Attack Trees

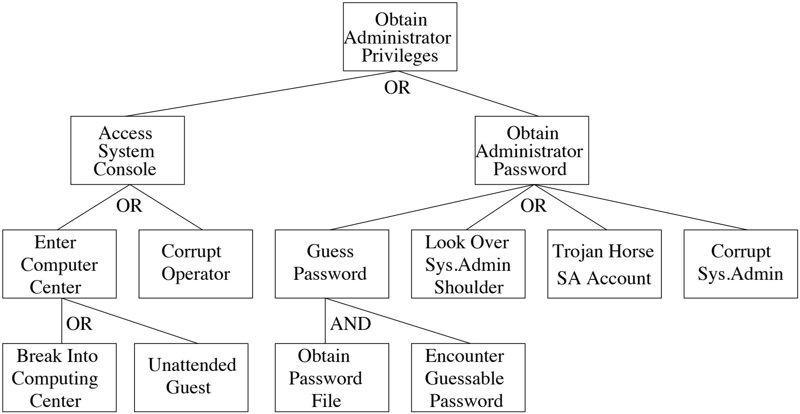
Models and Computation

Jan Willemson, Aivo Jürgenson, Margus Niitsoo

Cybernetica, Estonia
http://www.cybernetica.eu/

January 19th, 2010
University of Luxembourg

An Attack Tree



An Attack RDAG

Brief History

Hierarchical approach to security evaluation:

- Fault trees (Vesely, Goldberg, Roberts, Haasl, 1981)
- Threat logic trees (Weiss, 1991)
- Attack trees (Schneier, 1999)

- Foundations of Attack Trees (Mauw & Oostdijk, 2005)
- Multi-parameter attack trees (Buldas *et al.*, 2006)

Our Papers

- Buldas, Laud, Priisalu, Saarepera, Willemson, Rational Choice of Security Measures via Multi-Parameter Attack Trees, CRITIS 2006
- Jürgenson, Willemson, Processing Multi-parameter Attacktrees with Estimated Parameter Values, IWSEC 2007
- Jürgenson, Willemson, Computing Exact Outcomes of Multi-parameter Attack Trees, OTM 2008, IS 2008
- Jürgenson, Willemson, Serial Model for Attack Tree Computations, ICISC 2009
- Jürgenson, Willemson, On Fast and Approximate Attack Tree Computations, submitted to ISPEC 2010
- Niitsoo, Finding the Optimal Behavior for Adaptive Attack trees, submitted to ???

From Qualitative to Quantitative Analysis

Once an attack tree is complete, one can . . .

- . . . use it for qualitative description of attack scenarios
  - An Attack Tree for the Border Gateway Protocol, IETF draft, 2004

From Qualitative to Quantitative Analysis

Once an attack tree is complete, one can ...

- ... use it for qualitative description of attack scenarios
  - An Attack Tree for the Border Gateway Protocol, IETF draft, 2004
- ... analyze some property of the attacks (cost, feasibility, skill level required, etc.)
  - Schneier, 1999
  - Mauw&Oostdijk, 2005

From Qualitative to Quantitative Analysis

Once an attack tree is complete, one can . . .

- . . . use it for qualitative description of attack scenarios
  - An Attack Tree for the Border Gateway Protocol, IETF draft, 2004
- . . . analyze some property of the attacks (cost, feasibility, skill level required, etc.)
  - Schneier, 1999
  - Mauw&Oostdijk, 2005
- . . . try to find the attack most profitable for the attacker
  - Buldas *et al.*, 2006

Rational Attacker Paradigm

In order to find the best attack, we must assume some kind of rationality of the attacker

- The original model of Buldas *et al.* assumes that the attacker is a fully rational utility maximizer

Rational Attacker Paradigm

In order to find the best attack, we must assume some kind of rationality of the attacker

- The original model of Buldas *et al.* assumes that the attacker is a fully rational utility maximizer
- Jürgenson&Willemson, 2009, builds on another framework:
  - The attacker tries to
    - first, maximize success probability
    - second, achieve the best possible outcome
  - Hence, a certain form of irrational behavior is obtained
    - This is the first known treatment of irrational attacks using quantitative methods

Rational Attacker Paradigm

In order to find the best attack, we must assume some kind of rationality of the attacker

