Computing Exact Outcomes of Multi-Parameter Attack Trees

Aivo Jürgenson[2,3] Jan Willemson[1]

[1]Cybernetica, Tartu, Estonia

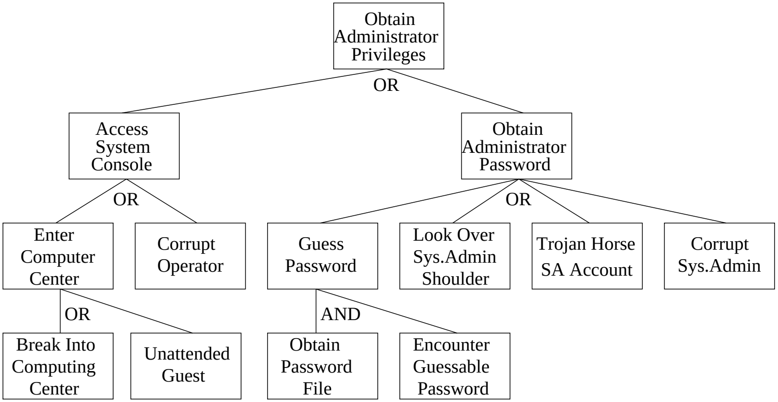
[2]Tallinn University of Technology, Tallinn, Estonia

[3]Elion Enterprises Ltd, Tallinn, Estonia

3rd October 2008

Outline

Attack Trees[1]



[1] J. D. Weiss 1991, Bruce Schneier 1999

Multi-parameter Attack Trees[2]

- Cost_i – the cost of the elementary attack, p_i – success probability
- π_i^- – the expected penalty in case the attack was unsuccessful
- π_i^+ – the expected penalty in case the attack was successful

[2] Buldas, Laud, Priisalu, Saarepera, Willemson, 2006

Multi-parameter Attack Trees[2]

- Cost_i – the cost of the elementary attack, p_i – success probability
- π_i^- – the expected penalty in case the attack was unsuccessful
- π_i^+ – the expected penalty in case the attack was successful

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

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

[2] Buldas, Laud, Priisalu, Saarepera, Willemson, 2006

Multi-parameter Attack Trees[2]

- Cost_i – the cost of the elementary attack, p_i – success probability
- π_i^- – the expected penalty in case the attack was unsuccessful
- π_i^+ – the expected penalty in case the attack was successful

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

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

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

[2] Buldas, Laud, Priisalu, Saarepera, Willemson, 2006

Foundations of Attack Trees[3]

1. Formal foundations of attack trees. Definition of attack components, attacks, attack suites, attack trees.
2. Transformations of attack trees. Transformations are proved to be sound and complete.

[3] Mauw and Oostdjik, 2005

Foundations of Attack Trees[3]

1. Formal foundations of attack trees. Definition of attack components, attacks, attack suites, attack trees.
2. Transformations of attack trees. Transformations are proved to be sound and complete.
3. Rules for calculating value of attack tree.
    1. Conjunctive combinators calculate the value of attack from attack components.
    2. Disjunctive combinators calculate the value of attack suite from attacks.
    3. When certain algebraic properties hold, we say that we have *distributive attributive domain*.

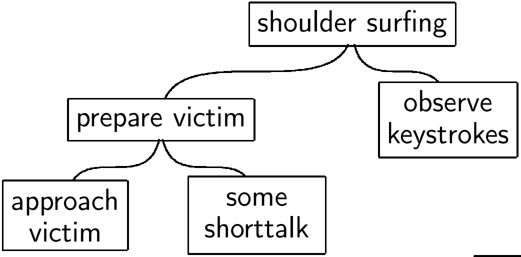
[3] Mauw and Oostdjik, 2005

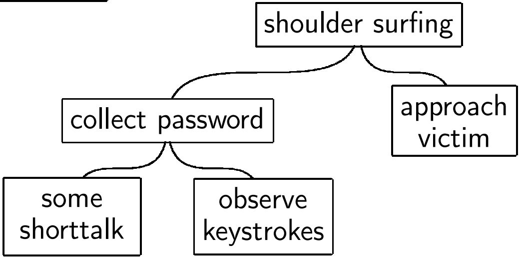
Foundations of Attack Trees[3]

1. Formal foundations of attack trees. Definition of attack components, attacks, attack suites, attack trees.
2. Transformations of attack trees. Transformations are proved to be sound and complete.
3. Rules for calculating value of attack tree.
   1. Conjunctive combinators calculate the value of attack from attack components.
   2. Disjunctive combinators calculate the value of attack suite from attacks.
   3. When certain algebraic properties hold, we say that we have *distributive attributive domain*.
4. For example:
   1. $(\mathbb{N}, \min, +)$ could be interpreted as "cost of the cheapest attack".
   2. $(\mathbb{B}, \wedge, \vee)$ as "is the attack possible to complete".

