DSN 2009 - The 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks
June 29 – July 2 Estoril, Lisbon Portugal

CITY UNIVERSITY LONDON

Enhancing Fault/Intrusion Tolerance through Design and Configuration Diversity

Alysson Bessani, Alessandro Daidone,
Ilir Gashi, Rafael Obelheiro, Paulo Sousa,
Vladimir Stankovic

29th June 2009, Lisbon, Portugal

FOREVER

CRUTIAL
CRitical UTility InfrastructurAL resilience

FOREVER

CITY UNIVERSITY LONDON

Outline

- Team and motivation
- FOREVER service
- Configuration diversity rules
- Preliminary assessment of the FOREVER service
- Conclusions and future work

FOREVER CITY UNIVERSITY LONDON

The Team

- FOREVER – a mini-project inside the EU NoE ReSIST http://www.resist-noe.org
- Institutions
  - Universidade de Lisboa (Portugal)
  - City University London (UK)
  - Università di Pisa (Italy)
  - Universidade do Estado de Santa Catarina (Brazil) – Affiliate member
  - Universität Erlangen-Nürnberg (Germany) – Affiliate member
- People
  - Alysson Bessani @ Lisboa
  - Hans Reiser @ Lisboa
  - Paulo Sousa @ Lisboa
  - Ilir Gashi @ City
  - Vladimir Stankovic @ City
  - Alessandro Daidone @ Pisa
  - Rafael Obelheiro @ Santa Catarina
  - Tobias Distler @ Erlangen-Nürnberg
  - Rüdiger Kapitza @ Erlangen-Nürnberg

FOREVER CITY UNIVERSITY LONDON

Addressing WRAITS Topics

- Topics:
  - automatic recovery and response techniques
  - use of Byzantine fault-tolerant algorithms in IT
  - diversity and failure independence
- *"The workshop will be especially interested in 'practical intrusion-tolerant systems'"*
- *"How to build information systems that are inherently resilient to intrusions"*
- *"Papers can present ongoing work and/or speculative/futuristic ideas"*

Motivations

- Fault/intrusion tolerance is commonly the only viable way of improving the system dependability and (possibly) security.
- Designers are in need of advice about the architectural evolution of fault/intrusion-tolerant systems
  – Many solutions exist, but the current solutions concern (only) a specific snapshot in time, without considering how the system should evolve.
- Fault/intrusion tolerant systems need to counteract the evolution of threats and exploits against a running system.

Motivations (cont.)

- The defensive measures in the current practice are largely *reactive.*
  – E.g. issuance of patches
  – "at-risk-time" is (unnecessarily) long
- Proactive methods are needed to counteract the evolution of faults, threats and vulnerabilities.
  – With proactive methods, design novelty is used as a defence.
- We need a unifying, complementary approach: reactive approaches ought to be enhanced with a more dynamic response to evolving threats and vulnerabilities.

The FOREVER Service

(**F**ault/intrusi**O**n **RE**mo**V**al through **E**volution & **R**ecovery)