- The original model of Buldas *et al.* assumes that the attacker is a fully rational utility maximizer
- Jürgenson&Willemson, 2009, builds on another framework:
  - The attacker tries to
    - first, maximize success probability
    - second, achieve the best possible outcome
  - Hence, a certain form of irrational behavior is obtained
    - This is the first known treatment of irrational attacks using quantitative methods
- Niitsoo, 2010, has shown how to apply classical decision theory to attack tree computations

Parallel vs. Serial Approach

- Virtually all the present models of attack trees disregard the possible order of elementary attacks
  - Schneier, 1999
  - Mauw&Oostdijk, 2005
  - Buldas *et al.*, 2006
- This restriction is unrealistic
  - The attacker can use the knowledge concerning success/failure of some elementary attacks to decide, which attacks to skip or try next
  - Intuitively, this will allow the attacker to avoid hopeless branches, thus reducing the potential penalties and increasing the expected outcome

Flavors of the Serial Model

- Blocking vs. non-blocking
  - In practice, there exist elementary attacks, failed attempt of which blocks the execution of the whole tree, e.g. due to imprisonment of the attacker

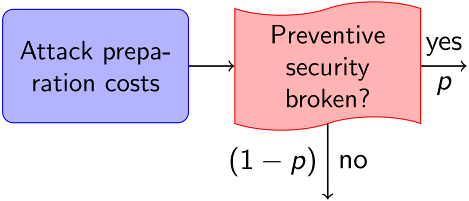
Flavors of the Serial Model

- Blocking vs. non-blocking
  - In practice, there exist elementary attacks, failed attempt of which blocks the execution of the whole tree, e.g. due to imprisonment of the attacker
- Fully adaptive vs. semi-adaptive
  - In reality, the attacker can freely choose the order of the next elementary attacks based on the results of already tried ones
  - From theoretical viewpoint, this gives a superexponential explosion
  - Hence, for an intermediate step we may limit ourselves to the model, where the attacker
    - Fixes the order of the elementary attacks in advance
    - Is only allowed to skip some of them or stop attacking altogether

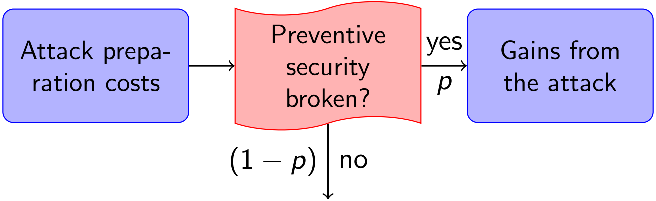
The Attack Game (Buldas *et al.*, 2006)

Attack preparation costs

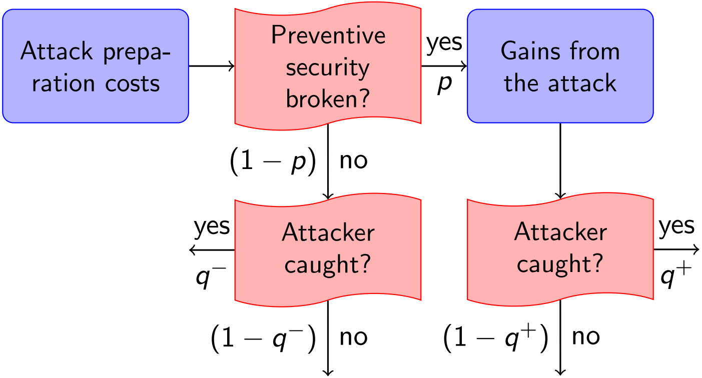
The Attack Game (Buldas *et al.*, 2006)



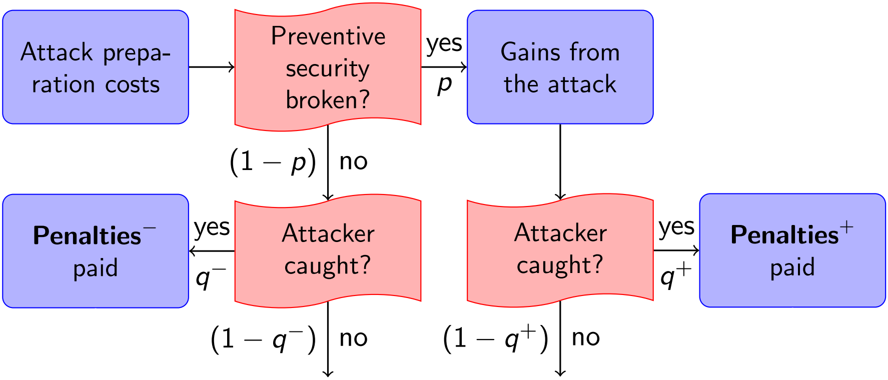
The Attack Game (Buldas *et al.*, 2006)



