Variability meets Security
Quantitative Security Modeling and Analysis
of Highly Customizable Attack Scenarios

Maurice ter Beek
Formal Methods and Tools lab, ISTI–CNR, Pisa, Italy

Axel Legay Alberto Lluch Lafuente Andrea Vandin
UCLouvain, BE DTU, DK Sant’Anna, Pisa, IT

VaMoS 2020
Magdeburg, Germany
5 February 2020
Outline

Variability meets Security
  QFLan framework spin-off

Domain Specific Language (DSL) and tool
  Configurable attack-defense diagrams
  Probabilistic attack behavior
  Quantitative SMC analysis (over time)

Conclusion

Vision and Roadmap
Schneier’s seminal safe lock scenario

Schneier on Security

Attack Trees

B. Schneier

Dr. Dobb's Journal, December 1999.

Modeling security threats

By Bruce Schneier

Few people truly understand computer security, as illustrated by computer-security company marketing literature that touts "hacker proof software," "triple-DES security," and the like. In truth, unbreakable security is broken all the time, often in ways its designers never imagined. Seemingly

Search

Powered by DuckDuckGo

Subscribe

About Bruce Schneier
QFLan spin-off


Variability meets Security

- Apply variability modeling and analysis techniques to security (attack trees \(\approx\) feature diagrams)
- Overall aim:
  - Graphical representations for attack scenarios (set of products)
  - Analyse the feasibility of an attack scenario on a specific system
- This paper:
  - Illustrate DSL + tool on example scenario from security domain
  - DSL syntax + semantics and tool details in forthcoming papers
- Highly customizable attack scenarios
- Types of quantitative analysis (SMC):
  - Average cost and probability of success of attacks
  - Effectiveness of defenses and countermeasures
- Conclude with a vision and roadmap for future research
Variability meets Security

- Apply variability modeling and analysis techniques to security (attack trees \(\approx\) feature diagrams)
- Overall aim:
  - Graphical representations for attack scenarios (set of products)
  - Analyse the feasibility of an attack scenario on a specific system
- This paper:
  - Illustrate DSL + tool on example scenario from security domain
  - DSL syntax + semantics and tool details in forthcoming papers
- Highly customizable attack scenarios
- Types of quantitative analysis (SMC):
  - Average cost and probability of success of attacks
  - Effectiveness of defenses and countermeasures
- Conclude with a vision and roadmap for future research
Variability meets Security

- Apply variability modeling and analysis techniques to security (attack trees ≈ feature diagrams)
- Overall aim:
  - Graphical representations for attack scenarios (set of products)
  - Analyse the feasibility of an attack scenario on a specific system
- This paper:
  - Illustrate DSL + tool on example scenario from security domain
  - DSL syntax + semantics and tool details in forthcoming papers
- Highly customizable attack scenarios
- Types of quantitative analysis (SMC):
  - Average cost and probability of success of attacks
  - Effectiveness of defenses and countermeasures
- Conclude with a vision and roadmap for future research
Schneier’s safe lock extended

Equip attack trees with defenses and countermeasures and behavior (attack-defense diagrams)

- Attack nodes need to be explicitly activated by attack behavior
- Static defenses and dynamic countermeasures
DSL: attack, defense, and countermeasure nodes

begin attack nodes
  OpenSafe BruteForce OpenLock GuessPIN LearnPIN
  FindOwner ForceOwner LaserCutter
end attack nodes

begin defense nodes
  Reinforced
end defense nodes

begin countermeasure nodes
  Lockdown = { GuessPIN }
end countermeasure nodes
DSL: hierarchical relations

begin attack diagram
  OpenSafe  ->  { BruteForce, OpenLock }  
  OpenLock  ->  { GuessPIN, LearnPIN }   
  LearnPIN  -OAND-  [ FindOwner, ForceOwner ]  
  BruteForce  ->  { Reinforced }  
  Reinforced  ->  { LaserCutter }  
  OpenLock  ->  { Lockdown }  
end attack diagram

