How Web Services can be Tolerant to Intruders through Diversification ?

Georges Ouffoué

Joint work with Fatiha Zaïdi, Ana R. Cavalli and Mounir Lallali

université PARIS-SACLAY

LABORATOIRE DE RECHERCHE EN INFORMATIQUE

Outline

Web services

- Web services allow the communication of heterogeneous systems through the Internet
- Web services are ubiquitous
- Cloud computing, a key factor
- In cloud computing, everything is offered as a service (Amazon, Google, Microsoft)

Research on Web services

A lot of research has been made over the decades to enhance the trust of Web services

- Web services Verification
- Web services Composition
- Web services Adaptation
- Web services Fault-Tolerance

Limitations of the state of the art

- Web services are the targets of malicious attackers;
- Existing solutions are mainly attack-specific.
- No evidence of their protection against harmful silent attacks
- Simple services must continue to respond to the requests of the consumers even in the presence of non-critical attacks

A distributed system is attack tolerant if it can continue to perform its main task even with a degraded level in the presence of attacks.

Attack tolerance

Our idea

In the case an attack has successfully occurred, on the fly we change the implementations of our Web service that seem attacked, in order to mitigate the effect of the attack.

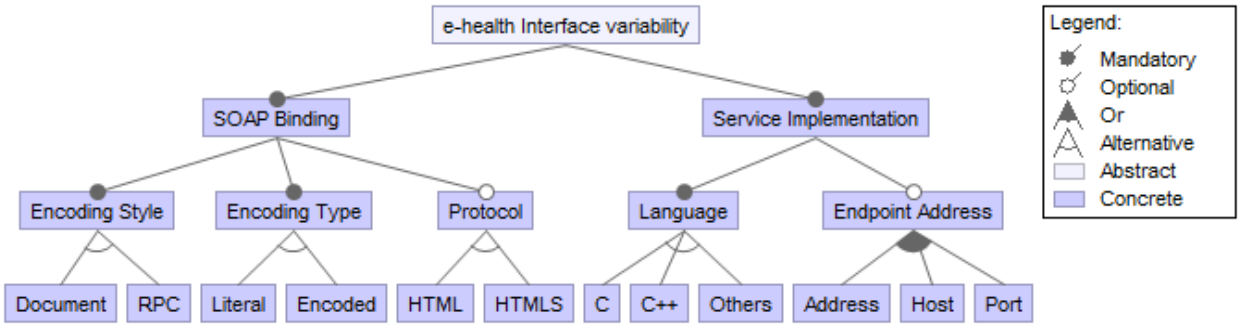
This approach is based on:

- Software Products Line (SPL) $\Rightarrow$ Definition of the Feature Model and the variability points(Encoding style/type, Languages)
- Diversity,having different forms, types, ideas... $\Rightarrow$ Fault Tolerance
- Monitoring $\Rightarrow$ Detection of attacks and reaction ;

Methodology(1)

We illustrated the approach with a Web service that simplifies the management task in a hospital. The methodology is the following:

- Definition of the Feature Model and the variability patterns.



- Generation of the services with gSOAP
- Diversification of the binaries by multicompilation

Methodology(2)

Configuration of the system in two ways :

1. *Prevention mode*. Time is divided into slots called epochs. In each epoch only one variant is chosen at all; another implementation is deployed after the slot elapsed;
2. *Attack mode*. Reaction by switching to another implementation before the epoch elapsed.

Experiments

- Illustration of the approach with an e-health Web service .
- Diversified variants obtained with RPC/Document encoding styles, Literal/Encoded encoding types, multicompiler (25, 50, 75 %of obfuscation), C/C++ languages;
- Testing with a DDoS attack and a cache based-side channel attack to check the attack tolerance.

Our Monitoring tool

We used MMT (Montimage Monitoring Tool) of which a property is an IF-THEN expression that describes constraints on network events captured in a trace $T = \{p_1, ..., p_m\}$. It has the following syntax:

$$e_1 \xrightarrow{W,n,t} e_2$$

W $\in$ { BEFORE, AFTER }, n $\in \mathbb{N}$, t $\in \mathbb{R}_{>0}$ and e_1 and e_2 two events. If the event e_1 is satisfied (by one or several packets p_i , i $\in \{1, ..., m\}$, then event e_2 must be satisfied (by another set of packets p_j , j $\in \{1, ..., m\}$) before or after (depending on the W value) at most n packets and t units of time. e_1 is called triggering context and e_2 is called clause verdict.