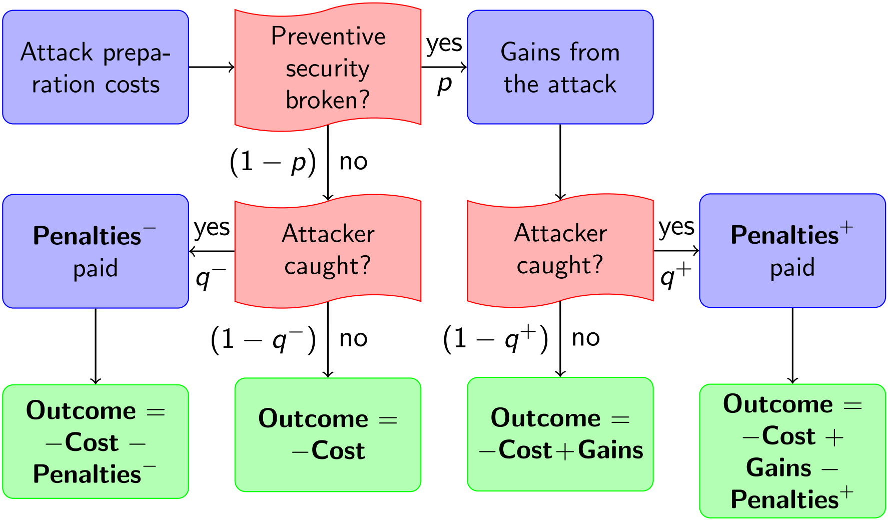
The Attack Game (Buldas *et al.*, 2006)



The Attack Game (Buldas *et al.*, 2006)



The Attack Game (Buldas *et al.*, 2006)



Multi-parameter Attack Trees (Buldas *et al.*, 2006)

- **Gains** – value gained from the successful attack
- $\mathbf{Cost}_i$ – cost of the elementary attack, p_i – success probability
- $\pi_i^- = q^- \cdot \mathbf{Penalty}^-$ – expected penalty, unsuccessful attack
- $\pi_i^+ = q^+ \cdot \mathbf{Penalty}^+$ – expected penalty, successful attack

$$\mathbf{Outcome}_i = p_i \cdot \mathbf{Gains} - \mathbf{Cost}_i - p_i \cdot \pi_i^+ - (1 - p_i) \cdot \pi_i^-$$

Multi-parameter Attack Trees (Buldas *et al.*, 2006)

- **Gains** – value gained from the successful attack
- $\textbf{Cost}_i$ – cost of the elementary attack, p_i – success probability
- $\pi_i^- = q^- \cdot \textbf{Penalty}^-$ – expected penalty, unsuccessful attack
- $\pi_i^+ = q^+ \cdot \textbf{Penalty}^+$ – expected penalty, successful attack

$$\textbf{Outcome}_i = p_i \cdot \textbf{Gains} - \textbf{Cost}_i - p_i \cdot \pi_i^+ - (1 - p_i) \cdot \pi_i^-$$

For an OR-node:

$$(\textbf{Cost}, p, \pi^+, \pi^-) = \begin{cases} (\textbf{Cost}_1, p_1, \pi_1^+, \pi_1^-), \text{if } \textbf{Outcome}_1 > \textbf{Outcome}_2 \\ (\textbf{Cost}_2, p_2, \pi_2^+, \pi_2^-), \text{if } \textbf{Outcome}_1 \leq \textbf{Outcome}_2 \end{cases}$$

Multi-parameter Attack Trees (Buldas *et al.*, 2006)

- **Gains** – value gained from the successful attack
- $\mathbf{Cost}_i$ – cost of the elementary attack, p_i – success probability
- $\pi_i^- = q^- \cdot \mathbf{Penalty}^-$ – expected penalty, unsuccessful attack
- $\pi_i^+ = q^+ \cdot \mathbf{Penalty}^+$ – expected penalty, successful attack

$$\mathbf{Outcome}_i = p_i \cdot \mathbf{Gains} - \mathbf{Cost}_i - p_i \cdot \pi_i^+ - (1 - p_i) \cdot \pi_i^-$$

