Modeling Attack-Defense Trees Countermeasures using Continuous Time Markov Chains

Karim Lounis and **Samir Ouchani**

International Workshop on Automated and verifiable Software sYstem DEvelopment (ASYDE'20)

Sept 15th, 2020

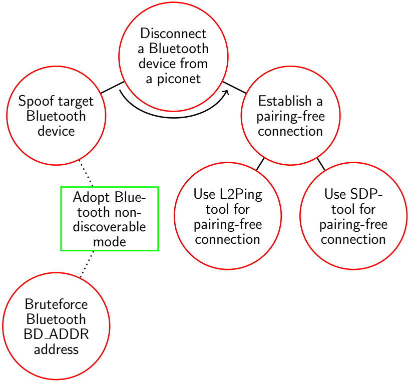
Outline

Why countermeasures modeling?

1. Modern systems: complex, heterogeneous, continuous, with multi disciplinary-aspects
2. System's modeling is challenging: scalable, flexible, user friendly, well founded semantics
3. Attacks are more sophisticated, complex, different nature at different level
4. Countermeasures have to deal with attack attempts and the system's resiliency

ADTrees

1. Graphical formalism representing attacks scenario
2. Extending Bruce Schneier's attack-trees by adding countermeasures
3. Allow a logical representation of attack scenarios along with their corresponding countermeasures in a user-friendly way
4. Quantitatively and qualitatively asses the security



Bluecutting attack

ADTrees Formalism

Definition 1

ADTrees are defined by means of an abstract syntax called ADTerms typed-terms over the signature $\Sigma = (\mathbb{S}, \mathbb{F})$, where:

- $\mathbb{S} = \{p, o\}$ is the set of types of players.
- $\mathbb{F} = \{(\vee_k^p)_{k\in\mathbb{N}}, (\wedge_k^p)_{k\in\mathbb{N}}, (\overrightarrow{\wedge}_k^p)_{k\in\mathbb{N}}, (\widetilde{\vee}_k^p)_{k\in\mathbb{N}}, (\vee_k^o)_{k\in\mathbb{N}}, (\wedge_k^o)_{k\in\mathbb{N}}, (\overrightarrow{\wedge}_k^o)_{k\in\mathbb{N}}, (\widetilde{\vee}_k^o)_{k\in\mathbb{N}}, c^p, c^o\} \cup \mathbb{B}^p \cup \mathbb{B}^o$ is a set of function symbols.

Definition 2

ADTrees are closed-terms over the signature $\Sigma = (\mathbb{S}, \mathbb{F})$, and generated by the following grammar, where $b^s \in \mathbb{B}$ and $s \in \mathbb{S}$:

$$t ::= b^s \mid \vee^s(t, \ldots, t) \mid \wedge^s(t, \ldots, t) \mid \overrightarrow{\wedge}^s(t, ..., t) \mid \widetilde{\vee}^s(t, ..., t) \mid c^s(t, t)$$

ADTrees Formalism

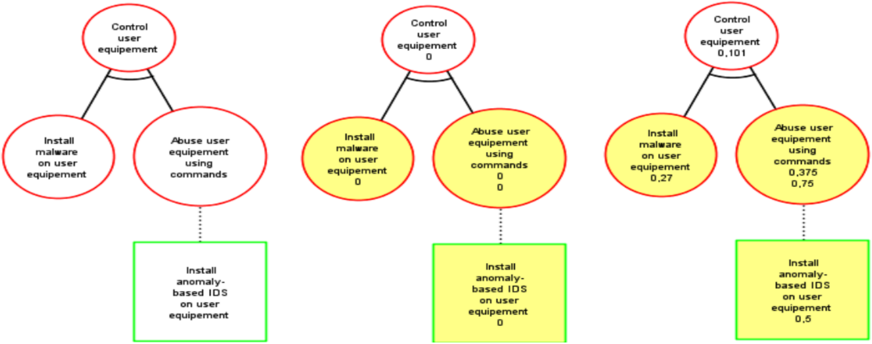
Examples of ADTerms:

- $t_0 = b_0^p$ (basic event)
- $t_1 = \vee^p(b_1^p, t_0)$ (Disjunctive refinement)
- $t_2 = \wedge^p(t1, b_2^p)$ (Conjunctive refinement)
- $t_3 = \overrightarrow{\wedge}^p(t_2, b_3^p, b_4^p)$ (Sequential conjunctive refinement)
- $t_4 = \widetilde{\vee}^p(t_3, b_5^p, b_6^p)$ (Parallel disjunctive refinements)
- $t_5 = c^o(b_0^o, t_4)$ (Countermeasure)

