Evaluating Security Against Rational Attackers

Jan Willemson

Cybernetica and Tartu University, Estonia

Final Workshop of CDC 2002–2007

January 21-22, 2008

Papers covered

- Ahto Buldas, Peeter Laud, Jaan Priisalu, Märt Saarepera, Jan Willemson, *Rational Choice of Security Measures via Multi-Parameter Attack Trees*, in CRITIS 2006, LNCS 4347, pp. 235–248
- Aivo Jürgenson, Jan Willemson, *Processing Multi-parameter Attacktrees with Estimated Parameter Values*, in IWSEC 2007, LNCS 4752, pp. 308–319

Motivation

- It is a complicated task to evaluate whether (IT-)infrastructure of a company is protected
  - sufficiently (i.e. achieving a satisfactory level), and
  - reasonably (i.e. not spending too much)

Motivation

- It is a complicated task to evaluate whether (IT-)infrastructure of a company is protected
  - sufficiently (i.e. achieving a satisfactory level), and
  - reasonably (i.e. not spending too much)
- Even if the losses associated with vulnerability exploits can be estimated, the corresponding probabilities are very difficult to evaluate

Motivation

- It is a complicated task to evaluate whether (IT-)infrastructure of a company is protected
  - sufficiently (i.e. achieving a satisfactory level), and
  - reasonably (i.e. not spending too much)
- Even if the losses associated with vulnerability exploits can be estimated, the corresponding probabilities are very difficult to evaluate
- This is especially true for targeted, company-specific attacks, since the required statistics does not exist or is difficult to get

Rational Attackers and Attack Trees

- Luckily, targeted attacks are often *rational*, i.e. the attackers
  - attack only if the attack is profitable, and
  - choose the attack with the highest outcome

Rational Attackers and Attack Trees

- Luckily, targeted attacks are often *rational*, i.e. the attackers
  - attack only if the attack is profitable, and
  - choose the attack with the highest outcome
- From now on, we will assume rational attacks

Rational Attackers and Attack Trees

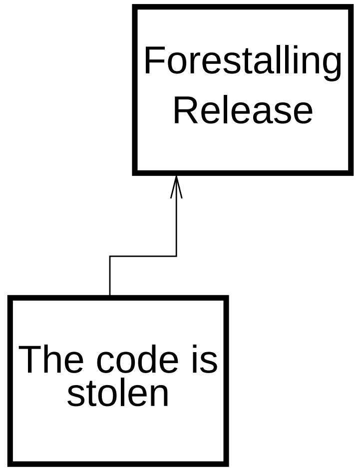
- Luckily, targeted attacks are often *rational*, i.e. the attackers
  - attack only if the attack is profitable, and
  - choose the attack with the highest outcome
- From now on, we will assume rational attacks
- Such attacks can be modeled using gradual refinement starting from primary threats and breaking them down to elementary attacks

Rational Attackers and Attack Trees

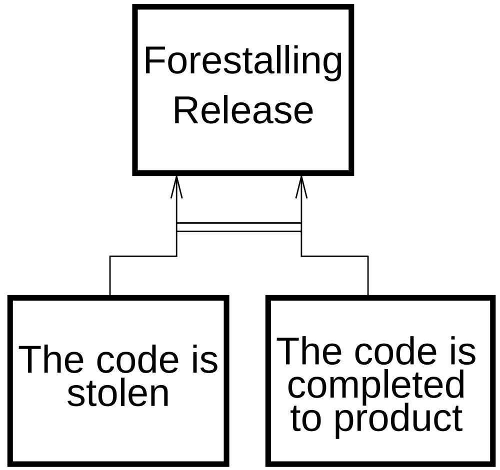
- Luckily, targeted attacks are often *rational*, i.e. the attackers
  - attack only if the attack is profitable, and
  - choose the attack with the highest outcome
- From now on, we will assume rational attacks
- Such attacks can be modeled using gradual refinement starting from primary threats and breaking them down to elementary attacks
- As a result, we obtain an *attack tree*