- Defense nodes cannot be refined
- Countermeasures can be refined: reactive defense nodes become effective upon (attack detection and) countermeasure activation
begin attributes
   Cost = { LaserCutter = 200, FindOwner = 20,
            ForceOwner = 10, Reinforced = 250 }
end attributes

- Attributes like cost or detection rate allow quantitative analysis and to impose constraints, like the maximum cost (default value is 0, i.e. Cost(GuessPIN) = 0)
- Also for defense nodes
DSL: attack detection rates

```
begin attack detection rates
  BruteForce = 0.5, OpenLock = 0.1, GuessPIN = 0.8,
  LearnPIN = 0.3, FindOwner = 0.6, ForceOwner = 0.7,
  LaserCutter = 0.6
end attack detection rates
```

- Determine the probability for attack attempts to be detected; attack attempt is the execution of `succ(·)` or `fail(·)` action (default value is 0, i.e. an attack is undetectable)
- Detection triggers the activation of affected countermeasures; higher detection rates mean more likely activations
In security nothing is ever 100% secure: specify effectiveness of a defensive node against any combination of attack nodes and attack behavior; probability of how likely an attack is thwarted (default value is 0, i.e. the defense has no effect)

Different attackers might be affected differently by a defense, even when attempting the same attack (e.g. a security guard is effective against a thief, but not against a military attack)
DSL: (probabilistic) attack behavior

- Defensive behavior is reactive while an attacker is proactive.
- Fine tune a security scenario by defining attack behavior, implicitly constrained by the attack-defense diagram.
- Explicit attack behavior allows for novel types of analysis that complement the classical best- and worst-case evaluations of attack trees (e.g. bottom-up evaluation in well-known ADTool).

- QFLan spin-off: each (optional) feature of a configurable system is a node which is installed by a configurator process similarly to how attack nodes are added by an attacker.
- Extended with different types of nodes and refinement, additional notions corresponding to defense effectiveness, attack detection rates, fail(·) actions, transition guards.
DSL: (probabilistic) attack behavior

- Defensive behavior is reactive while an attacker is proactive
- Fine tune a security scenario by defining attack behavior, implicitly constrained by the attack-defense diagram
- Explicit attack behavior allows for novel types of analysis that complement the classical best- and worst-case evaluations of attack trees (e.g. bottom-up evaluation in well-known ADTool)

- QFLan spin-off: each (optional) feature of a configurable system is a node which is installed by a configurator process similarly to how attack nodes are added by an attacker
  - Extended with different types of nodes and refinement, additional notions corresponding to defense effectiveness, attack detection rates, fail(·) actions, transition guards
DSL: (probabilistic) attack behavior

(e.g. from start to stratGP with probability $\frac{1}{1+2+10}$)
DSL: user-defined and built-in actions

(user-defined actions)

begin actions
  choose try success
end actions

- Built-in actions, like `succ(·)` or `fail(·)`, denote the success or failure of an attack attempt
- Attack attempts are modeled by probabilistic choice between `succ(·)` and `fail(·)` actions

Transition weights determine the success likelihood, together with (the effectiveness of) the involved defenses
"the accumulated cost of an attack may not exceed 250" and
"an attacker may not attempt more than 15 attacks"

begin quantitative constraints
{ sum(Cost) < 250 }
{ AttackAttempts < 15 }
end quantitative constraints

"an attacker is not allowed to attempt using a laser cutter
if the cost already exceeded 100"

begin action constraints
 do(succ(LaserCutter)) -> {sum(Cost) < 100}
 do(fail(LaserCutter)) -> {sum(Cost) < 100}
end action constraints
DSL: hierarchical, quantitative, and action constraints

“the accumulated cost of an attack may not exceed 250” and
“an attacker may not attempt more than 15 attacks”

begin quantitative constraints
{ sum(Cost) < 250 }
{ AttackAttempts < 15 }
end quantitative constraints