MMT DDoS Detection Metrics

Metric / type of alarm	# of request per second	Response time	# of distinct IPs	Bandwidth usage	CPU/memory usage	Storage operations
Warning	> 2 x threshold*	> 2 x average response time	> 2 x last 24 hours range	> 0,5 x bandwidth capacity	> 0,7 capacity	1 active request & storage inactive
Alarm	> 10 x threshold*	> 4 x average response time	> 4 x last 24 hours range	> 0,8 x bandwidth capacity	> 0,9 capacity	> 1 active request & storage inactive

Figure 1: MMT DDoS Detection Metrics

DDoS attack

A DDoS attack were launched against our Web service using LOIC (Low Orbit Ion Cannon)

Table 1: Time elapsed to complete a client request in the presence of a DDoS attack

Nb. of packets/s	Before (seconds)	After (seconds)
1000	4.45	5.93
2000	5.25	6.91
3000	8.09	9.05
4000	8.34	9.51
5000	9.72	11

From Table 1 the framework seems highly reactive. In fact, the time to access to the Web service after the change of implementation is almost the same as before the approach.

Side channel attack

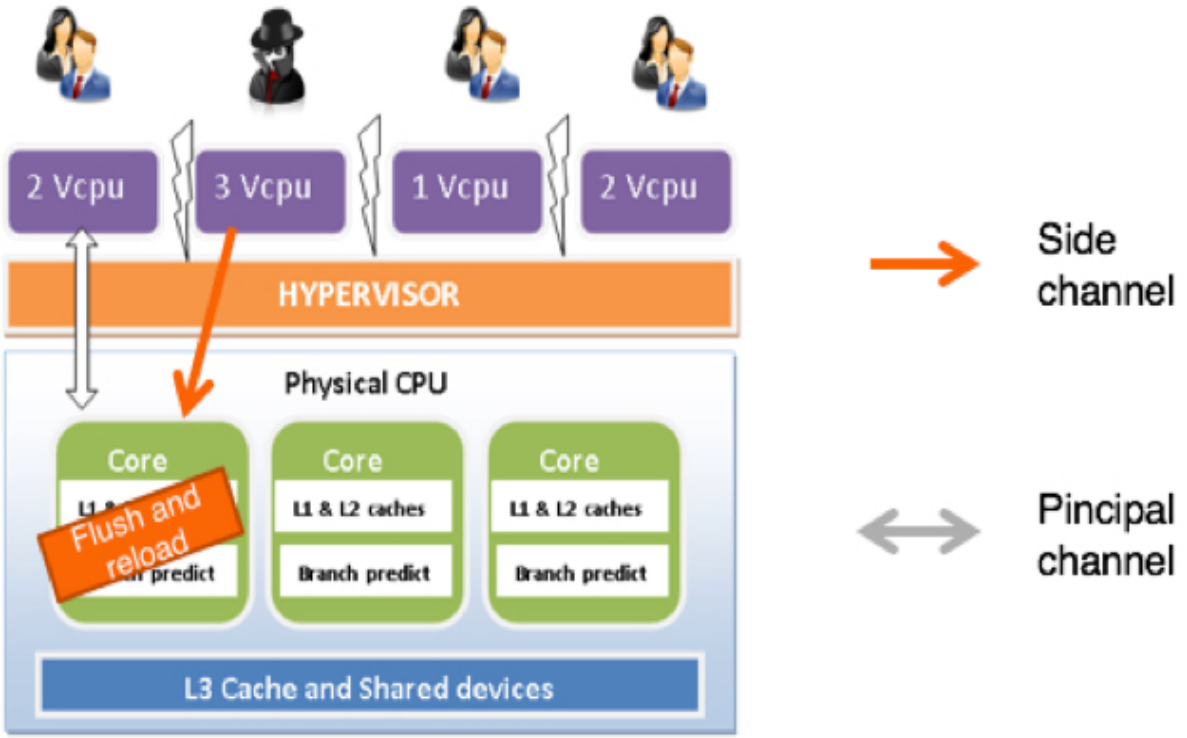


Figure 2: Overview of a side channel attack on a cloud platform.

Remediation: Side channel attack

Diversification is effective in countering this attack because:

- a side-channel attack requires a fine evaluation of the memory footprint of the operations and the data control flow of the program.
- With diversification, when we changed the implementation of our web service, the binaries are different and then the fingerprints of the memory used by the program are also different.
- Applications not using too much the resources $\Rightarrow$ 10 % performance overhead. In the cloud, few excessively CPU-intensive applications.

Summary

- Attack tolerance for Web Services: an open challenge
- A solution: Diversity-based attack tolerance framework
- Experiments on an e-health Web service, confirm the effectiveness of the approach to enable attack-tolerance.

Future works

- Enhancement of the expressiveness of the monitoring tool to detect behavioral attacks
- Automation of the diversification step leveraging Software Product Lines
- Investigation of other diversity axes
- Formalization of attack tolerance

Thank you