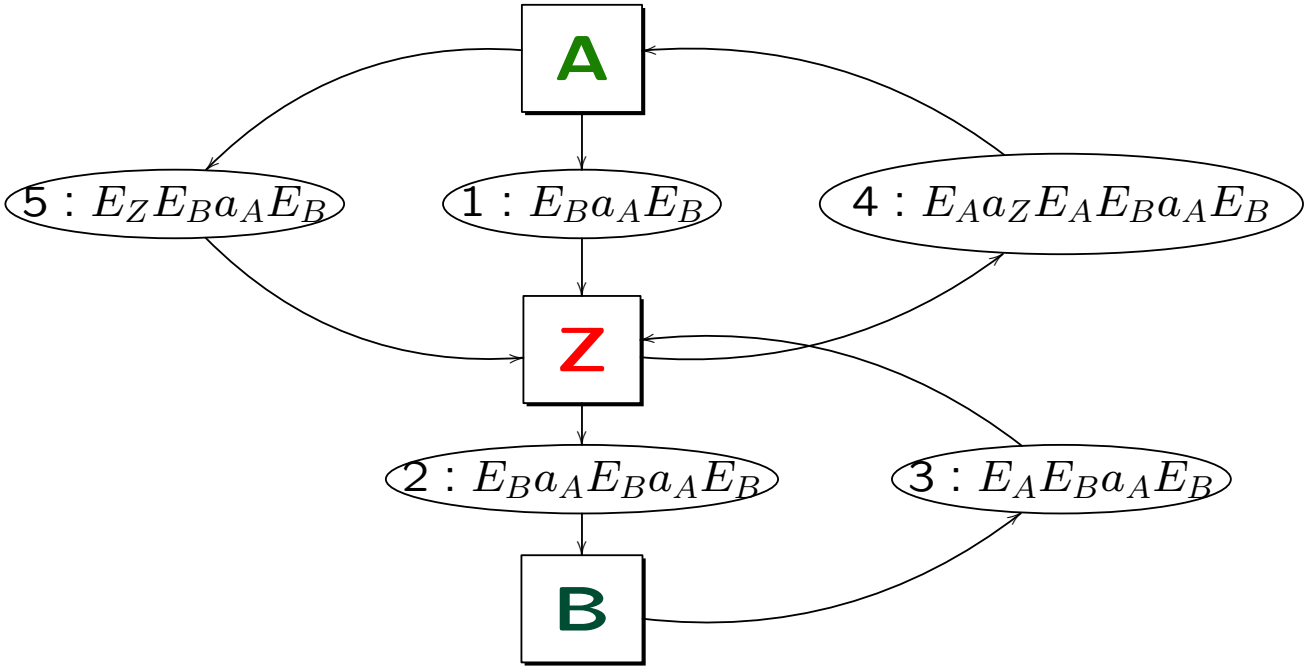
The first attack



The second attack



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



After the step 5, $\mathbf{Z}$ applies his decryption key D_Z to $E_Z E_B a_A E_B$ and the current state 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.

Consider $\langle \Upsilon, \mathbf{R}, \Gamma_0 \rangle$, 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 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*.

A one-step action over a protocol controlled by the Dolev-Yao intruder model can be considered as a set of prefix rewrite rules:

- If the action is an application of $\alpha_i[\mathbf{U_1}, \ldots \mathbf{U_n}]$, it can be modeled by appending $\alpha_i[\mathbf{U_1}, \ldots \mathbf{U_n}]$ and then doing all possible variants of cancellations (applications of $xy \to \varepsilon$, e.g. $D_x E_x \to \varepsilon$).

- If the action is an action of an intruder, it can be modeled either by the rule $\varepsilon \to x$ (if $x \in \Sigma_Z$) or by the rule $x \to \varepsilon$ (if there is some $y \in \Sigma_Z$ s.t. $yx \to \varepsilon$).

Reducing the Size of a Resulting Model

Having n parties, every protocol word must generate a rewrite rule for every $[U_1, \ldots, U_n]$ that can appear in the restricted attack model $\Rightarrow$ the size of the rule set grows exponentially $\Rightarrow$ every extra grammar rule can cause practical non-applicability.

If operators used in a protocol are connected by no equations other than $xy \approx \varepsilon$, then the corresponding algebra is called *an algebra with the freeness assumption*.

Example

The operator set $\{E_{x_1}, \ldots, E_{x_n}, D_{x_1}, \ldots, D_{x_n}, a_{x_1}, \ldots, a_{x_n}, d_{x_1}, \ldots, d_{x_n}\}$ generates an algebra with the freeness assumption.

This property helps to reduce the number of considerable variants of cancellations by doing all cancellations as early as it is possible.

Reducing the Size of the Prefix Grammar

Having operators with no left inverses (i.e. the elements x that satisfy no equations of the form $yx \approx \varepsilon$ but maybe satisfy some equations of the form $xy \approx \varepsilon$), we can reduce the number of rules: all such elements x must not appear in the right-hand sides (such x are erased either immediately or never).