For an OR-node:

$$(\mathbf{Cost}, p, \pi^+, \pi^-) = \begin{cases} (\mathbf{Cost}_1, p_1, \pi_1^+, \pi_1^-), \text{if } \mathbf{Outcome}_1 > \mathbf{Outcome}_2 \\ (\mathbf{Cost}_2, p_2, \pi_2^+, \pi_2^-), \text{if } \mathbf{Outcome}_1 \leq \mathbf{Outcome}_2 \end{cases}$$

For an AND-node:

$$\begin{aligned} \mathbf{Cost} &= \mathbf{Cost}_1 + \mathbf{Cost}_2, \quad p = p_1 \cdot p_2, \quad \pi^+ = \pi_1^+ + \pi_2^+, \\ \pi^- &= \frac{p_1(1-p_2)(\pi_1^+ + \pi_2^-) + (1-p_1)p_2(\pi_1^- + \pi_2^+)}{1 - p_1 p_2} + \\ &\quad + \frac{(1-p_1)(1-p_2)(\pi_1^- + \pi_2^-)}{1 - p_1 p_2} \end{aligned}$$

Buldas *et al.*, 2006: pros and cons

Pros:

- The semantics uses several intuitively relevant parameters
- The semantics is very fast, works by one tree traversal in time $O(n)$

Buldas *et al.*, 2006: pros and cons

Pros:

- The semantics uses several intuitively relevant parameters
- The semantics is very fast, works by one tree traversal in time $O(n)$

Cons:

- In each OR-node, **Outcome** needs to be computed, which needs **Gains** for each OR-node, but **Gains** only has a meaning globally
- The model (as most of the other previous models) assumes that exactly one descendant is picked in an OR-node
- The model is inconsistent with Mauw&Oostdijk 2005

Jürgenson & Willemson, 2008

$\mathcal{F}$ — Boolean formula corresponding to the attack tree

$\mathcal{X}$ — set of elementary attacks

σ — attack suite, satisfying the root node of the attack tree

Jürgenson & Willemson, 2008

$\mathcal{F}$ — Boolean formula corresponding to the attack tree

$\mathcal{X}$ — set of elementary attacks

σ — attack suite, satisfying the root node of the attack tree

$$\textbf{Outcome} = \max_{\sigma}\{\textbf{Outcome}_\sigma : \sigma \subseteq \mathcal{X},\ \mathcal{F}(\sigma := \text{true}) = \text{true}\}$$

$$\textbf{Outcome}_\sigma = p_\sigma \cdot \textbf{Gains} - \sum_{X_i \in \sigma} \textbf{Expenses}_i$$

$$\textbf{Expenses}_i = \textbf{Cost}_i + p_i \cdot \pi_i^+ + (1 - p_i) \cdot \pi_i^-$$

$$p_\sigma = \sum_{\substack{\rho \subseteq \sigma \\ \mathcal{F}(\rho := \text{true}) = \text{true}}} \prod_{X_i \in \rho} p_i \prod_{X_j \in \sigma \setminus \rho} (1 - p_j)$$

Implementation & Results

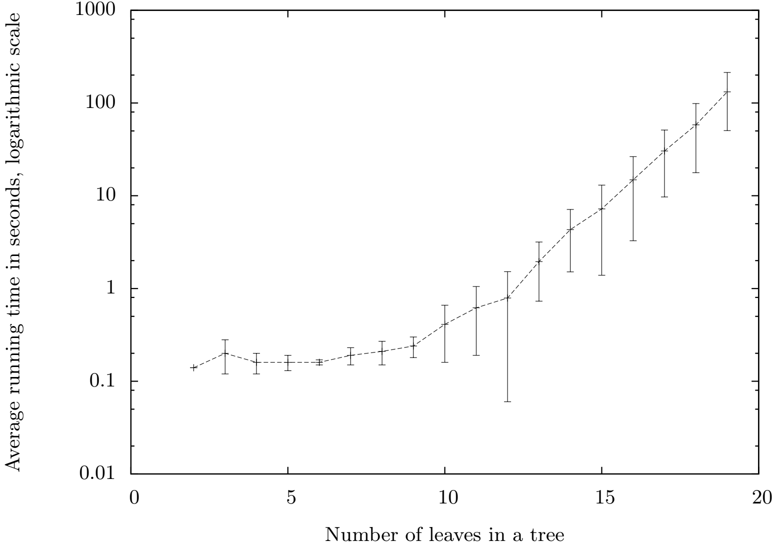
- Implemented in Perl programming language, using terribly inefficient data structures
- p_σ can be computed in linear time
  - Going through potentially all the subsets of $\mathcal{X}$ still remains exponential, of course
- Using a modified DPLL algorithm for finding all such attack suites, which satisfy the attack tree
- Theorem: We don't need to consider AND nodes, where some child node is not satisfied