Where $b_{i \in \mathbb{N}} \in \mathbb{B}$ are atomic actions, and $s \in \{o, p\}$

ADTree Evaluation

ADTree follows bottom-up procedure to assess a set of measures: Probability, cost, and time



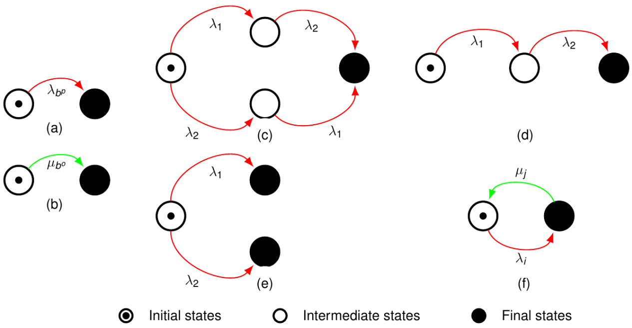
ADTree (Toward a new semantics)

1. Denotational semantics (e.g., propositional, mutiset, and De Morgan Lattice) $\longrightarrow$ hard to specify action ordering
2. Bottom-up procedure works only for independent actions
3. Bayesian networks based was time and memory consuming, and error prone

   $\longrightarrow$ **we need to develop a new semantics for ADTree**
4. The new semantics should allow dependent events to occur, and provide modelling capabilities for defenses in a more realistic way. It should also provide a continuous-time analysis method for ADTree evaluation

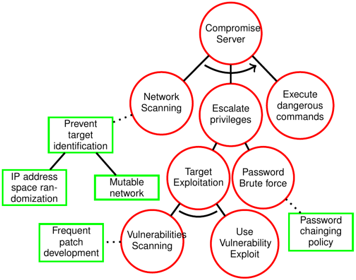
ADTree (Toward a new semantics)

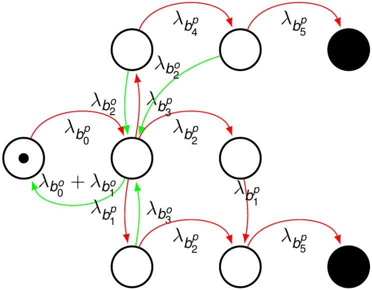
- Continuous Time Markov Chains were proposed as a new semantics for ADTrees
- Attack and defense nodes were modelled using transitions driven by an exponential distribution



ADTree(Toward a new semantics)

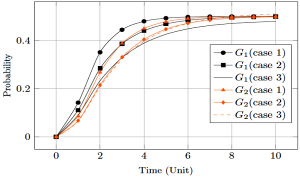
Enumerated-CTM framework transforms an ADTree into a CTMC

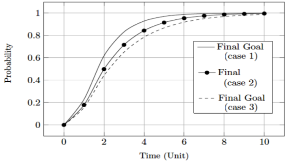


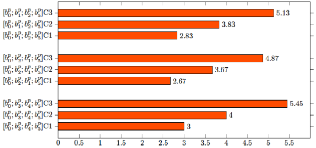


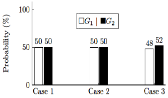
ADTree (From ADTrees to CTMCs)

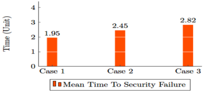
Using the analytical approach of CTMCs, we can evaluate several attributes, and perform a continuous analysis by the use of the Cumulative Distribution Function.





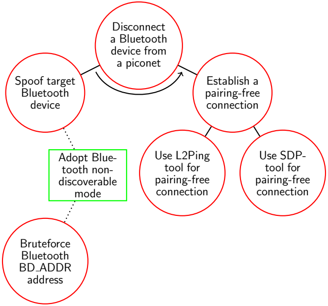


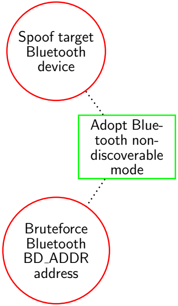




Cascaded-Countermeasure

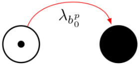
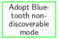
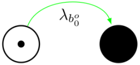
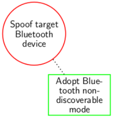
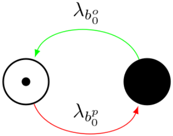
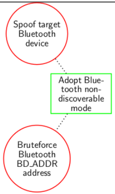
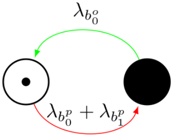
The proposed CTMC model does not allow a precise modeling for the countermeasure, in particular, when Cascaded-countermeasures occur





We have observed that when cascaded-countermeasures occur, the proposed transformation (CTMC-semantic) is not precise and complete.