[3] Mauw and Oostdjik, 2005

Equivalent attack trees





Equivalent attack trees

- $T_1 = A \vee (B \& C)$
- $T_2 = (A \vee B) \& (A \vee C)$
- $\text{Gains} = 10000$
- $p_A = 0.1,\ p_B = 0.5,\ p_C = 0.4$
- $\text{Expenses}_A = 1000,\ \text{Expenses}_B = 1500,\ \text{Expenses}_C = 1000$
- $\text{Outcome}_{\mathrm{T_1}} = 8000$
- $\text{Outcome}_{\mathrm{T_2}} = 6100$

Exact semantics

Exact semantics

$$\mathsf{Outcome} = \max\{\mathsf{Outcome}_\sigma : \sigma \subseteq \mathcal{X},\ \mathcal{F}(\sigma := \mathsf{true}) = \mathsf{true}\}\,. \tag{1}$$

Exact semantics

$$\mathsf{Outcome} = \max\{\mathsf{Outcome}_\sigma : \sigma \subseteq \mathcal{X},\ \mathcal{F}(\sigma := \mathsf{true}) = \mathsf{true}\}\,. \tag{1}$$

$$\mathsf{Outcome}_\sigma = p_\sigma \cdot \mathsf{Gains} - \sum_{X_i \in \sigma} \mathsf{Expenses}_i\,, \tag{2}$$

Exact semantics

$$\mathsf{Outcome} = \max\{\mathsf{Outcome}_\sigma : \sigma \subseteq \mathcal{X},\, \mathcal{F}(\sigma := \mathsf{true}) = \mathsf{true}\}\,. \tag{1}$$

$$\mathsf{Outcome}_\sigma = p_\sigma \cdot \mathsf{Gains} - \sum_{X_i \in \sigma} \mathsf{Expenses}_i\,, \tag{2}$$

$$p_\sigma = \sum_{\substack{\rho \subseteq \sigma \\ \mathcal{F}(\rho := \mathsf{true}) = \mathsf{true}}} \prod_{X_i \in \rho} p_i \prod_{X_j \in \sigma \setminus \rho} (1 - p_j)\,. \tag{3}$$

Exact Semantics: Example

- $T = (A \vee B) \& C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^+ = 1000$, $\pi^- = 1000$

Exact Semantics: Example

- $T = (A \vee B) \& C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^+ = 1000$, $\pi^- = 1000$
- $\sigma_1 = \{A, C\}, \sigma_2 = \{B, C\}, \sigma_3 = \{A, B, C\}$

Exact Semantics: Example

- $T = (A \vee B) \& C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^{+} = 1000$, $\pi^{-} = 1000$
- $\sigma_1 = \{A, C\}, \sigma_2 = \{B, C\}, \sigma_3 = \{A, B, C\}$
- $\mathrm{Outcome}_{\sigma_1} = \mathrm{Outcome}_{\sigma_2} = 4200$

Exact Semantics: Example

- $T = (A \vee B) \& C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^{+} = 1000$, $\pi^{-} = 1000$
- $\sigma_1 = \{A, C\}, \sigma_2 = \{B, C\}, \sigma_3 = \{A, B, C\}$
- $\text{Outcome}_{\sigma_1} = \text{Outcome}_{\sigma_2} = 4200$
- $\rho_1 = \{A, C\}$, $\rho_2 = \{B, C\}$, $\rho_3 = \{A, B, C\}$

Exact Semantics: Example

- $T = (A \vee B) \& C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^{+} = 1000$, $\pi^{-} = 1000$
- $\sigma_1 = \{A, C\}, \sigma_2 = \{B, C\}, \sigma_3 = \{A, B, C\}$
- $\text{Outcome}_{\sigma_1} = \text{Outcome}_{\sigma_2} = 4200$
- $\rho_1 = \{A, C\}$, $\rho_2 = \{B, C\}$, $\rho_3 = \{A, B, C\}$
- $p_{\sigma_3} = p_A p_B p_C + p_A p_C (1 - p_B) + p_B p_C (1 - p_A) = 0.768$

Exact Semantics: Example

- $T = (A \vee B)\&C$
- Gain $= 10000$
- $p = 0.8$, Cost $= 100$, $\pi^{+} = 1000$, $\pi^{-} = 1000$
- $\sigma_1 = \{A, C\}, \sigma_2 = \{B, C\}, \sigma_3 = \{A, B, C\}$
- $\text{Outcome}_{\sigma_1} = \text{Outcome}_{\sigma_2} = 4200$
- $\rho_1 = \{A, C\}$, $\rho_2 = \{B, C\}$, $\rho_3 = \{A, B, C\}$
- $p_{\sigma_3} = p_A p_B p_C + p_A p_C (1 - p_B) + p_B p_C (1 - p_A) = 0.768$
- $\text{Outcome}_{\sigma_3} = \text{Outcome}_T = 4380$

Realization

1. Using modified DPLL for finding all such attack suites, which satisfy the attack tree. Basically, finding all SAT solutions for a Boolean formula.