Implementation & Results

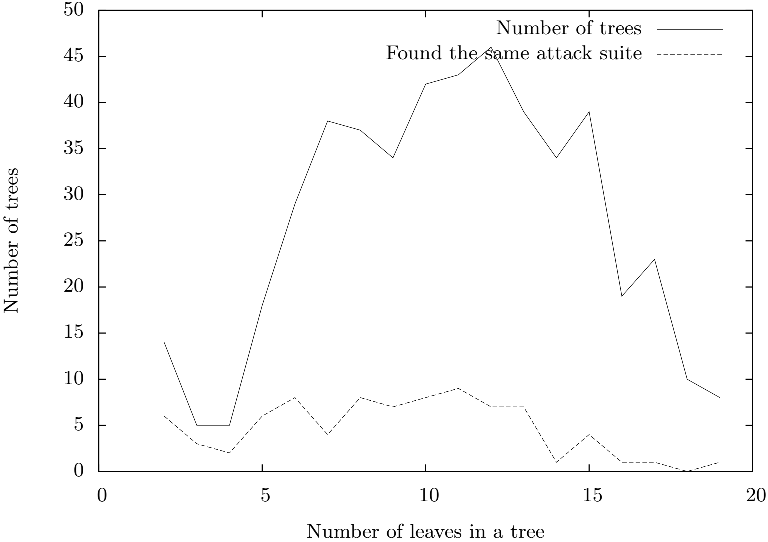
- Implemented in Perl programming language, using terribly inefficient data structures
- p_σ can be computed in linear time
  - Going through potentially all the subsets of $\mathcal{X}$ still remains exponential, of course
- Using a modified DPLL algorithm for finding all such attack suites, which satisfy the attack tree
- Theorem: We don't need to consider AND nodes, where some child node is not satisfied

- $\mathbf{Outcome}_{\mathrm{JW08}} \geq \mathbf{Outcome}_{\mathrm{B+06}}$
- If $T_1 \equiv T_2$ then $\mathbf{Outcome}(T_1) = \mathbf{Outcome}(T_2)$