Example

Name deleting operators d_{x_i} have no left inverses, so it is reasonable to apply a protocol word $\varepsilon \to E_1 a_2 d_3 D_4 d_5$ not to every word in the trace, but only to words with the prefix $a_5 E_4 a_3$.

Instances of protocol words $\alpha_i[\mathbf{U_1}, \ldots, \mathbf{U_n}]$ s.t. $\alpha_i[\mathbf{U_1}, \ldots, \mathbf{U_n}] \in \Sigma_Z^*$, also can be ignored.

Preliminaries for the Simplified Attack Criterion

A prefix grammar $\mathbf{G}$ is called *annotated* if every pair of rules either have the same right-hand side or have no letters shared by their right-hand sides. A prefix grammar can be annotated as follows: each right-hand side is numbered by an unique number, and all the letters in the right-hand side of the rule are marked by this number.

Let $\mathbf{G}$ be a prefix grammar with an alphabet Υ. We say that $\Gamma \in \Upsilon^*$ is redundant iff for some $a \in \Upsilon$ the number of occurrences of a in Γ is greater than the number of a occurrences in the left-hand sides of rewrite rules defining $\mathbf{G}$. For every $a \in \Upsilon$ the number of occurrences of a in the left-hand sides is called *an erasing limit of* a (denoted by $EL(a)$).

Example

Consider $\mathbf{G_{2EXP}} = \langle \{a, b, c, A, B, C, i\}, \mathbf{R_{2EXP}}, i \rangle$ with the following set of rewrite rules $\mathbf{R_{2EXP}}$:

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

$\mathbf{G_{2EXP}}$ is annotated since every two right-hand sides either coincide or share no letters. 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 redundant since $EL(C) = 3$.

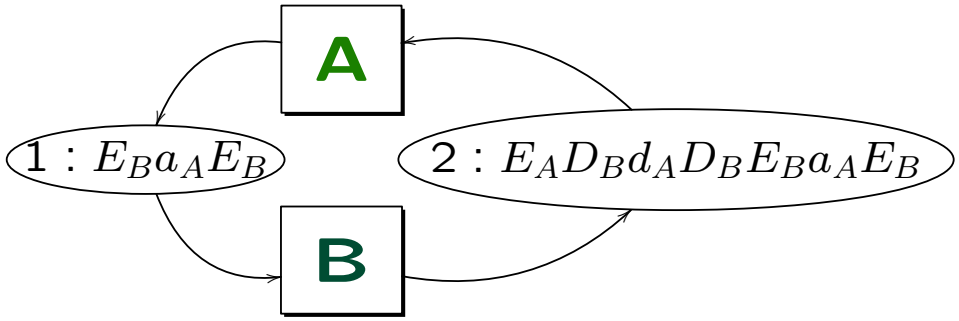
The Simplified Attack Criterion

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

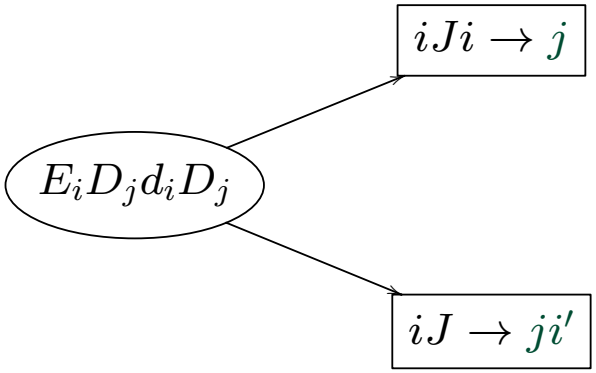
Let $\mathbf{G}$ be an arbitrary finite annotated prefix grammar. All short attack models generated by $\mathbf{G}$ contain no Γ and Δ such that $\Gamma = \Delta$ or Γ is redundant.

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



The useful instances are: $\alpha_2[\mathbf{B}, \mathbf{A}] = E_A D_B d_A D_B$; $\alpha_2[\mathbf{A}, \mathbf{B}] = E_B D_A d_B D_A$; $\alpha_2[\mathbf{A}, \mathbf{Z}] = E_Z D_A d_Z D_A$; $\alpha_2[\mathbf{B}, \mathbf{Z}] = E_Z D_B d_Z D_B$. Each of them generates two rewrite rules:



x models E_x; X models a_x; x' models D_x.

A Prefix Grammar from a Protocol: an Example, 2

The intruder alphabet is $\{E_A, E_B, E_Z, a_A, a_B, a_Z, D_Z, d_A, d_B, d_Z\}$. It together with the general cancellation conditions generate the following rules:

$\varepsilon \to a$	$A \to \varepsilon$	$aa' \to \varepsilon$
$\varepsilon \to b$	$B \to \varepsilon$	$bb' \to \varepsilon$
$\varepsilon \to z$	$Z \to \varepsilon$	
$\varepsilon \to A$	$z \to \varepsilon$	
$\varepsilon \to B$	$z' \to \varepsilon$	
$\varepsilon \to Z$		
$\varepsilon \to z'$		

The initial word is bAb.

The color does not have influence on the left-hand sides of the rules. The rules of the form $i'i \to \varepsilon$ are not useful, since in these cases $iJi \to j$ is used instead of $iJ \to ji'$. $zz' \to \varepsilon$ is not useful since it is a composition of two other rules.