An Attack Tree

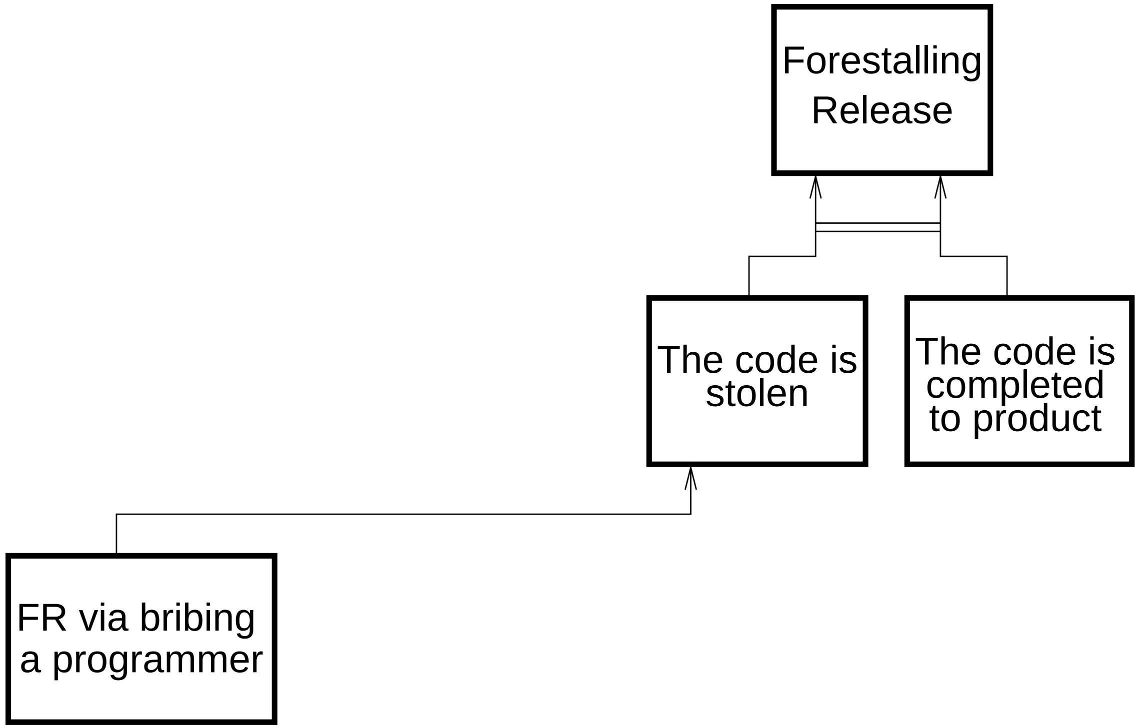
An Attack Tree



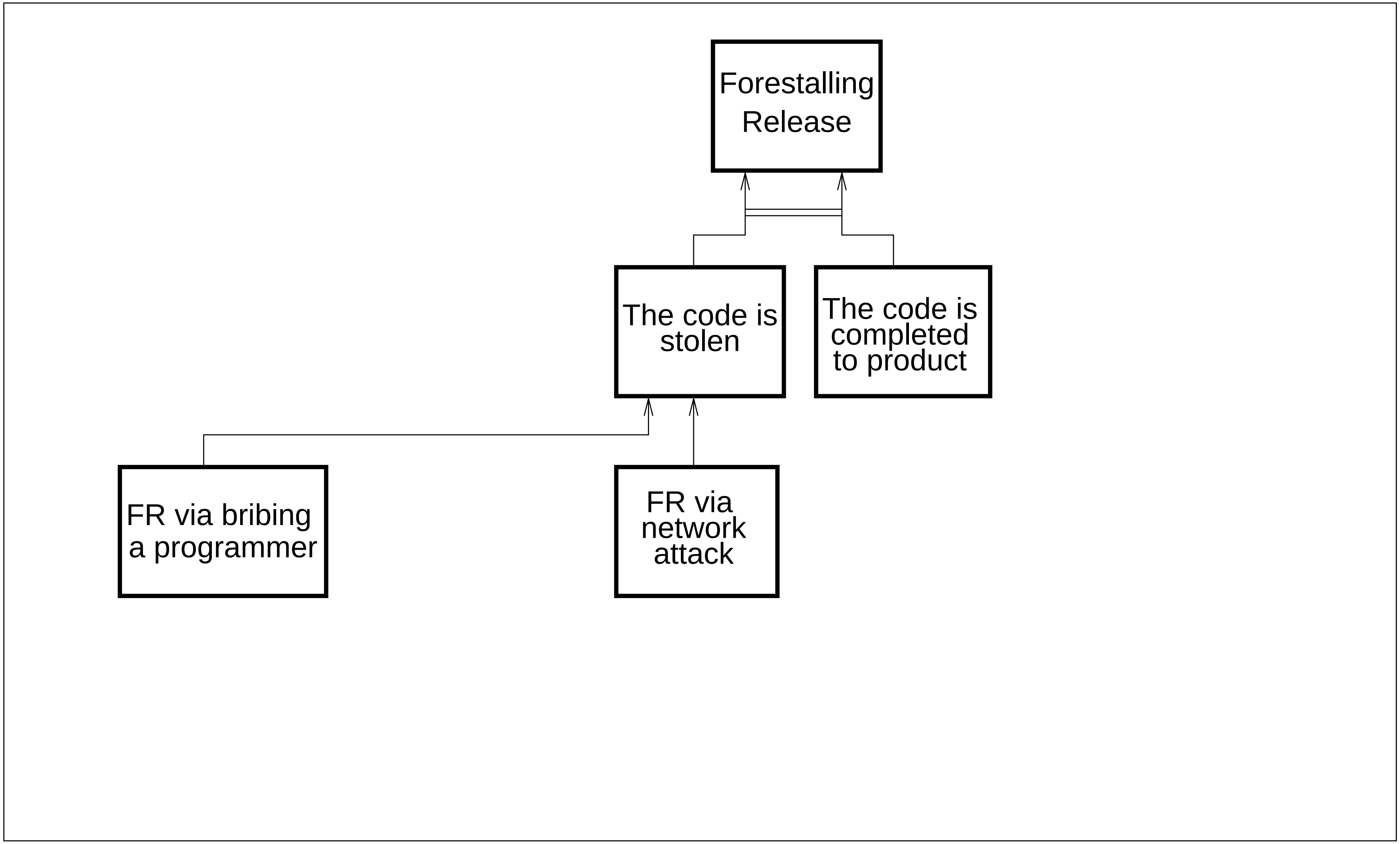
An Attack Tree



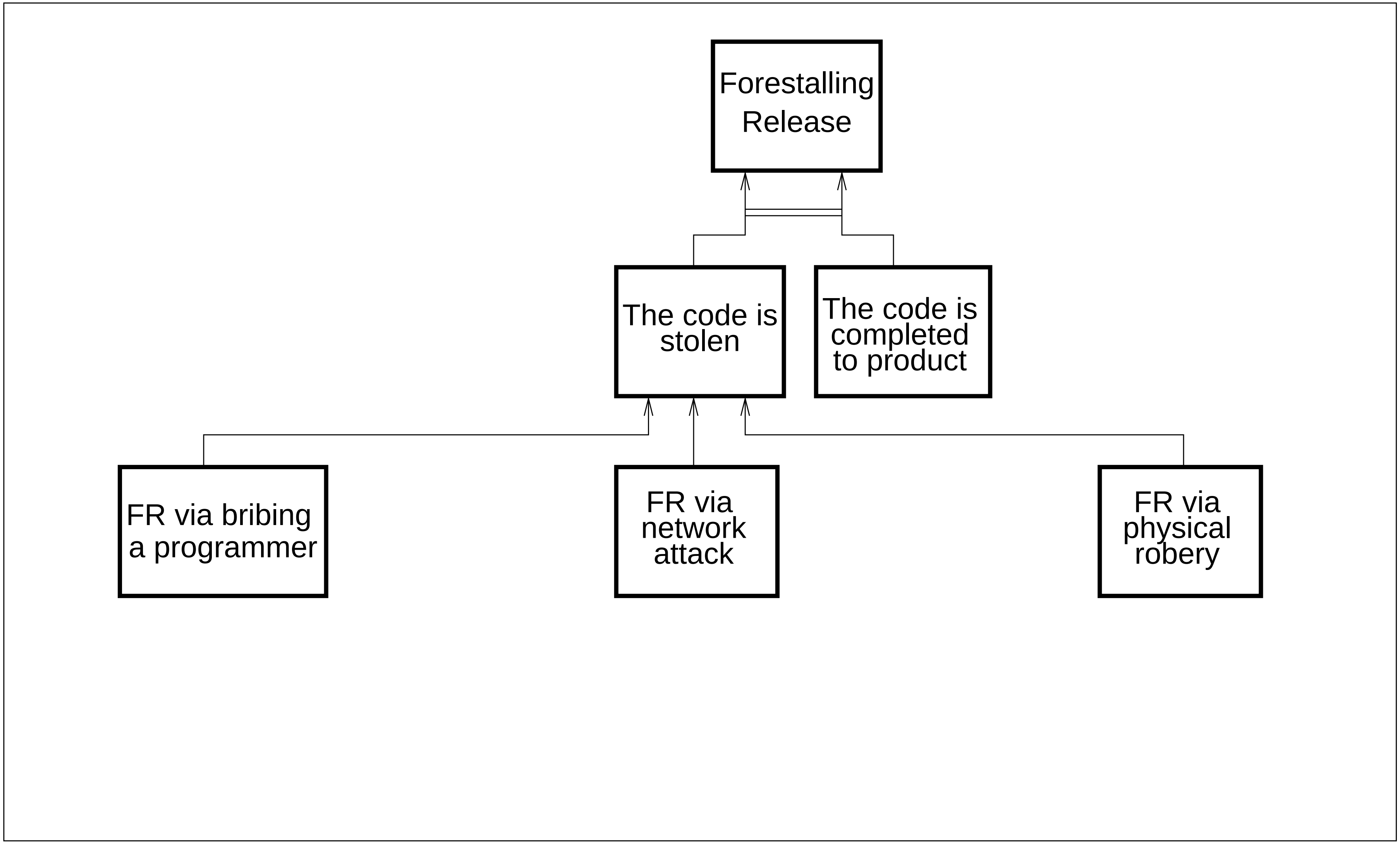
An Attack Tree



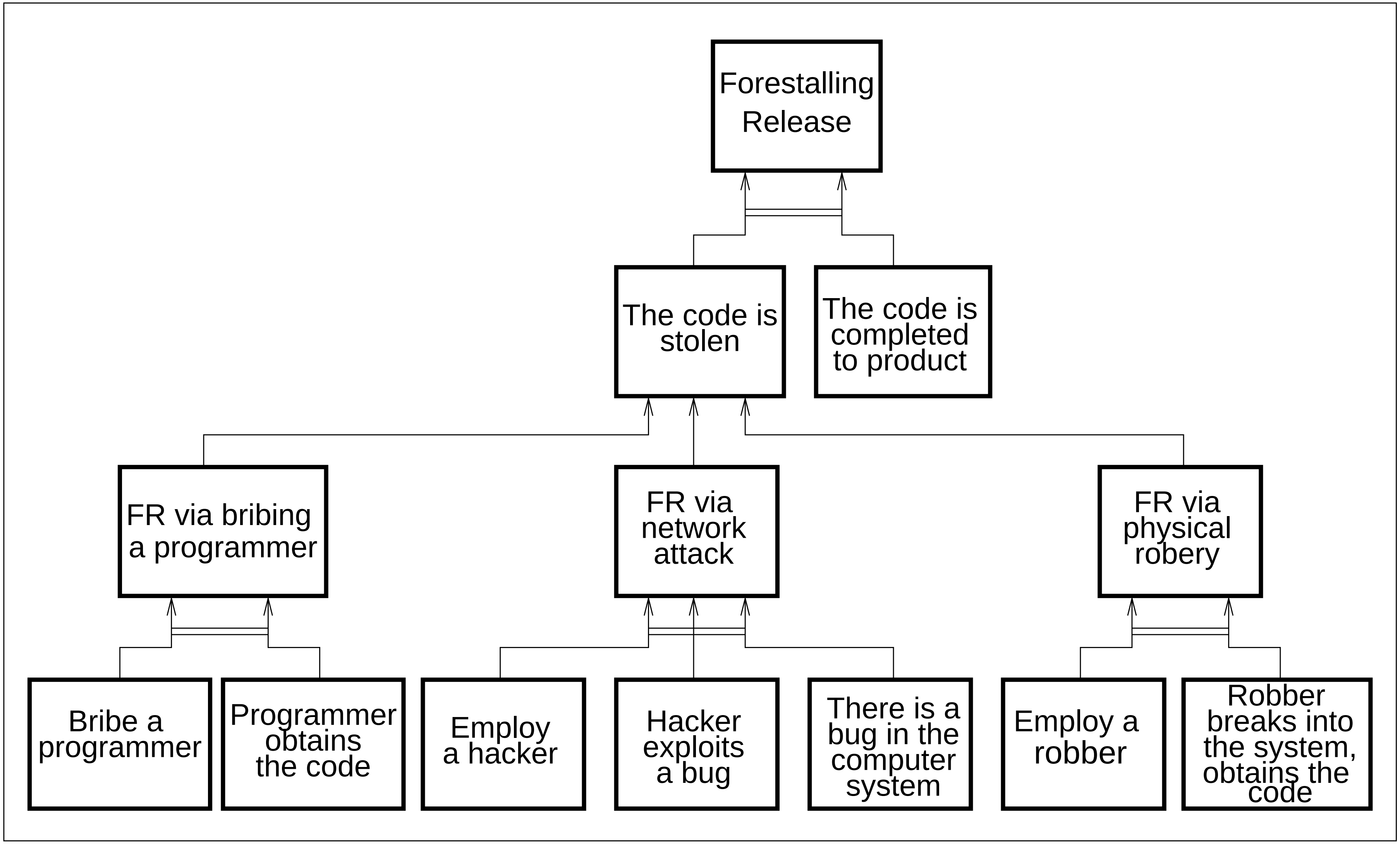
An Attack Tree



An Attack Tree



An Attack Tree



Parametrizing the Attack Tree

- When modelling the attack, we need to
  - set some parameter values to the leaves of the tree
  - define computation rules for parameter propagation

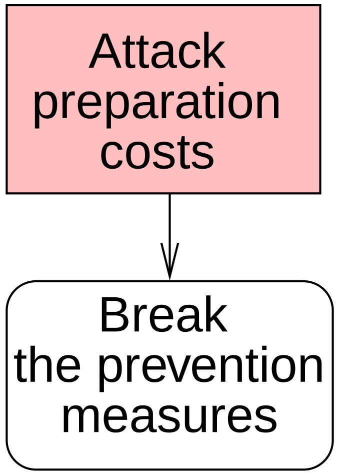
Parametrizing the Attack Tree

- When modelling the attack, we need to
  - set some parameter values to the leaves of the tree
  - define computation rules for parameter propagation
- In this framework we will consider the following parameters:
  - Gains – the gains of the attacker if attack succeeds
  - Costs – the cost of the attack
  - p – the success probability of the attack
  - q, $\mathsf{Penalties}$ – the probability of getting caught and penalties (if the attack was successful)
  - q_-, $\mathsf{Penalties}_-$ – the probability of getting caught and penalties (if the attack was unsuccessful)