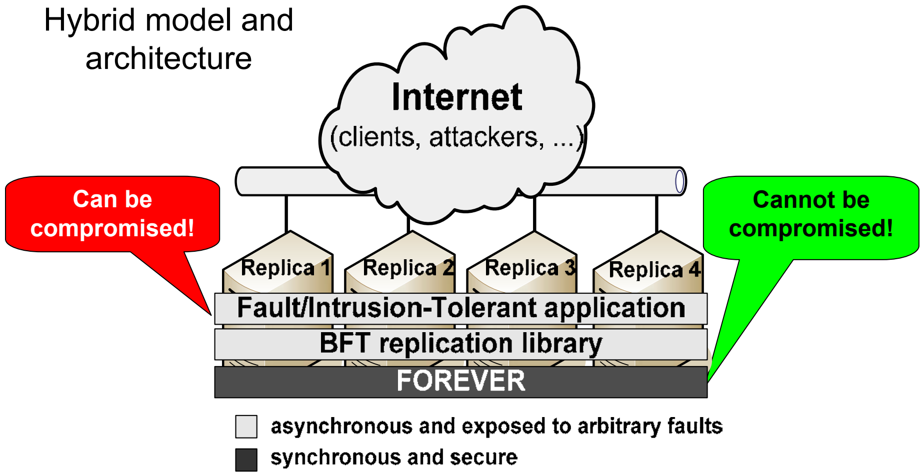
- Byzantine fault-tolerant protocols against arbitrary failures have limited utility
  - They are "only" useful to delay corruptions.
- While a higher number of replicas allows for tolerating more compromised replicas, it also means that there are more individual replicas that need to be different from each other.
  - E.g., 3f+1 and f=1 → 4 (diverse) replicas needed
  - But, 3f+1 and f=2 → 7 (diverse) replicas needed.
- FOREVER's main goal – increasing the number of tolerated arbitrary faults/vulnerabilities, without increasing the number of replicas.

The FOREVER Service (cont.)

- The ambitious goal is to be achieved through
  - Recovery
    - Event-triggered (when malicious behavior is detected or suspected)
    - Time-triggered (every replica is rejuvenated periodically)
  - Evolution
    - Through the use of **configuration diversity rules**
    - Recovered replicas are different from previous incarnations
    - This makes FOREVER *different* from the other recovery services.

The FOREVER Architecture

Hybrid model and architecture



Diversity Management

- **Offline** diversity generation
  - Pool of pre-built OS images (e.g., Linux, OpenBSD)
  - Different OS image started in each recovery
    - Recent studies showed some off-the-shelf sw have **low number of common faults/vulnerabilities**[1].
  - FOREVER selects the OS image that is **less similar** than the OS images running in the remaining replicas.
- **Online** diversity generation
  - FOREVER applies a set of **configuration diversity rules** to the selected OS image.

[1] A. N. Bessani, R. Obelheiro, P. Sousa, and I. Gashi: “*The effects of diversity on intrusion tolerance*”, DI-FCUL TR 08–30, Univ. of Lisbon, Dec. 2008.
I. Gashi, P. Popov, and L. Strigini: “*Fault tolerance via diversity for off-the-shelf products: A study with SQL database servers*”, IEEE TDSC 4(4):280–294, Oct.- Dec.2007.
B. Vandiver: “*Detecting and tolerating byzantine faults in database systems*”, MIT-CSAIL- TR-2008-040, MIT, June 2008.

FOREVER CITY UNIVERSITY LONDON

Configuration Diversity Rules

- We produced a set of configuration diversity rules, which aim to enhance the resilience of software in between recoveries[1]
- The rules were generated using three approaches
  - *Bottom-up* – scrutinising the implementations of the OSs and applications
  - *Top-down* – exploring reported vulnerabilities
  - *Lateral* – reviewing existing literature of configuration rules for protection against "malicious" behaviour

[1] Full description of rules can be found at http://www.csr.city.ac.uk/people/ilir.gashi/ConfigDiv/

FOREVER CITY UNIVERSITY LONDON

Configuration Diversity Rules (an excerpt)

ID	Rule name	Design Implication		Security Category
		Impl. intru-siveness	Client notifi-cation re-quired?	
1	Password change	B	Yes	C
2.1	Different authentication protocols	W, B or G	Yes	C
2.2	Different Trusted Third Parties	W, B or G	No	C and A
3	Different "factors" in n-factor authentication methods	W, B or G	Yes	C
4.1	Address Space Layout Randomisation (ASLR)	W	No	I
4.1.1	Pointer obfuscation	W	No	I
4.1.2	Randomisation of global variables and local vari-ables offsets	W	No	I
4.2	Address Space Partition-ing	W, B or G	No	I
4.3	Stack Frame Padding	W, B or G	No	I
4.4	Basic Block reordering	W, B or G	No	I
5.1	Instruction set randomi-sation	W, B or G	No	I
5.2	Instruction set tagging	W, B or G	No	I
5.3	Instruction Reordering	W, B or G	No	I
6.1	Diverse Linux User IDs (UID)	W, B or G	No	I and C
7	Change IP addresses of the hosts	B	Yes	C and A
8	Changing listening port numbers	W, B or G	Yes	C and A
9	Adding or deleting non-functional code	W, B or G	No	I
10.1	Varying dynamic li-braries and system calls	W	No	I
10.2	Varying unique names of system files	W	No	I
10.3	Varying magic numbers in certain files (e.g., ex-ecutables)	W	No	I