“an attacker is not allowed to attempt using a laser cutter
if the cost already exceeded 100”

begin action constraints
do(succ(LaserCutter)) -> {sum(Cost) < 100}
do(fail(LaserCutter)) -> {sum(Cost) < 100}
end action constraints
DSL: real-valued variables

```
begin variables
  AttackAttempts = 0
end variables
```

- Model context information: greatly facilitates analysis phase
- Updated as side effects when executing the specification, i.e. memory updates which label transitions
begin attacker behavior
begin attack
attacker = clever
states = start, finish, stratForce, stratF_OpenSafe, stratF_BruteForce, stratF_LaserCutter, stratLearnPIN, stratLP_OpenSafe, stratLP_OpenLock, stratLP_LearnPIN, stratLP_FindOwner, stratLP_ForceOwner, stratGuessPIN, stratGP_GuessPIN, stratGP_OpenSafe, stratGP_OpenLock
transitions =
//Pick a strategy:
start -(choose, 1) -> stratGuessPIN,
start -(choose, 2) -> stratForce,
start -(choose, 10) -> stratLearnPIN,
//Strategy GuessPIN:
(continued on next slide)
DSL: attack behavior (2/2)

(continuation of previous slide)

//Strategy GuessPIN:
stratGuessPIN -(try, 1, allowed(GuessPIN) and
!has(GuessPIN)) -> stratGP_GuessPIN,
stratGP_GuessPIN -(succ(GuessPIN), 1, {AttackAttempts=
AttackAttempts+1}) -> stratGuessPIN,
stratGP_GuessPIN -(fail(GuessPIN), 15, {AttackAttempts=
AttackAttempts+1}) -> stratGuessPIN,
...
stratGuessPIN -(success, 100, has(OpenSafe)) -> finish
//Strategies Force and LearnPIN ...

end attack
end attacker behavior
DSL: init

begin init
  clever = { FindOwner }
end init

- Attack behavior is completed by specifying the attacker used and pre-accomplished attacks
  (enriches expressiveness: assign initial advantage to attacker)
DSL: quantitative analysis

```
begin analysis
  query = eval when { AttackAttempts == 1 } : { OpenSafe, BruteForce, OpenLock, GuessPIN, LearnPIN, FindOwner, ForceOwner, LaserCutter, steps[delta = 0.5] }
  default alpha = 0.05 delta = 0.1 parallelism = 1
end analysis
```

⇒ Probability for each attack to be attempted first and succeed plus the average steps performed to attempt the first attack?

<table>
<thead>
<tr>
<th>Open Safe</th>
<th>Brute Force</th>
<th>Open Lock</th>
<th>Guess PIN</th>
<th>Learn PIN</th>
<th>Find Owner</th>
<th>Force Owner</th>
<th>Laser Cutter</th>
<th>steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.013</td>
<td>0</td>
<td>0.013</td>
<td>0</td>
<td>1</td>
<td>0.27</td>
<td>0.056</td>
<td>3</td>
</tr>
</tbody>
</table>
begin analysis
query = eval when { AttackAttempts == 1 } : { Open-Safe, BruteForce, OpenLock, GuessPIN, LearnPIN, Find-Owner, ForceOwner, LaserCutter, steps[delta = 0.5] } default alpha = 0.05 delta = 0.1 parallelism = 1 end analysis

⇒ Probability for each attack to be attempted first and succeed plus the average steps performed to attempt the first attack?