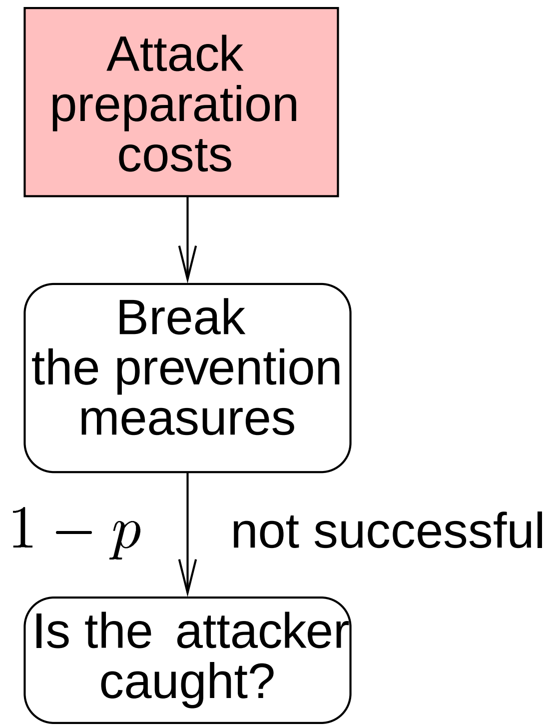
The Attack Game

Attack preparation costs

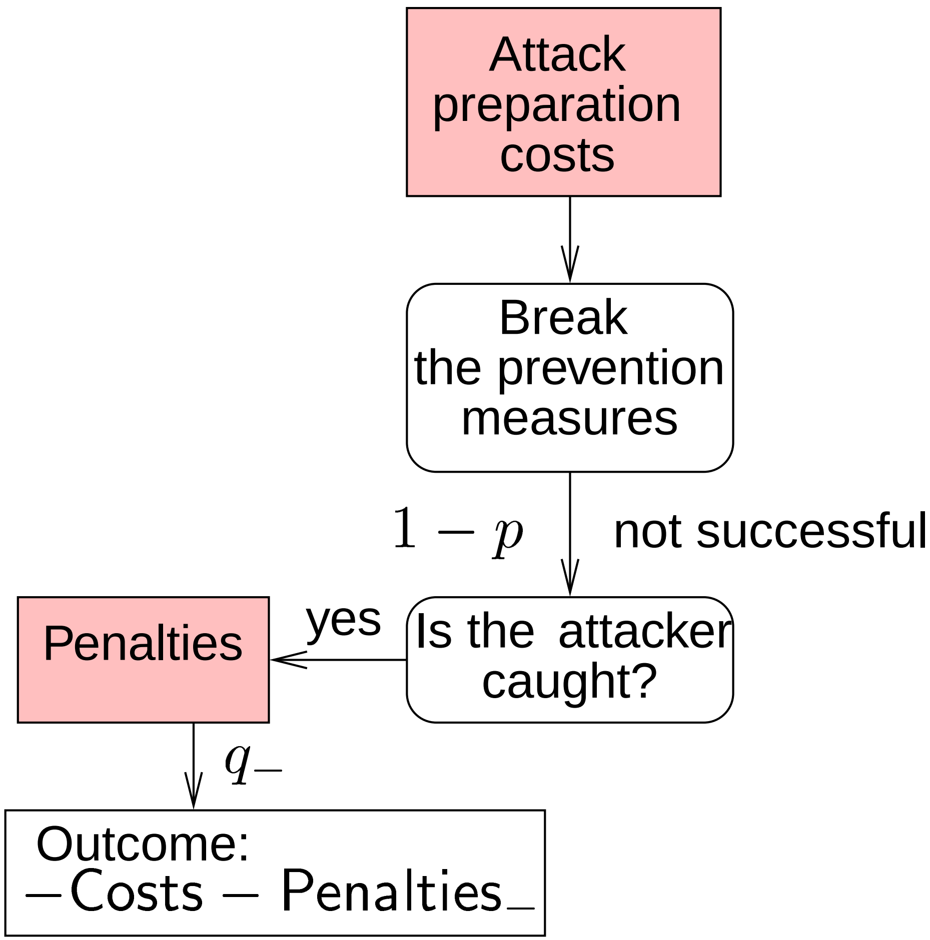
The Attack Game



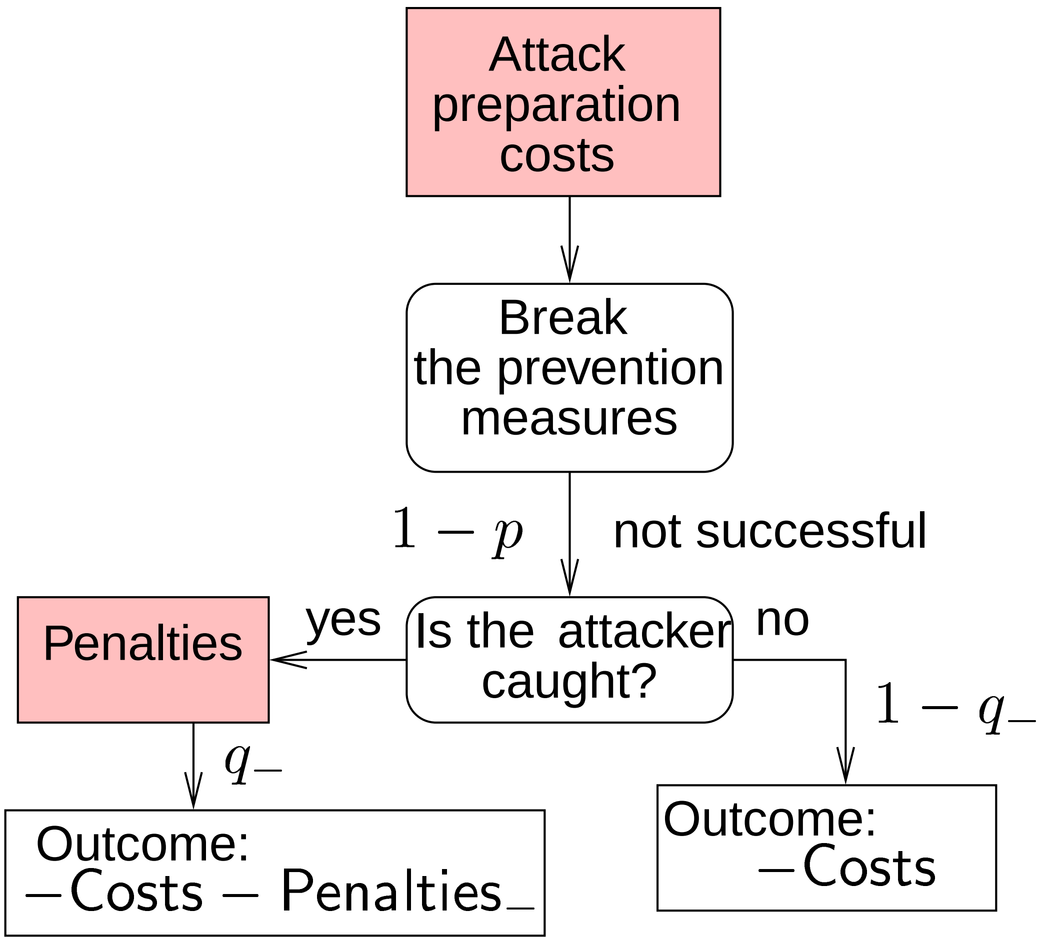
The Attack Game



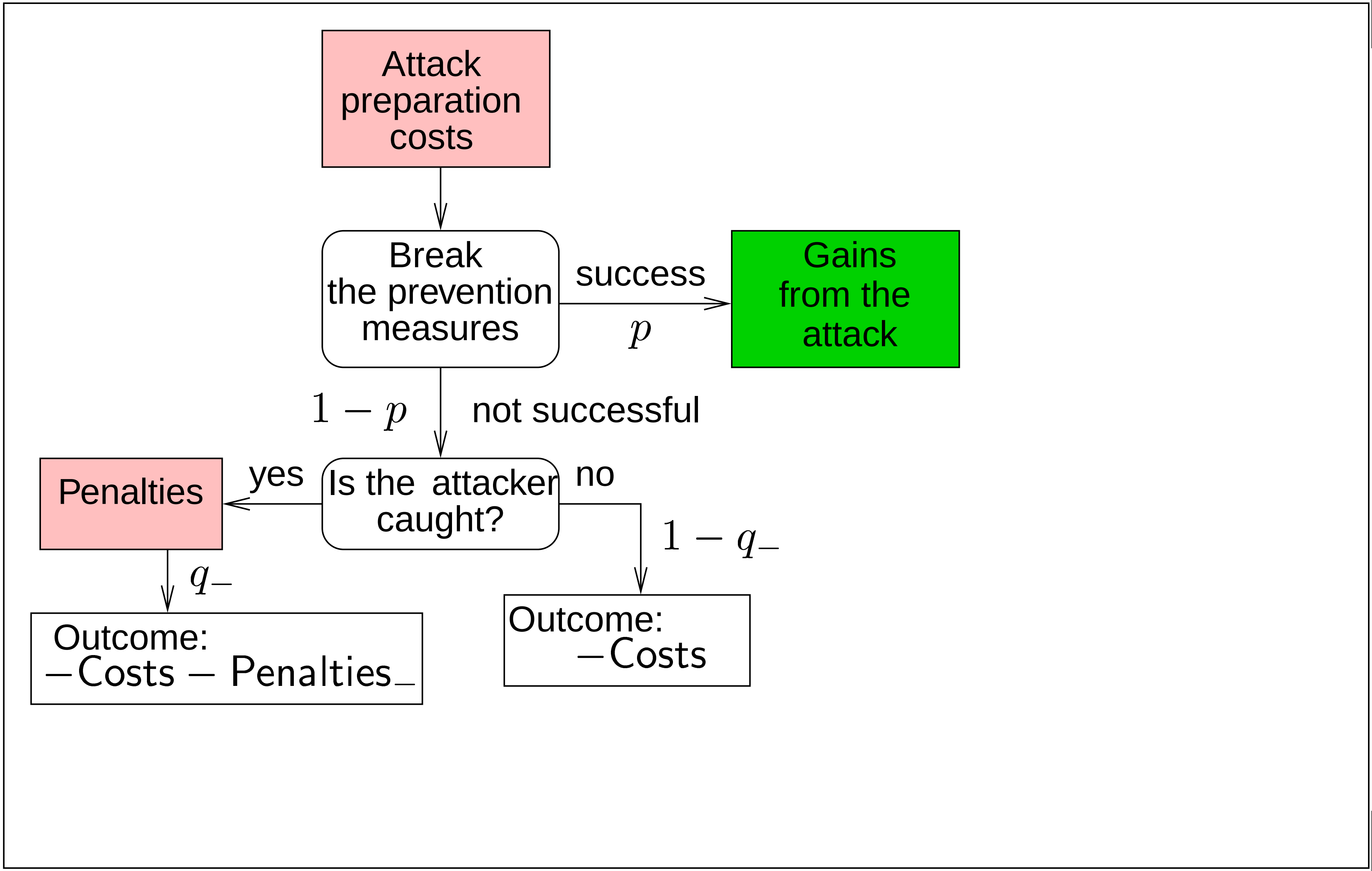
The Attack Game



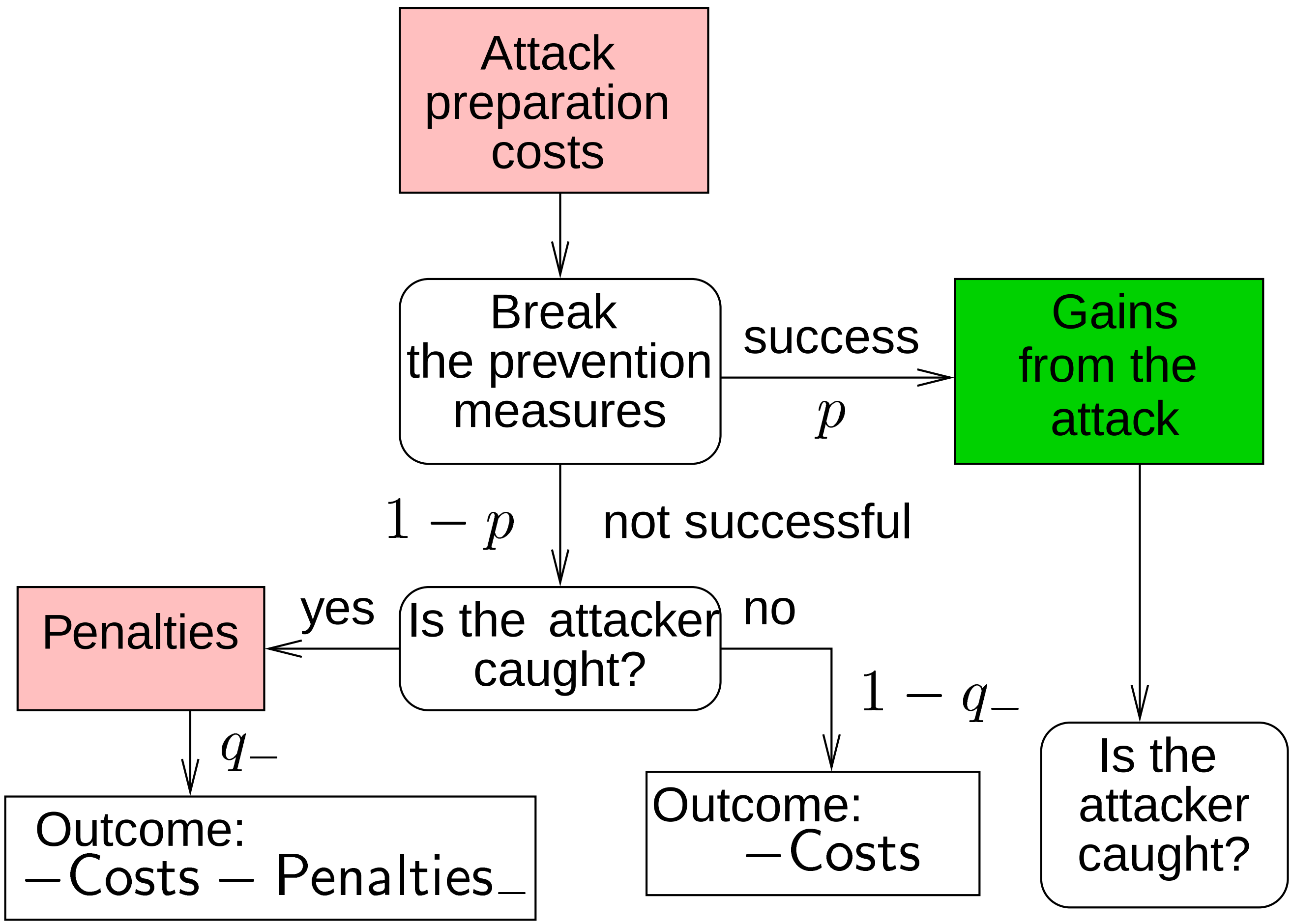
The Attack Game



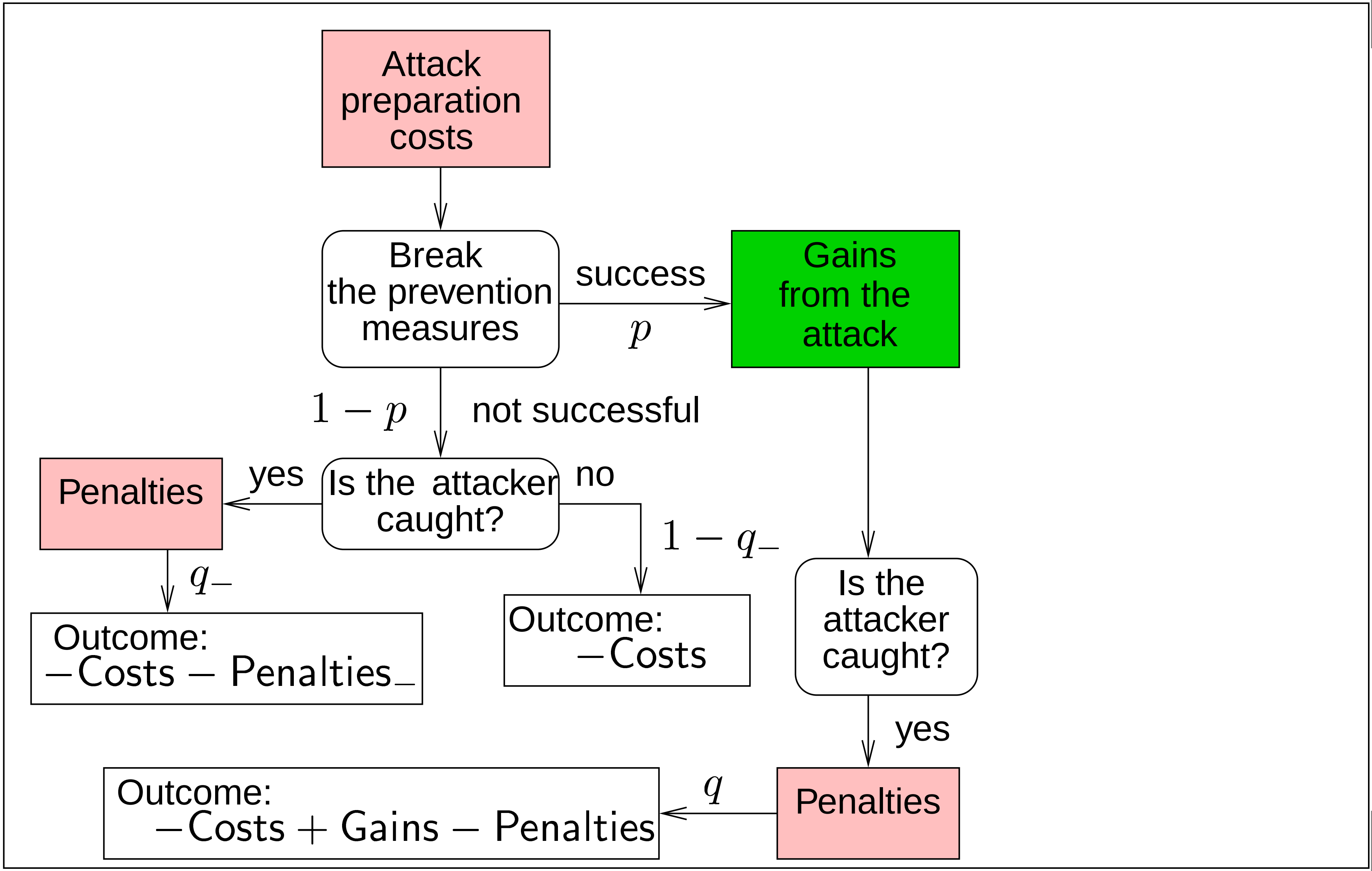
The Attack Game



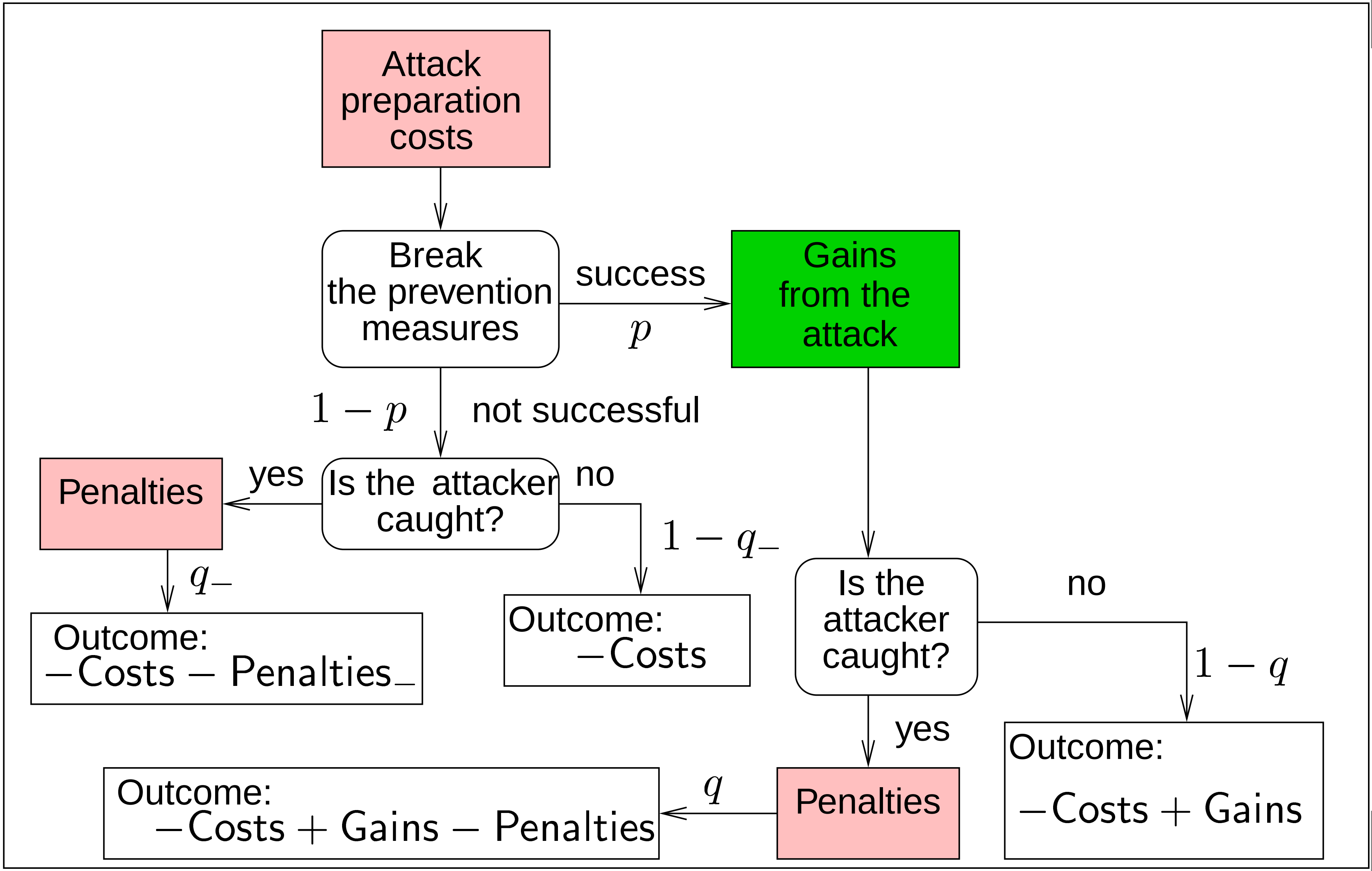
The Attack Game



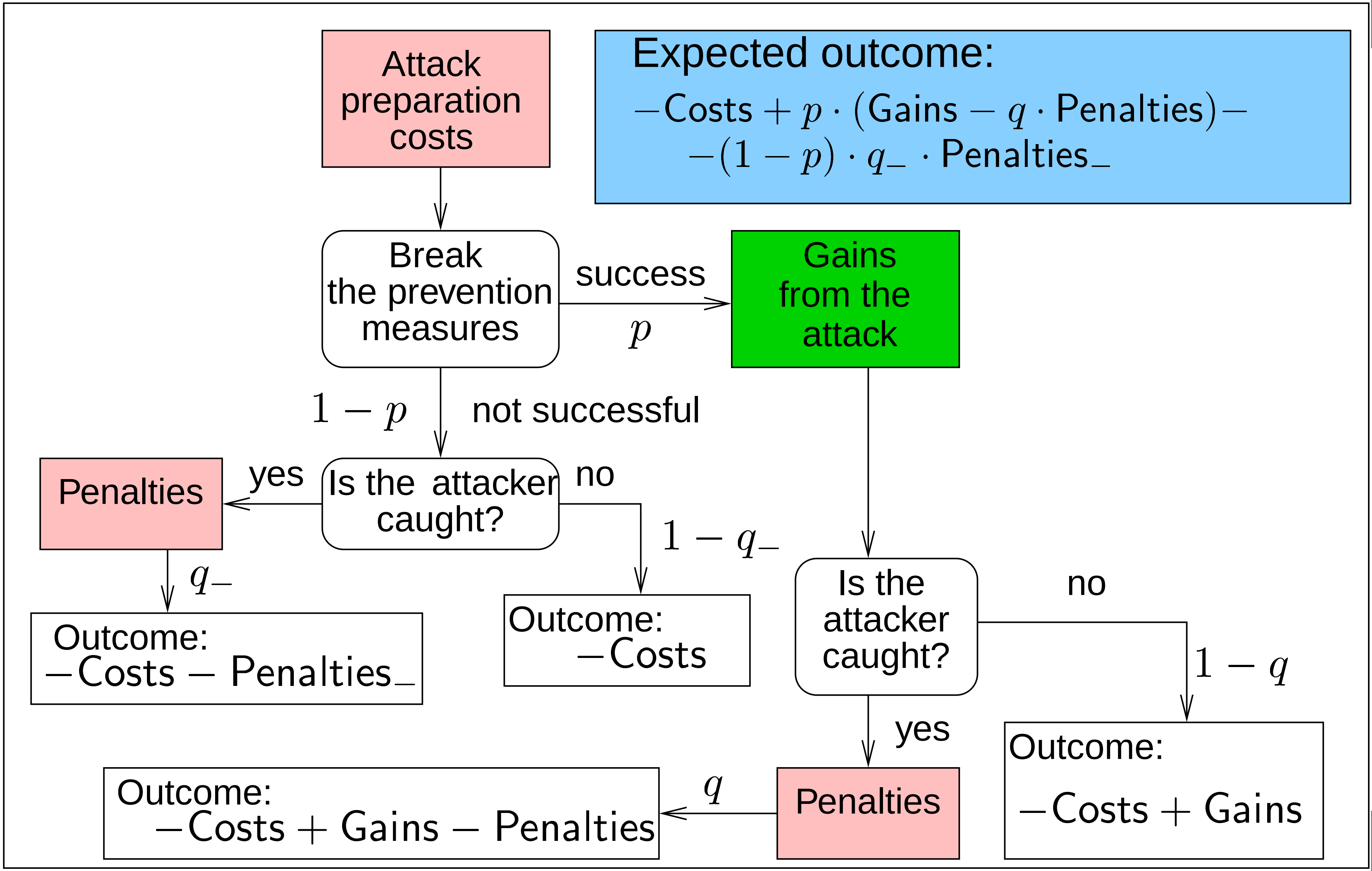
The Attack Game



The Attack Game



The Attack Game



Tree Computations (I)