FOREVER Service Preliminary Evaluation

- Preliminary assessment of the potential to enhance system resilience.
- Focus on the evaluation of the probability of system failure, through variation of several parameters:
  - The time between recoveries
  - The probability of common vulnerabilities
  - The mean effectiveness of configuration diversity rule-set
- Initial values chosen are pessimistic
  - The common practice when evaluating safety- and reliability-critical systems and/or parameter values are unknown.
- The quantitative evaluation uses a modelling methodology which assumes Multiple Phased Systems.
  - see "I. Mura and A. Bondavalli: "*Markov regenerative stochastic Petri nets to model and evaluate the dependability of phased Missions*", IEEE Transactions on Computers 2001.

System Properties/Assumptions

- n = 4 replicas can tolerate up to f = 1 faults
  - $n \geq 3f+1$
  - Win2k, Linux, Solaris, FreeBSD
- Both "space" (design diversity) and "time" (configuration diversity rules) diversity is considered.
- Arbitrary faults/vulnerabilities; a binary state space: "failed", "OK".
- Common faults/vulnerabilities exist
  - The ratio value used is based on the study[1] using National Vulnerability Database (NVD) data
- Proactive, sequential, fault-free recovery strategy has been assumed.

[1] A. N. Bessani, R. Obelheiro, P. Sousa, and I. Gashi: "*The effects of diversity on intrusion tolerance*", DI-FCUL TR 08–30, Dec. 2008.

Measures of Interest and Parameters

- **p_F** - System failure probability, i.e., probability of having more than f failed replicas.
- **t** - Mission time; we are interested in investigating how system failure probability changes over time.
- **T_P** - Recovery period, based on the waiting time T_W between the recovery of replica *i* and the recovery of replica *i+1*.
  - $T_P = (T_R + T_W) \cdot n$
- **δ_λ** - Base value of the penalty of replica failure rate, if diversity is not applied.
  - Discovery of vulnerabilities is assumed to be progressing
- **δ_{ij}** - Probability of common faults/vulnerabilities among different replicas
  - Since n = 4, the overall system fails as soon as one common fault/vulnerability, affecting a pair of replicas, occurs.
- **δ_x** - Mean value for the effectiveness of the configuration diversity rule-set.

Parameter Values and Common Faults Prob.

Study	T_W	δ_λ	δ_x	δ_{ij}
1	{0, 360, 480, 840}	0	0	δ_{ij}
2	0	$\{0, 10^{-6}, 5x10^{-7}, 10^{-7}\}$	0.8	δ_{ij}
3	0	0	0	$\{0, 1, 5, 10\}x\delta_{ij}$
4	0	10^{-6}	{0, 0.2, 0.4, 0.6, 0.8, 1}	δ_{ij}

O.S.	Vulns	Common Vulnerabilities		
		Win2K	Linux	Solaris
FreeBSD	229	3	11	18
Solaris	411	3	5	
Linux	437	3		
Win2K	347			