<table>
<thead>
<tr>
<th>Open</th>
<th>Brute</th>
<th>Open</th>
<th>Guess</th>
<th>Learn</th>
<th>Find</th>
<th>Force</th>
<th>Laser</th>
<th>steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td>Force</td>
<td>Lock</td>
<td>PIN</td>
<td>PIN</td>
<td>Owner</td>
<td>Owner</td>
<td>Cutter</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.013</td>
<td>0</td>
<td>0.013</td>
<td>0</td>
<td>1</td>
<td>0.27</td>
<td>0.056</td>
<td>3</td>
</tr>
</tbody>
</table>
DSL: quantitative analysis (over time)

⇒ Analyse properties over time: from 9 to 500 properties!

begin analysis

query = eval from 1 to 50 by 1 : { OpenSafe, Brute-Force, OpenLock, GuessPIN, LearnPIN, FindOwner, ForceOwner, LaserCutter, Lockdown, Reinforced }

default alpha = 0.05 delta = 0.1 parallelism = 1

end analysis

⇒ Probability for each attack that is attempted successfully and probability of activating the two defensive nodes
Results: probability of successful attacks (over time)
Results: probability of active defensive nodes (over time)
begin model OpenSafe
  begin variables
    AttackAttempts = 0
  end variables
  begin attack nodes
    OpenSafe BruteForce OpenLock GuessPIN LearnPIN LaserCutter ForceOwner FindOwner
  end attack nodes
  begin defense nodes
    Reinforced
  end defense nodes
  begin countermeasure nodes
    Lockdown = { GuessPIN }
  end countermeasure nodes
  begin actions
    tryAction
  end actions
  begin attributes
    Cost = { LaserCutter = 200, ForceOwner = 10, FindOwner = 20, Reinforced = 250 }
  end attributes
  begin attack detection rates
end model OpenSafe
Conclusion

[TSE18]:
- Integrated modeling + analysis approach for configurable systems
- Illustrated flexibility by application to a confined security scenario
  ⇒ limitations requiring intermediate encoding into security notions

[VaMoS20]:
- Special-purpose DSL
- Quantitative analysis
  ⇒ illustrate DSL + tool on example scenario from security domain

Future twin papers:
  ⇒ syntax and semantics
  ⇒ full tool presentation
Conclusion

[TSE18]:
- Integrated modeling + analysis approach for configurable systems
- Illustrated flexibility by application to a confined security scenario
  ⇒ limitations requiring intermediate encoding into security notions

[VaMoS20]:
- Special-purpose DSL
- Quantitative analysis
  ⇒ illustrate DSL + tool on example scenario from security domain

Future twin papers:
  ⇒ syntax and semantics
  ⇒ full tool presentation
Conclusion

[TSE18]:
- Integrated modeling + analysis approach for configurable systems
- Illustrated flexibility by application to a confined security scenario
⇒ limitations requiring intermediate encoding into security notions

[VaMoS20]:
- Special-purpose DSL
- Quantitative analysis
⇒ illustrate DSL + tool on example scenario from security domain

Future twin papers:
⇒ syntax and semantics
⇒ full tool presentation
Vision and Roadmap

⇒ more fine-grained correlation of steps (temporal logic $\Diamond$ and $\Box$)
  ▶ “if an attack step (or, in fact, any step) A is performed, then defense step B is performed in less than $X$ units of time”

⇒ synthesize the best attacker (Uppaal Stratego)
  ▶ “all attackers with feature A satisfy a given property”

⇒ more generic properties by requirements in full LTL
  ▶ “if a person has the authorization to enter and wants to enter, then (s)he will eventually enter”

⇒ try to derive attackers with the best chance of success (e.g. leave some weights off the edges and rather derive them)
Workshop **QAVS’20** on Quantitative Aspects of Variant-rich Systems:

- Design of performance in feature-oriented systems
- Modeling of reliability in product lines
- Implementation costs in highly configurable systems
- Analysis of stochastic effects
- Verification

PC chairs: Maurice ter Beek and Clemens Dubslaff (TU Dresden, DE)

⇒ submission deadline: 12 June 2020
The aim of the FMICS conference series is to provide a forum for researchers who are interested in the development and application of formal methods in industry. FMICS brings together scientists and engineers who are active in the area of formal methods and interested in exchanging their experiences in the industrial usage of these methods. The FMICS conference series also strives to promote research and development for the improvement of formal methods and tools for industrial applications.