Preliminaries for the Program Constructing

- Assign the erasing limit to 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: e.g. some letters from Σ_U can be private (like in the case of the Needham—Shroeder protocol); then the set of final states must contain not only ε but also such letters.

- If the program transformation technique uses generalization, it must be made unavailable. The only need of the verification process is the unfolding and looping back to the same configuration.

The Program Constructing

Only one function is in the model program. Its definition looks as follows:

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

or (if `Counter_Ai`$> EL(A_i)$)

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

or

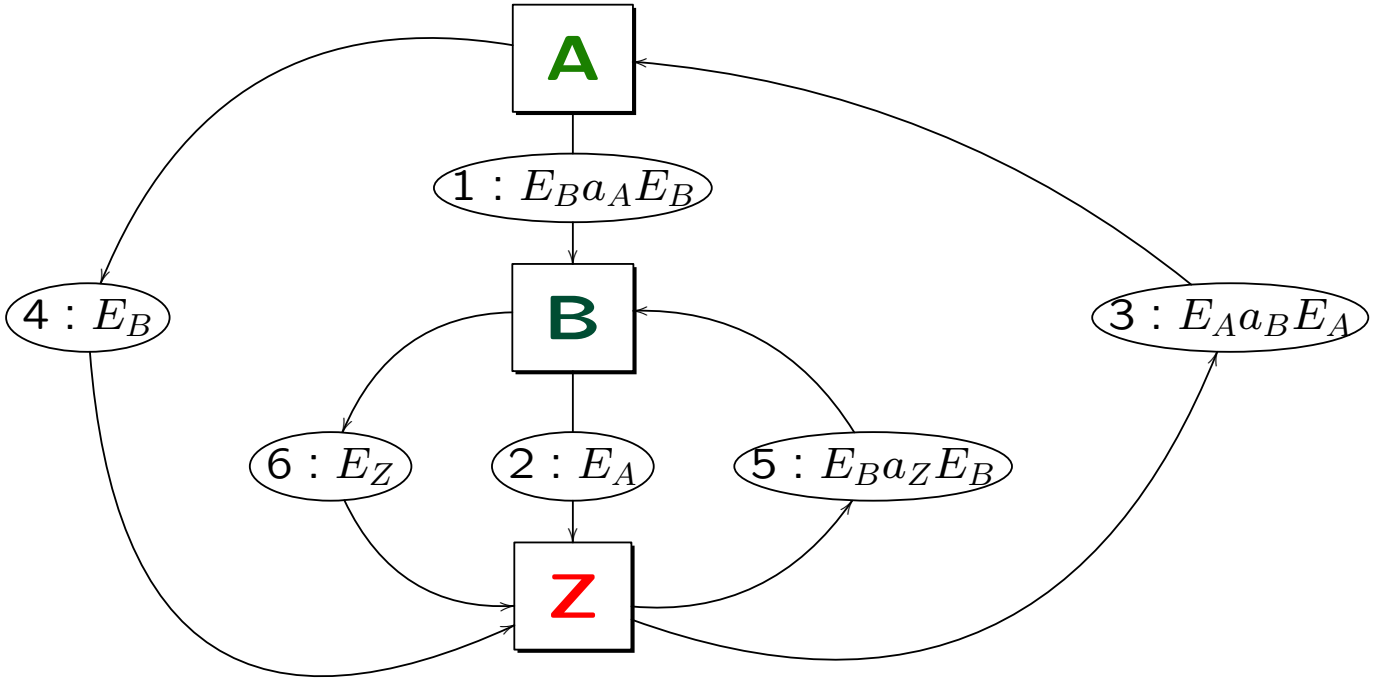
```
F(((R_i), History),(Array_of_Counters), Current_Word) =
                F((History), (Upd_Array_of_Counters), New_Word);
```

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

The initial call is `F((History_Param),(Array_Const),Init_Word_Const)`, where `History_Param` is a parameter and `Array_Const` and `Init_Word_Const` are fixed.

Using Symmetry to Reduce the Model

If all the alphabets contain the same set of operator forms (i.e. 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 $U_{k_i} \neq Z$ are arranged in the same way as in the initial protocol word $\alpha_1[\mathbf{U_1}, \mathbf{U_2}, \ldots, \mathbf{U_m}]$, 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.



Using Inversion to Reduce the Model

In most protocols, an intruder is allowed rather to append than to erase. If all the rules are transformed from $R_l \to R_r$ to $R_r \to R_l$, and the sets of the initial and final states are swapped, sometimes the verification process takes significantly less time.

For instance, $\mathbf{G_{2EXP+INV}}$ with the initial word ε, the secure word i and the following rewrite rules:

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

generates the same "attack model" of the length 80 as its "straightforward" sibling, $\mathbf{G_{2EXP}}$, but does it much faster.

Conclusion

- CAN be applicable.
- Without a restriction to a special generalization / termination technique;
- Widely applicable (for multi-party protocols, multiple protocols with the common keys, etc.)
- 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.

Thanks!