ADTree (From ADTrees to CTMCs)

<table>
<tr><th>ADTerm</th><th>ADTree</th><th>CTMC</th></tr>
<tr><td>$b_0^p \in \mathbb{B}^p$</td><td>
</td><td>
</td></tr>
<tr><td>$b_0^o \in \mathbb{B}^o$</td><td>
</td><td>
</td></tr>
<tr><td>$t = c^p(b_0^p, b_0^o)$</td><td>
</td><td>
</td></tr>
<tr><td>$t = c^p(b_0^p, c^o(b_0^o, b_1^p))$</td><td>
</td><td>
</td></tr>
</table>

Tokenized Continuous Time Markov Chains

Definition 3

Colored indexed-tokens are indexed elements of a set $\mathbb{C} = \mathbb{C}^p \bigcup \mathbb{C}^o$ that can take one of the two colors: red ($\bullet$) or green ($\bullet$). We use the red color ($\bullet$) to refer to the proponent and the green color ($\bullet$) for the opponent.

Definition 4

Let $\mathbb{B}$ be the set of *basic actions* and let $\mathbb{C} = \{\bullet_0, \ldots, \bullet_n, \bullet_0, \ldots., \bullet_m\}$ be the *set of colored indexed-tokens*, for $n, m \in \mathbb{N}$. Then, an *action-coloring* is a function $\sigma : \mathbb{B} \to \mathbb{C}$, which associates for each basic action $b \in \mathbb{B}$ a singleton of a colored indexed-token, i.e., $\{\bullet\}$ or $\{\bullet\}$.

Example. If we consider the ADTree $t = c^p(b_0^p, b_0^o)$, where the attacker is the proponent and the defender is the opponent, we can write:

$$\sigma(b_0^p) = \{\bullet_0\}, \sigma(b_1^p) = \{\bullet_1\}, \text{and } \sigma(b_0^o) = \{\bullet_0\}.$$

Tokenized Continuous Time Markov Chains

We can associate to each state of the CTMC, an ordered set of colored indexed-tokens to indicate which action/actions has/have been successfully achieved at that state.

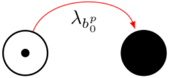
$\longrightarrow$ determine whether the proponent or the opponent is the vanquisher at a specific state of the system.

Definition 5

Let S be *the set of states* of a given enumerated-CTMC, and let $\mathbb{C}$ be a *set of colored indexed-tokens*. Then, a state-coloring is a function $\tau\colon S \to \mathcal{P}(\mathbb{C})$, which associates for each state $s \in S$ an ordered set of colored indexed-tokens from $\mathcal{P}(\mathbb{C})$.

Tokenized Continuous Time Markov Chains

Example. a single attack action $b_0^p \in \mathbb{B}^p$

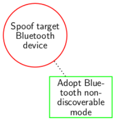
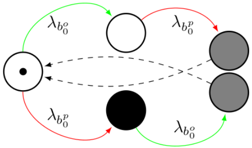
ADTerm	ADTree	CTMC
$b_0^p \in \mathbb{B}^p$		

The colored indexed-token associated to the states of its CTMC is $\tau(S_0^{b_0^p}) = \{\} = \emptyset$ and $\tau(S_*^{b_0^p}) = \sigma(b_0^p) = \{\bullet_0\}$.

An enumerated-CTMC associated with a states-coloring function is called a tokenized-CTMC, or T-CTMC for short.

Improving the CTMC-model for countermeasures

To have a more precise and complete model to represent countermeasures, we slightly modify the existing CTMC-model for the countermeasure in such a way that we allow the countermeasure and the countered action to evolve in parallel

ADTerm	ADTree	CTMC
$t = r^p(b_0^p, b_0^o)$		

We formally redefined the tokenized-CTMC definition so that the new model for countermeasures is adopted