FMICS 25:
This 25th edition of FMICS will be celebrated in a special way. Topics of interest include (but are not limited to):

- Case studies and experience reports on industrial applications of formal methods, focusing on lessons learned or identification of new research directions.
- Methods, techniques and tools to support automated analysis, certification, debugging, descriptions, learning, optimisation and transformation of complex, distributed, real-time, embedded, mobile and autonomous systems.
- Verification and validation methods (model checking, theorem proving, SAT/SMT constraint solving, abstract interpretation, etc.) that address shortcomings of existing methods with respect to their industrial applicability (e.g., scalability and usability issues).
- Impact of the adoption of formal methods on the development process and associated costs. Application of formal methods in standardisation and industrial forums.

NEW THIS YEAR:
SPECIAL TRACK on "Formal Methods for Security in IoT"
We invite submissions in topics related to the secure development and security assessment of IoT-based applications using formal methods.

FMICS'25th International Conference on Formal Methods for Industrial Critical Systems
2 - 3 September 2020 | Vienna | Austria

PC Chairs
Maurice ter Beek | ISTI-CNR
Dejan Ničković | AIT

PC Members
Bernhard Aichernig | TU Graz
Jiri Barnat | Masaryk University
Davide Basile | University of Florence & ISTI-CNR
Carlos Budde | University of Twente
Rance Cleaveland | University of Maryland & NSF
Thao Dang | VERIMAG
Michael Dierkes | Rockwell Collins
Georgios Fainekos | ASU
Alessandro Fantechi | University of Florence
Wan Fokkink | VU Amsterdam
Maria Del Mar Gallardo | University of Malaga
Ichiro Hasuo | University of Tokyo
Klaus Havelund | NASA
Thierry Lecomte | CLEARSY
Axel Legay | UCLouvain
Gabriele Lenzini | University of Luxembourg
Alberto Lluch Lafuente | DTU
Florian Lorber | Aalborg University
Tiziana Margaria | University of Limerick & Lero
Radu Mateescu | INRIA
Franco Mazzanti | ISTI-CNR
Stefan Mitsch | CMU
José N. Oliveira | University of Minho & INESC TEC
Jaco van de Pol | Aarhus University
Adam Rogaliewicz | Brno University of Technology
Markus Roggenbach | Swansea University
Matteo Rossi | Polytechnic University of Milan
Stefano Tonetta | FBK
Jan Tretmans | TNO
Andrea Vandin | Sant'Anna School of Adv. Studies
Tim Willemse | TU Eindhoven
Kirsten Winter | University of Queensland
Lijun Zhang | Chinese Academy of Science

Steering Committee
Alessandro Fantechi | University of Florence
Hubert Garavel | INRIA
Stefania Gnesi | ISTI-CNR
Diego Latella | ISTI-CNR
Tiziana Margaria | University of Limerick & Lero
Radu Mateescu | INRIA
Jaco van de Pol | Aarhus University

Important Dates
Abstract submission: May 8, 2020
Paper submission: May 15, 2020
Author notification: July 1, 2020
Camera-ready version: July 15, 2020

FMICS is part of QONFEST 2020
CONCUR | FMICS | FORMATS | QEST
August 31 – September 5, 2020

Keynote Speakers
Thomas Henzinger | IST Austria
CONCUR | QEST | FMICS Speaker

Stefan Resch | THALES
FMICS Speaker

Roderick Bloem | TU Graz
CONCUR | FORMATS | FMICS Speaker

Call for Papers
The aim of the FMICS conference series is to provide a forum for researchers who are interested in the development and application of formal methods in industry. FMICS brings together scientists and engineers who are active in the area of formal methods and interested in exchanging their experiences in the industrial usage of these methods . . .

New in 2020: special track on Formal Methods for Security in IoT