Performance



Comparison with Buldas *et al.*, 2006

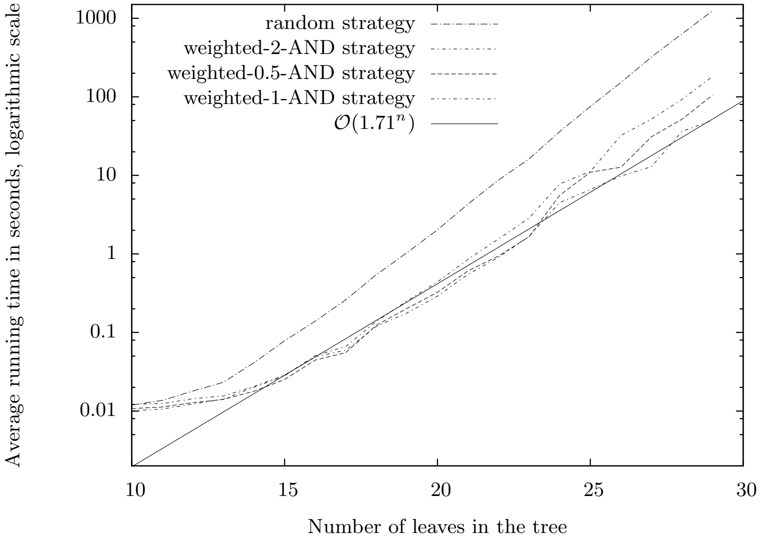


Jürgenson & Willemson, 2010

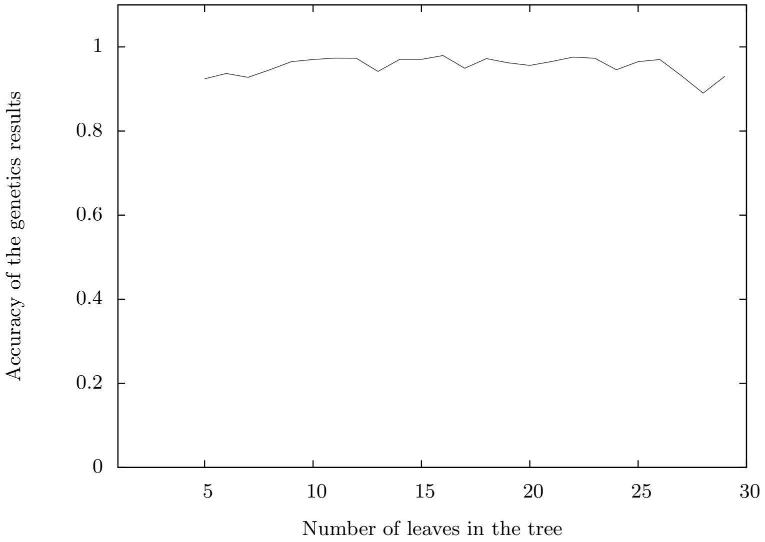
Reimplementation of Jürgenson & Willemson, 2008

- C++ instead of Perl
- Removing unnecessary DPLL overhead (e.g. transformation to CNF)
- Bit vectors instead of classes representing sets of subsets
- Catching true&false as soon as it occurs
- Implementing better strategies for choosing undefined literals
  - Most-AND and Weighted-AND
  - Heuristic complexity of the resulting algorithm: $O(1.71^n)$
    - The best $\#SAT$-solver works in time $O(1.6423^n)$
- Fast approximation using a custom genetic algorithm
  - At least 89% accuracy within 2 seconds for the trees with less than 30 leaves

Comparing Strategies



Accuracy of the Genetic Algorithm

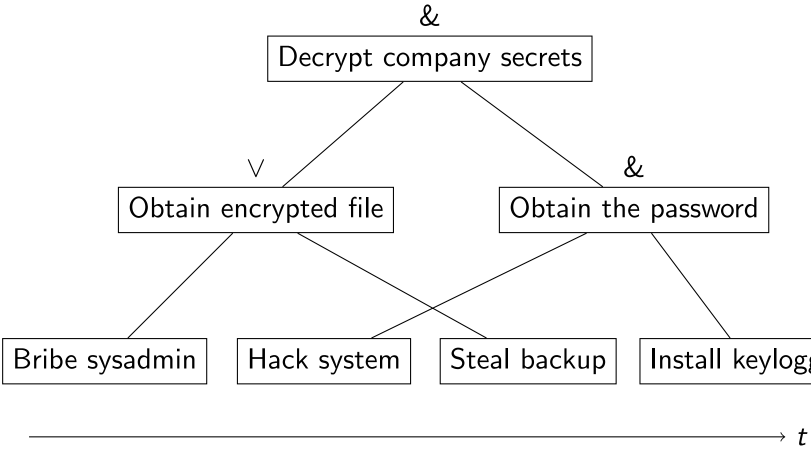


Jürgenson & Willemson, 2009

Introduction of the serial model

- Semi-adaptive, non-blocking case, i.e.
  - The attacker fixes the order of the elementary attacks in advance
  - He is allowed to skip the elementary attacks that have become useless
  - No failure blocks the entire execution

Attacker's Choices



Outcome in the Serial Model (I)

The expected outcome of the attack based on permutation α is

$$\mathbf{Outcome}_\alpha = p_\alpha \cdot \mathbf{Gains} - \sum_{X_i \in \mathcal{X}} p_{\alpha,i} \cdot \mathbf{Expenses}_i \,,$$

where p_α is the success probability of the primary threat and $p_{\alpha,i}$ denotes the probability that the node X_i

Outcome in the Serial Model (I)

The expected outcome of the attack based on permutation α is

$$\mathbf{Outcome}_\alpha = p_\alpha \cdot \mathbf{Gains} - \sum_{X_i \in \mathcal{X}} p_{\alpha,i} \cdot \mathbf{Expenses}_i \,,$$

where p_α is the success probability of the primary threat and $p_{\alpha,i}$ denotes the probability that the node X_i