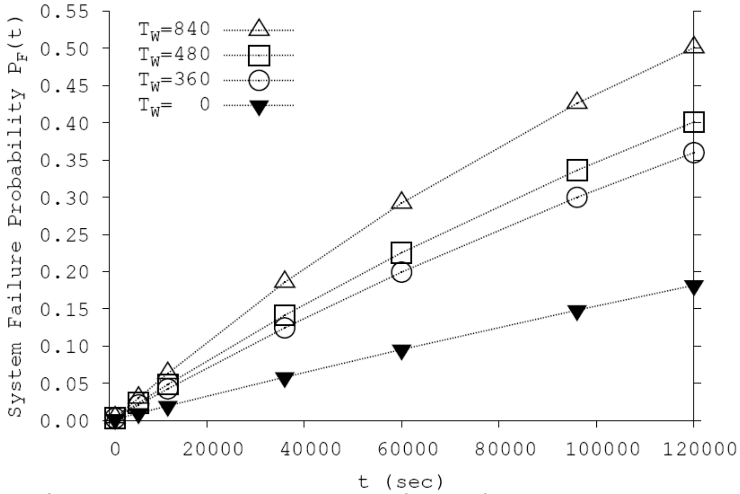
δ_{ij}		j			
		FreeBSD	Solaris	Linux	Win2K
i	FreeBSD	-	0.029	0.017	0.005
	Solaris	0.029	-	0.006	0.004
	Linux	0.017	0.006	-	0.004
	Win2K	0.005	0.004	0.004	-

FOREVER

CITY UNIVERSITY LONDON

Preliminary Evaluation Results (Study 1)

Waiting Time (T_w) Sensitivity Analysis



$\delta_x = 0$

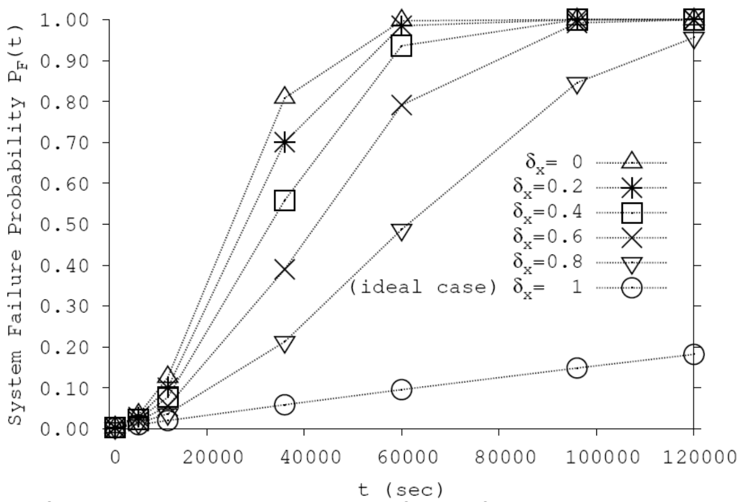
$\delta_\lambda = 0$

FOREVER

CITY UNIVERSITY LONDON

Preliminary Evaluation Results (Study 4)

Rule(s) Effectiveness (δ_x) Sensitivity Analysis



$T_w = 0$

δ_λ NOT changed

Diversity Configuration Rules - Design Implications

- Interface and input language complexity
- Possibility of introducing new faults/vulnerabilities
- Security issues
- Performance issues
- Data consistency issues

Conclusions

- We performed initial study about system evolution wrt. tolerating ever-growing and ever-present threats.
- We defined a set of configuration diversity rules to be used as part of the *“preventive”* FOREVER architecture
  - FOREVER uses both reactive and proactive recovery
- FOREVER uses online and offline diversity generation mechanisms
- Performed *preliminary* evaluation of the FOREVER

Future Work

- Extending configuration diversity rules
  - Possibility of using an alternative categorisation
- Developing proof-of-concept implementations
  - Possibility of using fault-injection
- Extending/improving the evaluation
  - Include the reactive recoveries; and thus e.g. explore trade-off between reactive and proactive strategies.
  - More precise evaluation of the effectiveness of the configuration diversity rules (currently, only a single value represents the effectiveness).
  - Possibility to compare with other (e.g., non-redundant) architecture
- Evaluating the effects on the system performance
  - Aiding the analysis of the "eternal" trade-off between the improvement of dependability/security and performance.

Thank you!