- Denoting $\pi = q \cdot \mathsf{Penalties}$ and $\pi_- = q_- \cdot \mathsf{Penalties}_-$, we set the parameters $(\mathsf{Costs}, p, \pi, \pi_-)$ for every leaf node. Then we have

$$\mathsf{Outcome} = -\mathsf{Costs} + p \cdot \mathsf{Gains} - p \cdot \pi - (1-p) \cdot \pi_-$$

- For an OR-node with child nodes with parameters $(\mathsf{Costs}_1, p_1, \pi_1, \pi_{1-})$ and $(\mathsf{Costs}_2, p_2, \pi_2, \pi_{2-})$ the parameters $(\mathsf{Costs}, p, \pi, \pi_-)$ are computed as:

$$(\mathsf{Costs}, p, \pi, \pi_-) =$$

$$\begin{cases} (\mathsf{Costs}_1, p_1, \pi_1, \pi_{1-}), & \text{if } \mathsf{Outcome}_1 > \mathsf{Outcome}_2 \\ (\mathsf{Costs}_2, p_2, \pi_2, \pi_{2-}), & \text{if } \mathsf{Outcome}_1 \leq \mathsf{Outcome}_2 \end{cases}$$

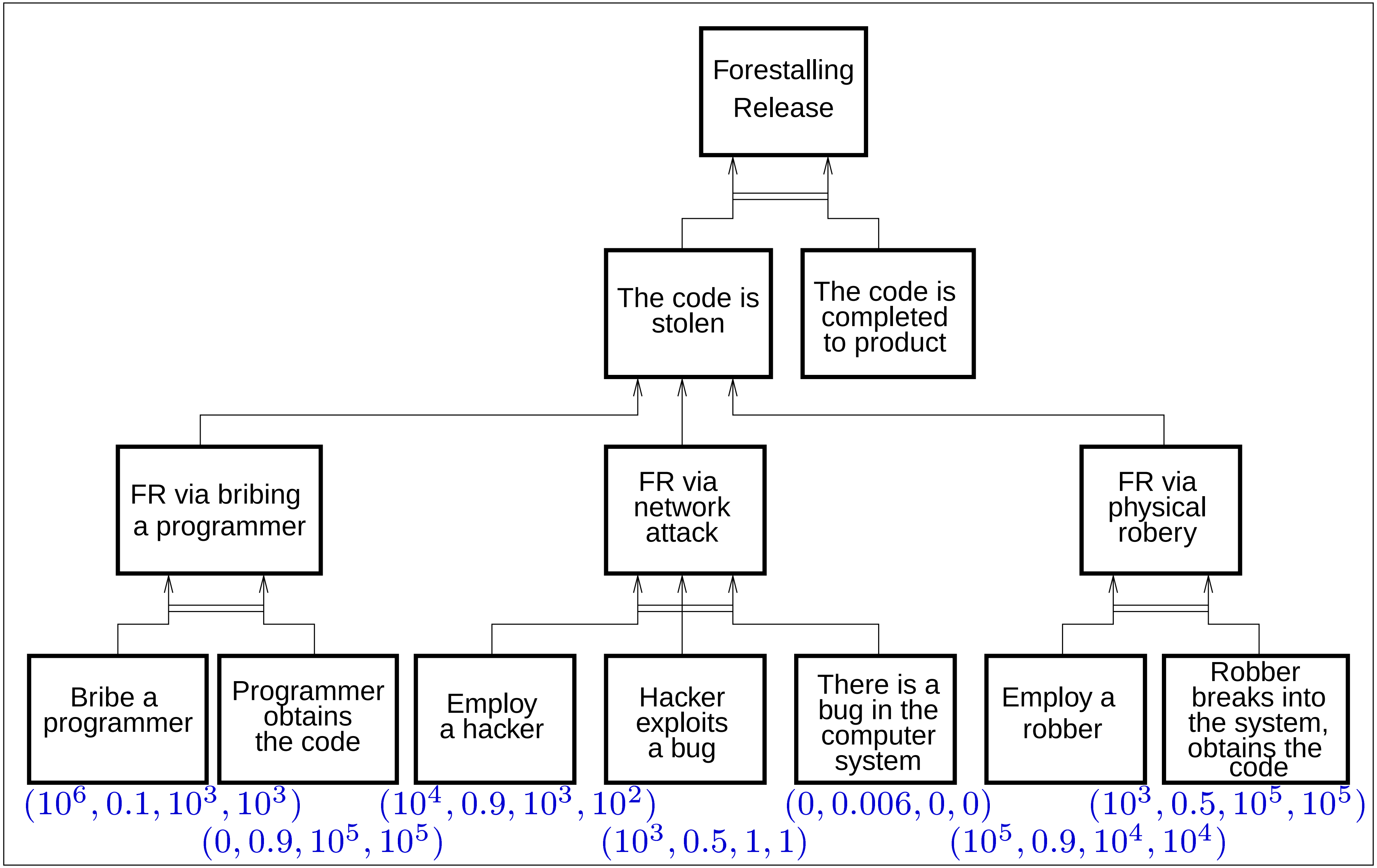
Tree Computations (II)

- For a AND-node with child nodes with parameters $(\mathsf{Costs}_1, p_1, \pi_1, \pi_{1-})$ and $(\mathsf{Costs}_2, p_2, \pi_2, \pi_{2-})$ the parameters $(\mathsf{Costs}, p, \pi, \pi_-)$ are computed as follows:

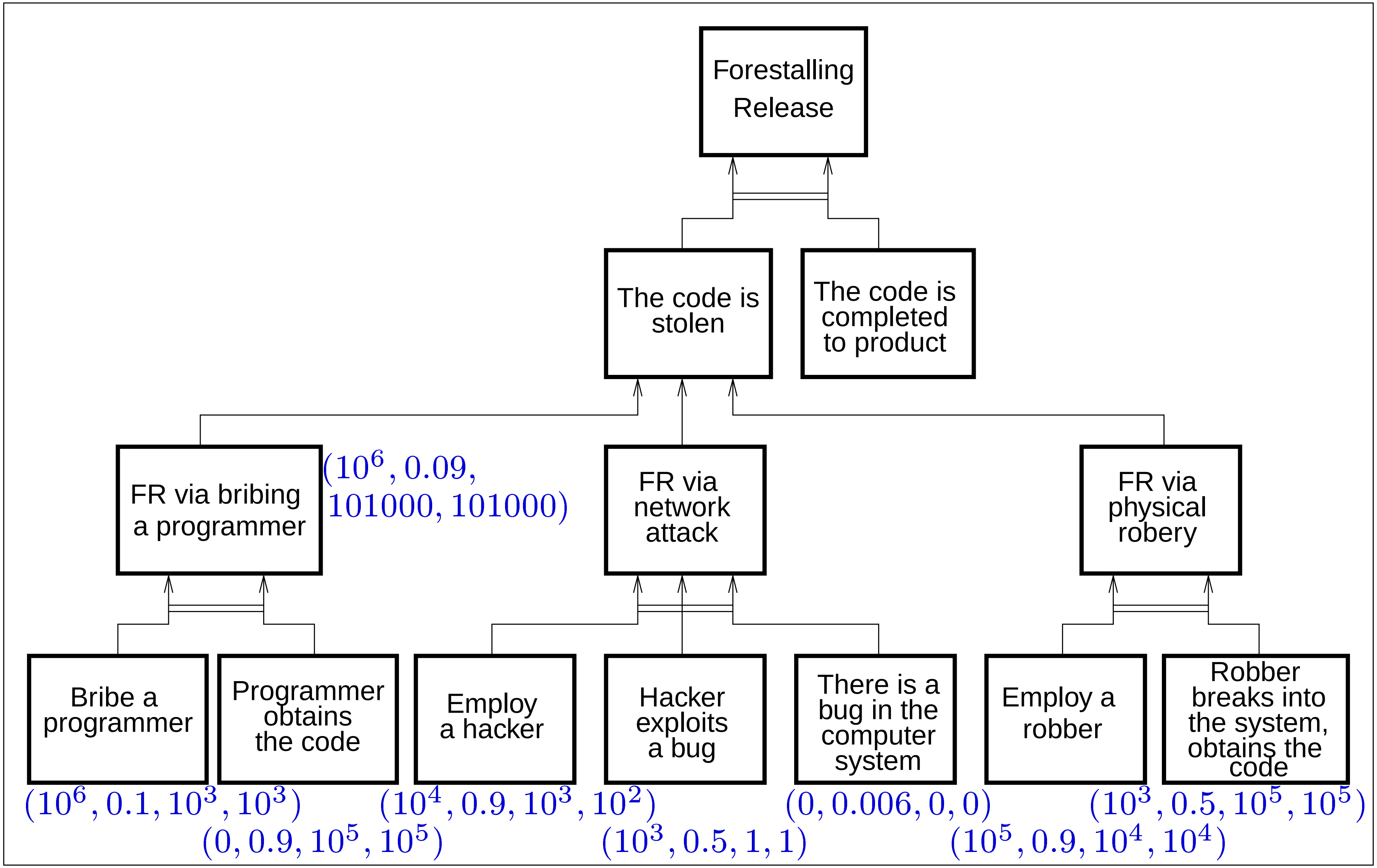
$$\begin{aligned}
\mathsf{Costs} &= \mathsf{Costs}_1 + \mathsf{Costs}_2 \\
p &= p_1 \cdot p_2 \\
\pi &= \pi_1 + \pi_2 \\
\pi_- &= \frac{p_1(1-p_2)(\pi_1 + \pi_{2-}) + (1-p_1)p_2(\pi_{1-} + \pi_2)}{1 - p_1 p_2} + \\
&\quad \frac{(1-p_1)(1-p_2)(\pi_{1-} + \pi_{2-})}{1 - p_1 p_2}
\end{aligned}$$