Theorem

Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two monotone Boolean formulae such that $\mathcal{F}_1 \equiv \mathcal{F}_2$, and let $\mathbf{Outcome}^1_\alpha$ and $\mathbf{Outcome}^2_\alpha$ be the expected outcomes obtained running the algorithm on the corresponding formulae using the leaf set permutation α. Then

$$\mathbf{Outcome}^1_\alpha = \mathbf{Outcome}^2_\alpha \,.$$

Outcome in the Serial Model (II)

Theorem

We have

$$\mathbf{Outcome}_{\mathrm{JW09}} \geq \mathbf{Outcome}_{\mathrm{JW08}} .$$

If for all the elementary attacks X_i ($i = 1, \ldots, n$) one also has $\mathbf{Expenses}_i > 0$, *then strict inequality holds in the above inequality.*

Outcome in the Serial Model (II)

Theorem

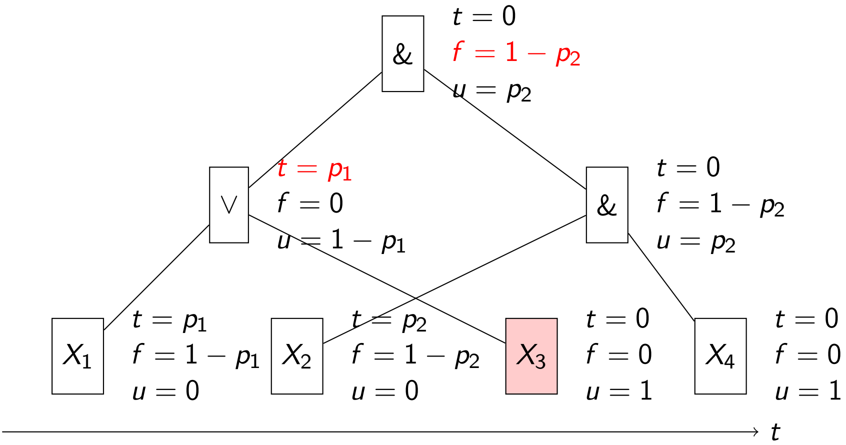
We have

$$\textbf{Outcome}_{\mathrm{JW09}} \geq \textbf{Outcome}_{\mathrm{JW08}}\,.$$

If for all the elementary attacks X_i ($i = 1, \ldots, n$) one also has $\textbf{Expenses}_i > 0$, *then strict inequality holds in the above inequality.*

- The naïve algorithm for computing the attacker's outcome is average-case exponential in the number of leaves n
- We propose an efficient algorithm with complexity $O(n^2)$
  - Recall, need only the quantities p_α and $p_{\alpha,i}$

The Algorithm



$$p_{\alpha,3} = (1 - p_1) \cdot (1 - (1 - p_2))$$

Sequential model revised

- Jürgenson&Willemson, 2009, builds on another framework:
  - The attacker tries to
    - first, maximize success probability
    - second, achieve the best possible outcome
  - Hence, a certain form of irrational behavior is obtained

Sequential model revised

- Jürgenson&Willemson, 2009, builds on another framework:
  - The attacker tries to
    - first, maximize success probability
    - second, achieve the best possible outcome
  - Hence, a certain form of irrational behavior is obtained
- Niitsoo, 2010 analyzes the rational case
  - Builds on classical decision theory
  - Attacks can be skipped if they are too expensive
  - Otherwise same as JW09
    - Order of attacks fixed before the attack
    - Full information about the past

Sequential model computation

- Decision tree optimization algorithm
  - Decision trees usually exponential in general
- Attack trees provide for a simple structure
  - We do not optimize Trees but BDD-s
- Non-crossing orders optimized in $O(n)$ time.
  - Modeling goal-oriented behavior
  - Optimal non-crossing order for JW10 can be found in $O(n \lg n)$ time (Niitsoo, 2010)

Fully rational model

- Pros:
  - Fully rational behavior (easy to justify)
  - Optimal subset found automatically
  - Highest expected utility of all models to date
  - Efficient $O(n)$ computation for some orders
  - Highly extensible:
    - Blocking case (even partial blocking)
    - Bribes and uncertainty
    - Intermediate payments
- Cons:
  - Computation exponential for some orders
  - Still only semi-adaptive
  - Conventional