Tokenized Continuous Time Markov Chains

Definition 6

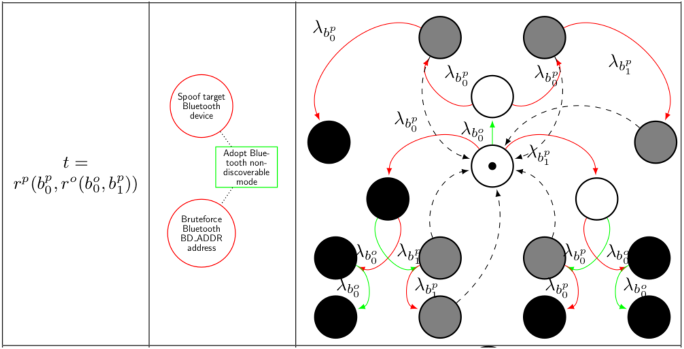
A *colored-indexed token evaluator* is a function $\xi \colon \mathcal{P}(\mathbb{C}) \to \mathcal{P}(\mathbb{C})$, which selects a subset of colored indexed-tokens from a larger set of colored indexed-tokens w.r.t. the following rules, where $c_{i\in\mathbb{N}}, c_{j\in\mathbb{N}}, c_{k\in\mathbb{N}} \in \mathbb{C}$ are *colored indexed-tokens*, and $c_{i\in\mathbb{N}}$ *the selected ones*:

- $\nexists j, k \in \mathbb{N} \mid r^s(\sigma^{-1}(c_i), \sigma^{-1}(c_j)) \wedge r^s(\sigma^{-1}(c_j), \sigma^{-1}(c_k))$.
- $\exists j, k \in \mathbb{N}, j > k > i \mid r^s(\sigma^{-1}(c_i), \sigma^{-1}(c_j)) \wedge r^s(\sigma^{-1}(c_j), \sigma^{-1}(c_k))$.
- $\exists j, k \in \mathbb{N}, k > i > j \mid r^s(\sigma^{-1}(c_i), \sigma^{-1}(c_j)) \wedge r^s(\sigma^{-1}(c_j), \sigma^{-1}(c_k))$.
- $\exists j, k \in \mathbb{N}, i > j, i > k \mid r^s(\sigma^{-1}(c_i), \sigma^{-1}(c_j)) \wedge r^s(\sigma^{-1}(c_j), \sigma^{-1}(c_k))$.

Example. If we consider the ADTree $t = r^p(b_0^p, r^o(b_0^o, b_1^p))$, where the attacker is the proponent and the defender is the opponent, then for a given set of colored indexed-tokens $\{\textcolor{green}{\bullet}_0, \textcolor{red}{\bullet}_1, \textcolor{red}{\bullet}_0\}$ associated to a given state $s' \in S$, we have $\xi(\{\textcolor{green}{\bullet}_0, \textcolor{red}{\bullet}_1, \textcolor{red}{\bullet}_0\}) = \{\textcolor{red}{\bullet}_1, \textcolor{red}{\bullet}_0\}$.

Tokenized Continuous Time Markov Chains

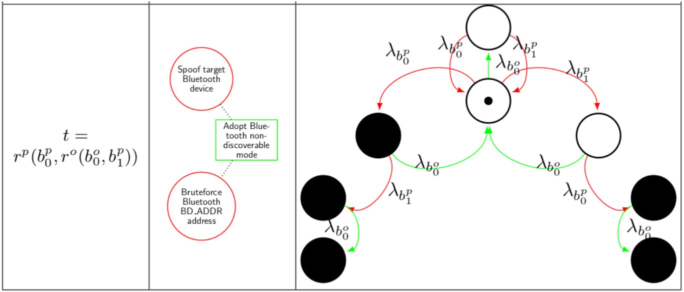
By applying the new formalism of the Tokenized-CTMC, we construct the T-CTMC for the cascaded-countermeasure $t = r^p(b_0^p, r^o(b_0^o, b_1^p))$:



The gray states are called, dump states. They are states where neither the proponent nor the opponent is the vanquisher, just like the initial state. They are eliminated from the CTMC for optimization.

Tokenized Continuous Time Markov Chains

After eliminating the dump sates, we can obtain a lighter version of the T-CTMC.



This T-CTMC can be evaluated using the traditional CTMC analytical approaches to conduct quantitative analysis.

Conclusion

In this work, we have proposed a new CTMC-model to represent countermeasure in ADTrees

We have introduced the notion of tokenized-CTMC to precisely represent countermeasures and handle the order in which actions are occurring
The new model allows a correct and complete modelling of ADTrees using CTMC, in particular, when cascaded-countermeasures are present

Nevertheless, the approach still need some improvements, in particular:

- State explosion handling
- Evaluation of the model for different case studies
- Integration to ADTool for an upgraded version

Thanks you,
Questions!