- The last formula represents the average penalty of an attacker, assuming that at least one of the two child-attacks was not successful

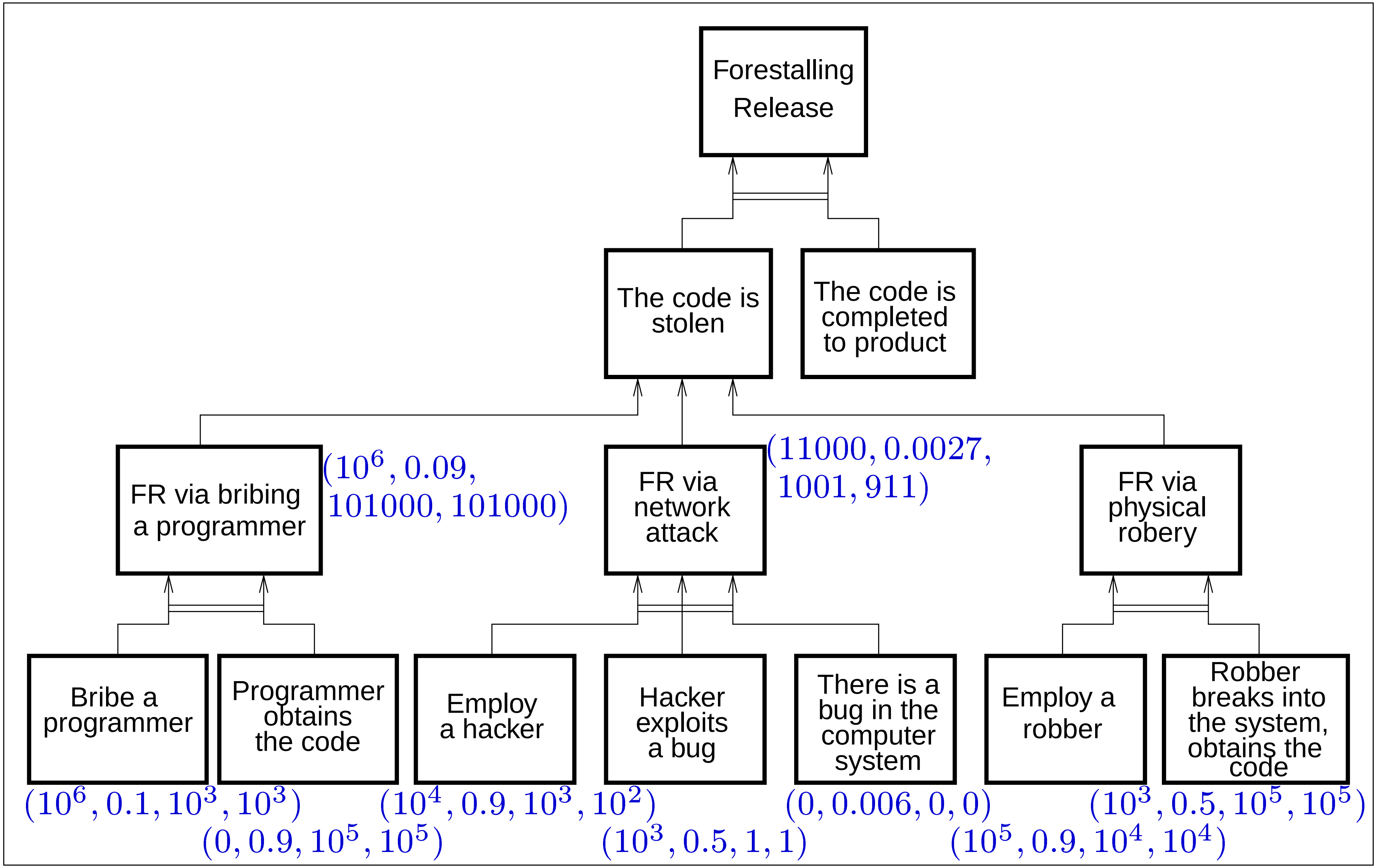
Tree Computations: Example



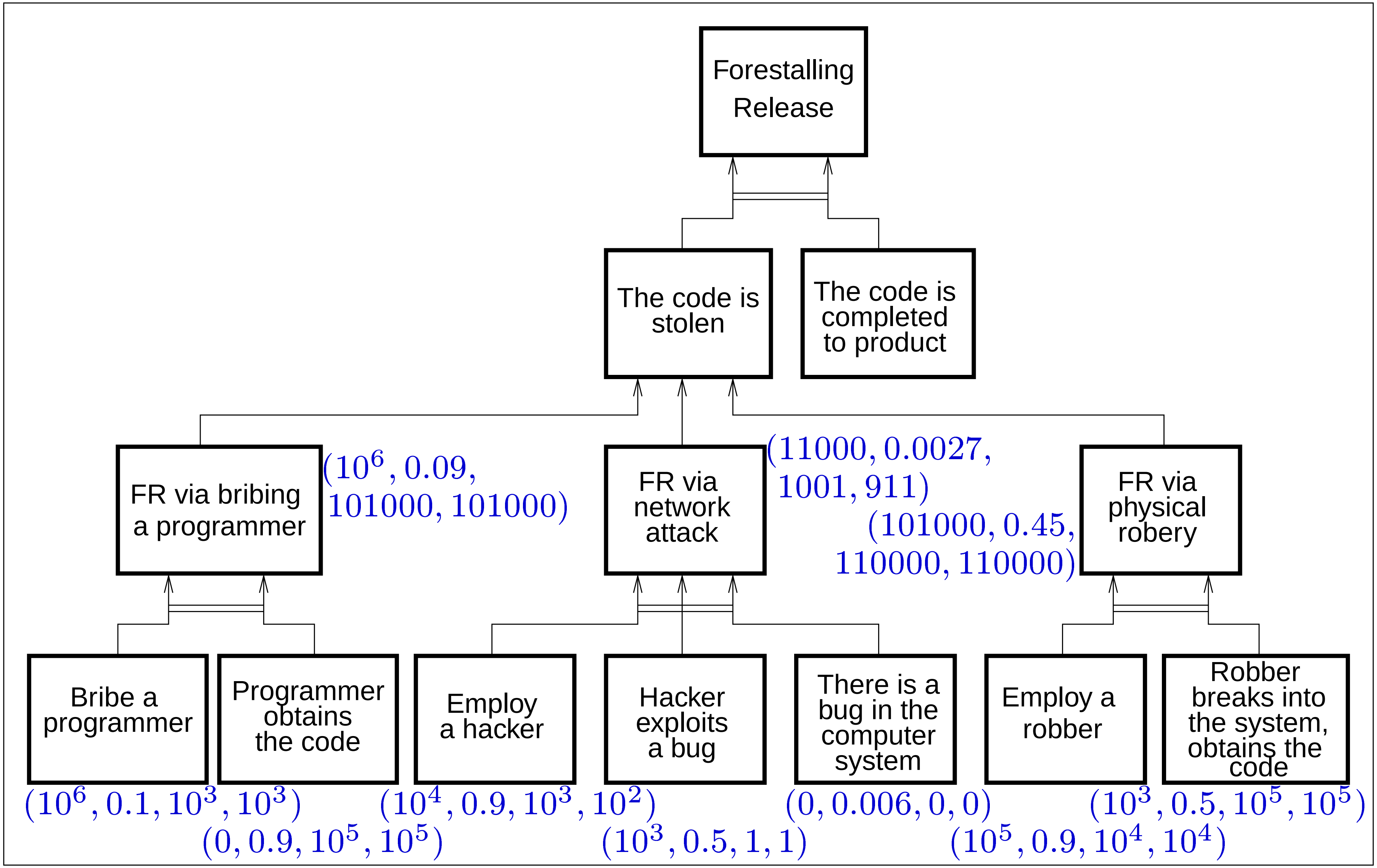
Tree Computations: Example



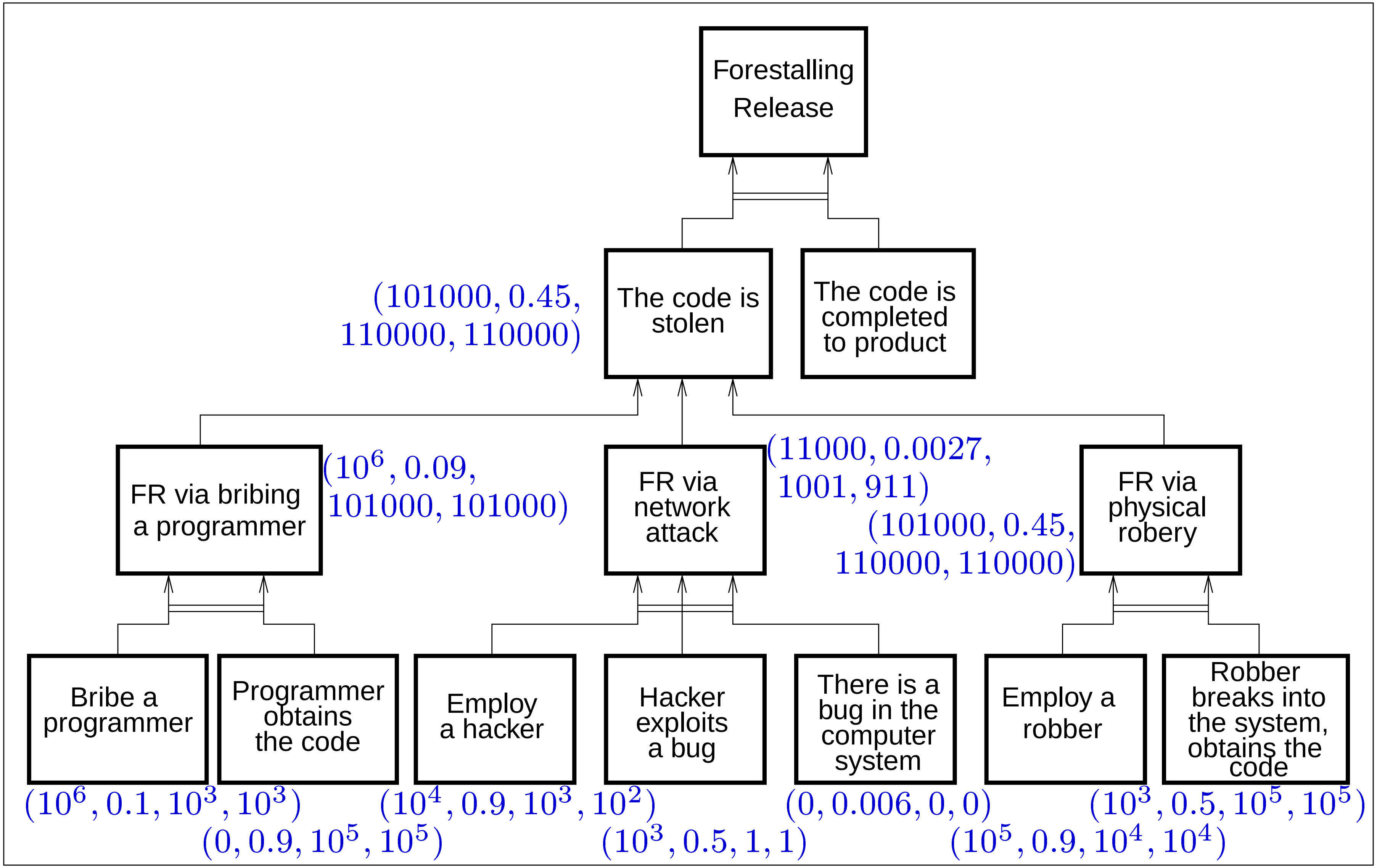
Tree Computations: Example



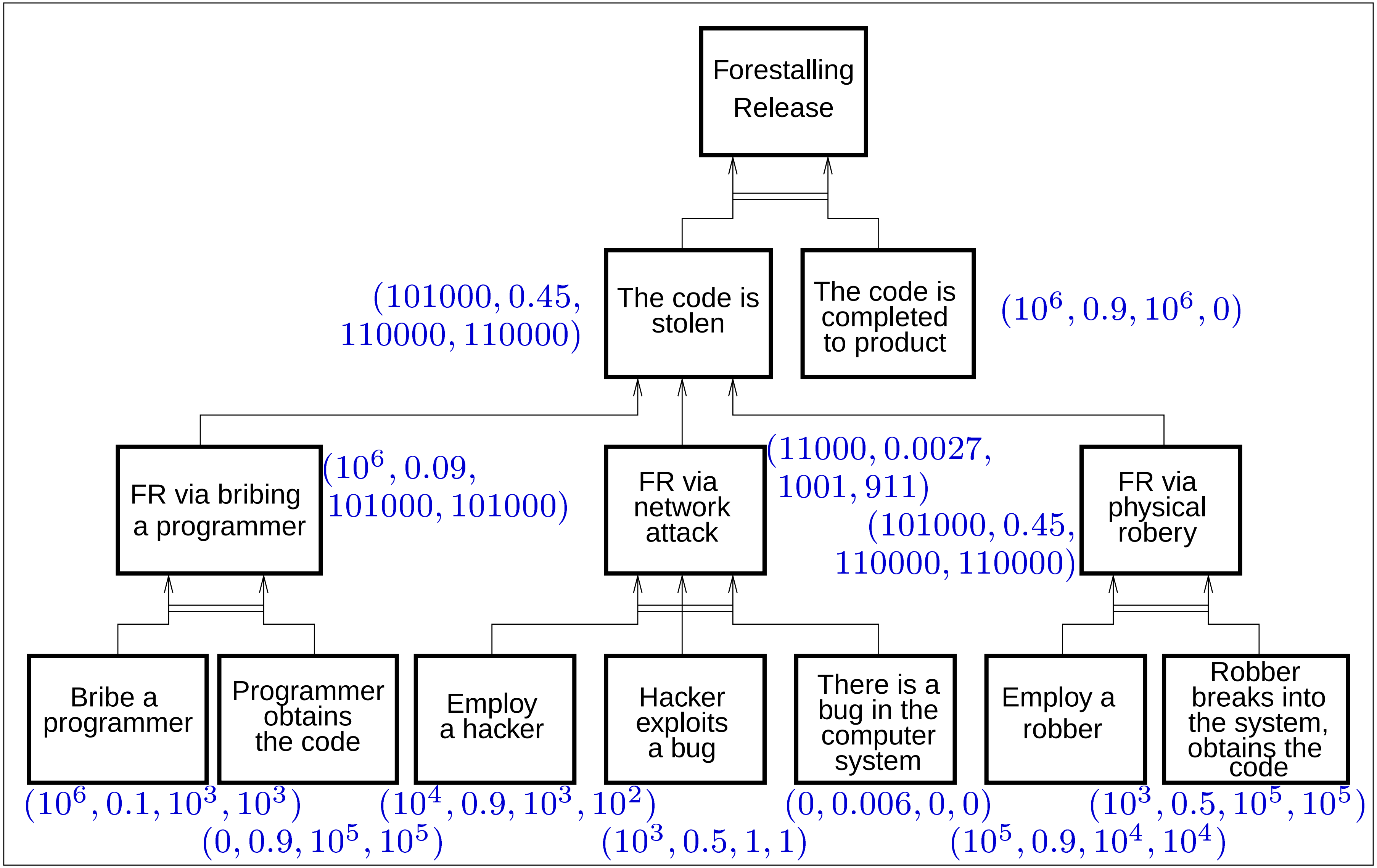
Tree Computations: Example



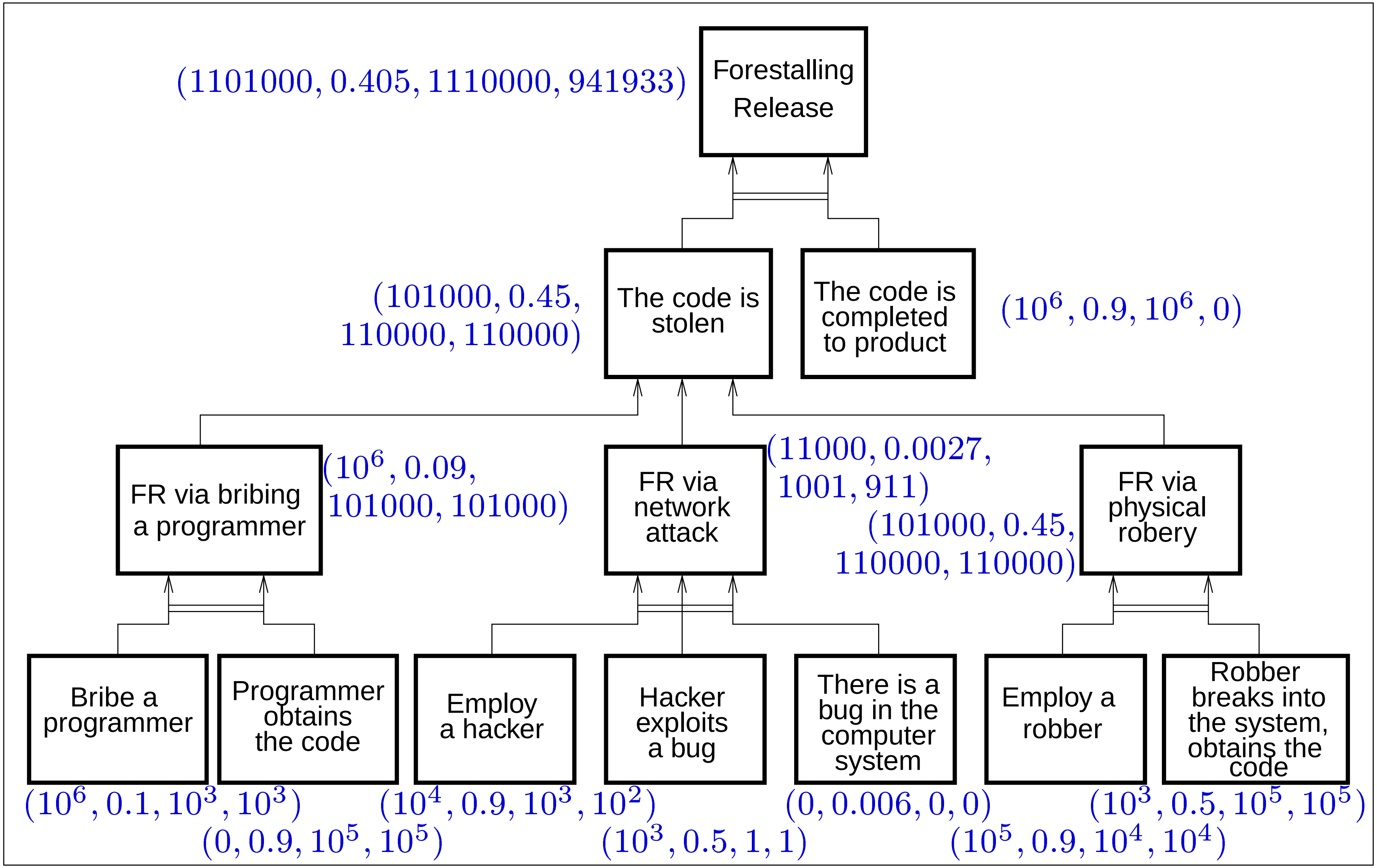
Tree Computations: Example



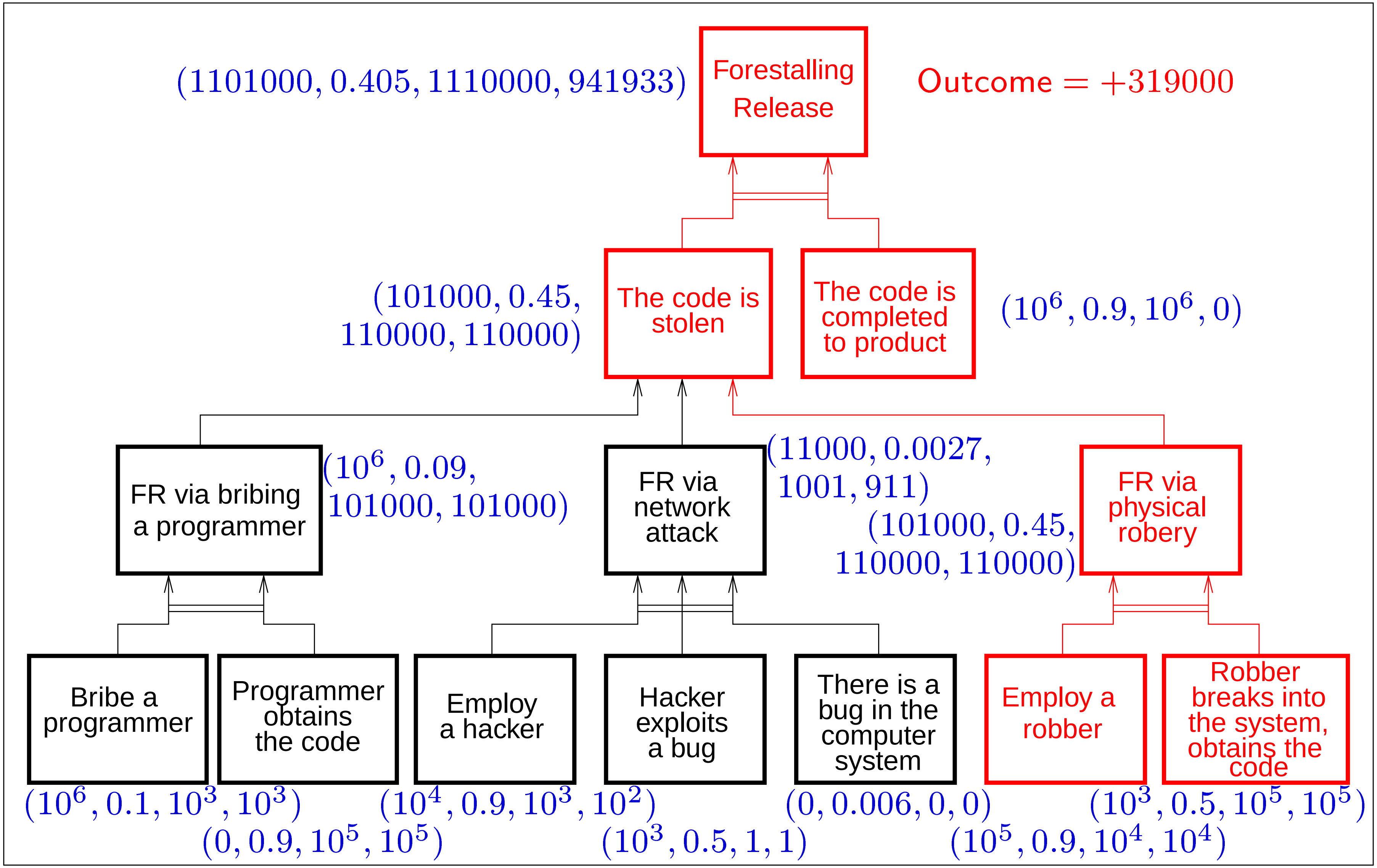
Tree Computations: Example



Tree Computations: Example



Tree Computations: Example



Evaluating Security Measures

- After building the attack tree, we can evaluate and compare potential security measures
- Security measures may
  - increase detection probability, hence increasing π and/or π_-,
  - increase Costs of the attack,
  - reduce the probability p of the attack success.
  - etc
- We can then change the respective parameters, make the tree computations again and see, whether the $\mathsf{Outcome}$ has become negative

Modeling Parameter Estimations

- Usually, when an expert evaluates some parameter, his estimation is not absolute, but holds with some confidence
- Thus, we can consider *estimated values* of the form

$$p_X = \Pr[k_1 \leq X \leq k_2] \ ,$$

where p_X is the *probability* of the unknown value of the parameter X being within the interval of $[k_1, k_2]$

Modeling Parameter Estimations

- Usually, when an expert evaluates some parameter, his estimation is not absolute, but holds with some confidence
- Thus, we can consider *estimated values* of the form

$$p_X = \Pr[k_1 \leq X \leq k_2] \ ,$$

  where p_X is the *probability* of the unknown value of the parameter X being within the interval of $[k_1, k_2]$
- We will later refer to p_X as *confidence* or *confidence level* and $\mathcal{X} = (p_X, k_1, k_2)$ as *estimation*

Modeling Parameter Estimations

- Usually, when an expert evaluates some parameter, his estimation is not absolute, but holds with some confidence