Realization

1. Using modified DPLL for finding all such attack suites, which satisfy the attack tree. Basically, finding all SAT solutions for a Boolean formula.
2. Using some optimizations and cutting of hopeless branches.
    1. Theorem: We don't need to consider AND nodes, where some subnode is not satisfied.

Realization

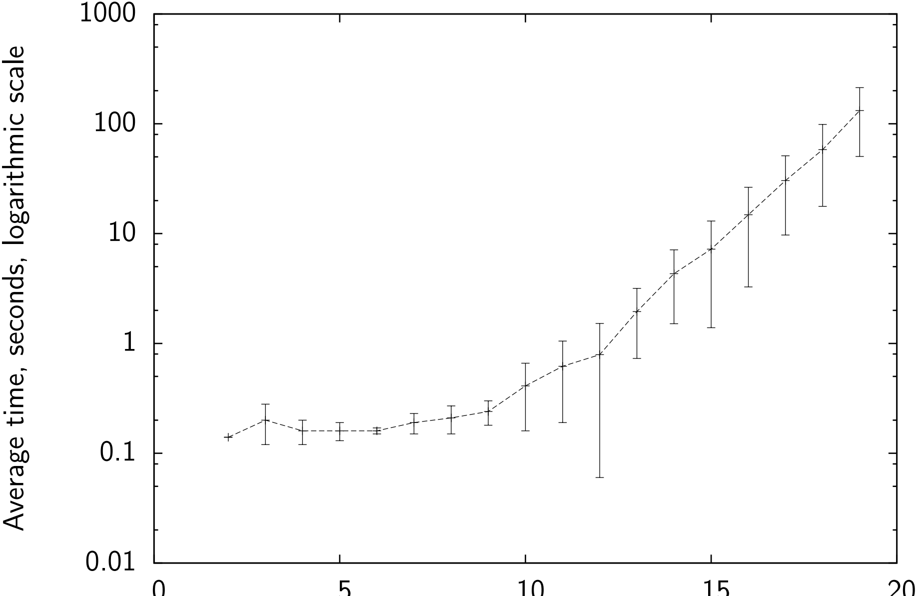
1. Using modified DPLL for finding all such attack suites, which satisfy the attack tree. Basically, finding all SAT solutions for a Boolean formula.
2. Using some optimizations and cutting of hopeless branches.
    1. Theorem: We don't need to consider AND nodes, where some subnode is not satisfied.
3. Saving memory and only storing the most useful attack suite and not all possible combinations.

Realization

1. Using modified DPLL for finding all such attack suites, which satisfy the attack tree. Basically, finding all SAT solutions for a Boolean formula.
2. Using some optimizations and cutting of hopeless branches.
   1. Theorem: We don't need to consider AND nodes, where some subnode is not satisfied.
3. Saving memory and only storing the most useful attack suite and not all possible combinations.
4. Possibly, pre-calculating tables of frequently used values.

Realization

1. Using modified DPLL for finding all such attack suites, which satisfy the attack tree. Basically, finding all SAT solutions for a Boolean formula.
2. Using some optimizations and cutting of hopeless branches.
    1. Theorem: We don't need to consider AND nodes, where some subnode is not satisfied.
3. Saving memory and only storing the most useful attack suite and not all possible combinations.
4. Possibly, pre-calculating tables of frequently used values.
5. Possibly, parallelizing the calculations to multiple processors and multiple machines.


1000
100
10
1
0.1
0.01
Average time, seconds, logarithmic scale
0
5
10
15
20

How good is it?

Consistency with Mauw and Oostdijk

1. Arguing: Our computation model cannot form a distributive attribute domain.
2. At least we tried and it seems quite difficult.

Consistency with Mauw and Oostdijk

1. Arguing: Our computation model cannot form a distributive attribute domain.
2. At least we tried and it seems quite difficult.
3. Even still, our model is consistent with the main idea of Mauw and Oostdijk framework because our trees can be transformed and the tree value remains the same.

Consistency with Mauw and Oostdijk

1. Arguing: Our computation model cannot form a distributive attribute domain.
2. At least we tried and it seems quite difficult.
3. Even still, our model is consistent with the main idea of Mauw and Oostdijk framework because our trees can be transformed and the tree value remains the same.
4. This raises the question about differences of propagating and non-propagating computations. Perhaps the Mauw and Oostdjik framework could be extended to non-propagating computations as well.

Conclusions

1. We presented new attack tree computation rules, which model attacker choices more precisely and provides bigger outcomes than the old model.
2. It is very difficult to calculate outcome of bigger trees. In some sense, this is the perfect solution for attack tree outcome calculation and we need to search for practical approximations now.
3. There are interesting questions about consistency with Mauw and Oostdijk framework model.