

Security Monitoring via Runtime Verification

Minh Van Nguyen

School of Computer Science and Mathematics
Victoria University

nguyenminh2@gmail.com

<http://nguyenminh2.googlepages.com>

03 April 2008

Need for Security

Runtime
Verification

Literature Review

Project Plan

Project Update

Conclusion

Contents

- ① Need for Security
- ② Runtime Verification
- ③ Literature Review
- ④ Project Plan
- ⑤ Project Update
- ⑥ Conclusion

Need for Security

- Built-in security
 - network: intranet, Internet
 - software: operating systems
 - hardware: microprocessors
- Causes
 - subtle requirements
 - unforeseen risks
 - problematic implementations
 - ad hoc development

Runtime Verification

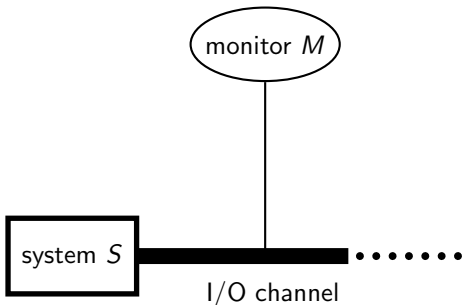


Figure: System monitoring during operation.

- Advantages
 - confidence in implementation
 - runtime information
 - behaviours dependent on operating environment
 - security critical systems

- Havelund & Rosu 2002
 - concept origin
- Havelund & Goldberg 2005; Colin & Mariani 2005
 - overview of runtime verification
- Schneider 2000
 - security monitoring with Büchi automata
- Bauer, Leucker & Schallhart 2006
 - 3-valued semantics monitor
- Bauer & Jürjens 2008
 - monitoring security protocols

Project Plan

- Temporal logic (LTL) & runtime verification
- Security critical systems & their monitoring
- Realization of RV framework
 - `1t12mon` utility
- Evaluate framework w.r.t SSL implementation
 - identify security properties
 - map properties to LTL specifications

Project Update

- Bauer, Leucker & Schallhart 2006 — 3-valued semantics monitor

φ	$\neg\varphi$	value
SAT	SAT	?
SAT	UNSAT	\top
UNSAT	SAT	\perp

Table: Satisfiability criteria in 3-valued LTL.

```
Terminal - mvngu@m
[mvngu@modular 2008-02-23]$ ./demo.sh
Spec. Phi:  a U b

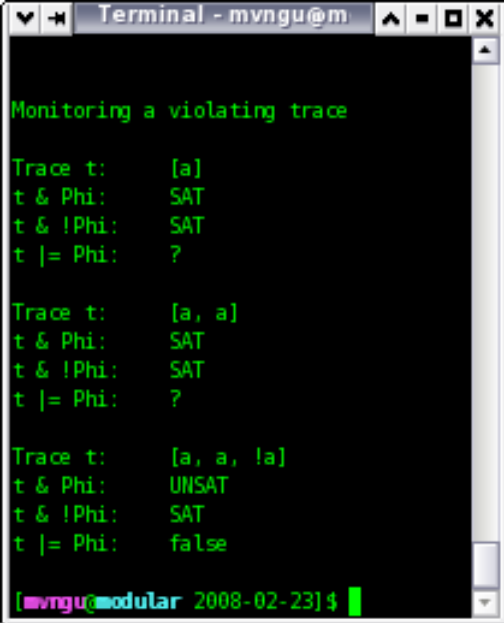
Monitoring a satisfying trace

Trace t:    [a]
t & Phi:    SAT
t & !Phi:   SAT
t |= Phi:   ?

Trace t:    [a, a]
t & Phi:    SAT
t & !Phi:   SAT
t |= Phi:   ?

Trace t:    [a, a, b]
t & Phi:    SAT
t & !Phi:   UNSAT
t |= Phi:   true
```

Figure: Monitoring a satisfying trace.



```
Terminal - mvngu@m
Monitoring a violating trace

Trace t:      [a]
t & Phi:     SAT
t & !Phi:    SAT
t |= Phi:    ?

Trace t:      [a, a]
t & Phi:     SAT
t & !Phi:    SAT
t |= Phi:    ?

Trace t:      [a, a, !a]
t & Phi:     UNSAT
t & !Phi:    SAT
t |= Phi:    false

[mvngu@modular 2008-02-23]$
```

Figure: Monitoring a violating trace.

Conclusion

- RV framework for security critical systems
- Evaluate using SSL case study
- Open source security monitor

References

- 1 A. Bauer & J. Jürjens. “Security Protocols, Properties, and Their Monitoring” in *The 4th International Workshop on Software Engineering for Secure Systems (SESS '08)*. Leipzig, Germany, May 17–18, 2008.
- 2 A. Bauer, M. Leucker & C. Schallhart. “Monitoring of Real-Time Properties” in *FSTTCS 2006: Foundations of Software Technology and Theoretical Computer Science*. LNCS 4337, Springer, 2006, pp.260–272.
- 3 S. Colin & L. Mariani. “Run-Time Verification” in *Model-Based Testing of Reactive Systems*. LNCS 3472, Springer, 2005, pp.525–555.
- 4 K. Havelund & A. Goldberg, 2008-04-03
<http://www.havelund.com/Publications/vstte05.pdf>
- 5 J. Jürjens, Open University, UK, 2008-03-31
<http://mcs.open.ac.uk/jj2924/research>
- 6 F.B. Schneider. “Enforceable Security Policies” in *ACM Transactions on Information and System Security*. ACM, vol.3, no.1, Feb. 2000, pp.30–50.

Thank You