- Thus, we can consider *estimated values* of the form

  $$p_X = \Pr[k_1 \leq X \leq k_2] \ ,$$

  where p_X is the *probability* of the unknown value of the parameter X being within the interval of $[k_1, k_2]$
- We will later refer to p_X as *confidence* or *confidence level* and $\mathcal{X} = (p_X, k_1, k_2)$ as *estimation*
- *How does one compute with estimated values?*

Estimation Arithmetic – Operations

Our tree computations need the following arithmetic primitives:

- Adding a fixed real number
- Multiplying by a fixed real number
- Adding/subtracting two estimated values
- Multiplying two estimated values
- Dividing two estimated values
- Comparing two estimated values

Estimation Arithmetic – Basic Pattern

The basic pattern of all the estimation arithmetic operations is the following:

- Convert the estimations to normally distributed random variables
  - If needed, centralize them to mean value 0
- Compute with the random variables
- Convert the resulting random variable back to an estimation
  - If needed, de-centralize

Estimation → Random Variable

- To convert the estimation $\mathcal{X}$ to a random variable X, we have to find out the mean a_X and standard deviation σ_X:

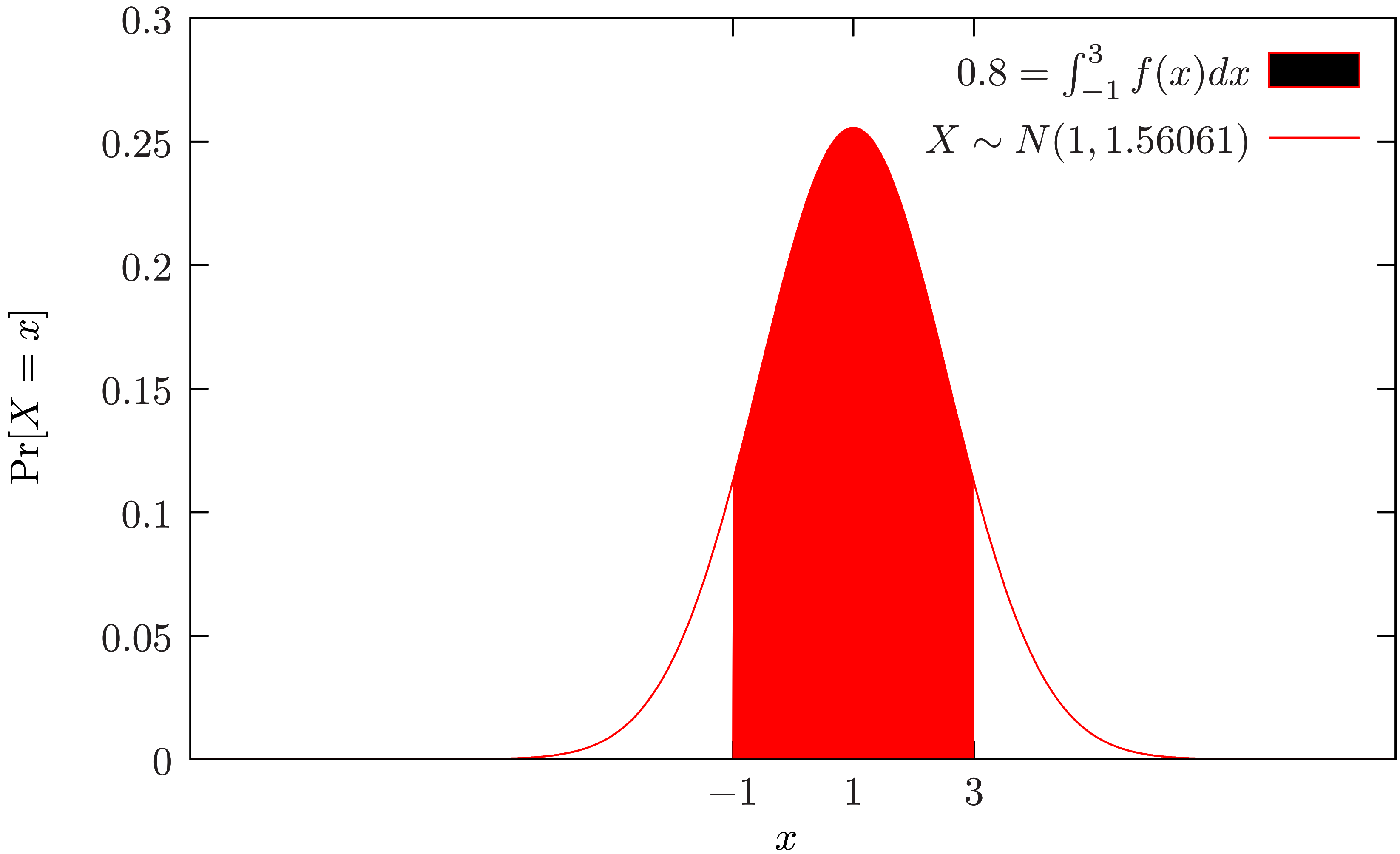
$$a_X = \mathbf{E}X = \frac{k_1 + k_2}{2} \quad ,$$

$$p_X = \Pr(k_1 \leq X \leq k_2) = \Phi\left(\frac{k_2 - a_X}{\sigma_X}\right) - \Phi\left(\frac{k_1 - a_X}{\sigma_X}\right) \quad ,$$

  where the $\Phi(x)$ is the Laplace's function

- We denote conversion of estimation $\mathcal{X}$ to normally distributed random variable X as $\mathcal{X} = (p_X, k_1, k_2) \rightarrow X \sim N(a_X, \sigma_X)$

$$\mathcal{X} = (p_X, k_1, k_2) \rightarrow X \sim N(a_X, \sigma_X)$$



Random Variable $\rightarrow$ Estimation

- To convert the probabilistic variable X back to an estimation $\mathcal{X}$, we need to specify the confidence p'_X
- To simplify the operations with our estimations of the attack-tree node parameters, we will convert all estimations to the same global confidence level p_T
- In effect, p_T defines the confidence level or the margin of error at which we would like to have the answer of our attack-tree analysis given
- If the original estimation $\mathcal{X}$ of an expert is given using some other confidence level p_X, we first convert $\mathcal{X} = (p_X, k_1, k_2) \rightarrow X \sim N(a_X, \sigma_X)$ and then find the new interval $[k'_1, k'_2]$ by $X \sim N(a_X, \sigma_X) \rightarrow \mathcal{X} = (p_T, k'_1, k'_2)$

Soundness of Computations

- Most parameters of the nodes have a limited value domain, e.g. $\mathsf{Cost} \geq 0$ and $p \in [0, 1]$
- However, as a result of conversions and tree computations, some values may drop out of this domain
- Generally, such a situation indicates that no sound conclusions can be drawn on the given confidence level p_T. This problem can be solved in a number of ways:
  - The global confidence level p_T can be decreased. It is possible to find the largest value p_T ensuring sound conclusions and this value can be considered as the confidence level of the whole tree
  - It is possible to define the required confidence level locally for each node

Result Interpretation

- As a result of the computations, `Outcome` of the root node is found as an estimation $\mathcal{X} = (p_X, k_1, k_2)$. There are three possible major cases:
  - $0 < k_1 < k_2$ – the vulnerability level is *high*;
  - $k_1 < k_2 < 0$ – the vulnerability level is *low*;
  - $k_1 \leq 0 \leq k_2$ – the vulnerability level is *medium*
    - $\frac{k_1+k_2}{2} < 0$ – the vulnerability level is *lower medium*
    - $\frac{k_1+k_2}{2} > 0$ – the vulnerability level is *higher medium*

Further directions

- More case studies
- An analysis tool
- Tree computations are known to be imprecise
  - We use Gains in every internal node, even though the attacker gets the whole gain after the primary threat has been materialized
  - Precise computations can not be done as tree computations – we would need to consider all the subsets of the leaf set
  - Can this work be optimized?

Thank You!

Questions?