

Sphinx: an anomaly-based Web Intrusion Detection System


Damiano Bolzoni & Emmanuele Zambon
University of Twente, The Netherlands



Agenda

- Overview
 - Web applications (in)security
- Current countermeasures
 - What are the scientists doing ?
- The definitive WIDS properties
- Our implementation
 - Training
 - Detection engine
 - Benchmarks
 - Generating signatures

Overview

- Web applications have become really popular nowadays
 - CMSs & BBSs
 - e-commerce & e-banking
 - Microsoft Live Search,  Google Docs & Spreadsheets BETA
- New technologies support web applications popularity
 - back-end on DBMSs (2001)
 - Java/ActionScript (2003)
 - AJAX (2005)
- Web applications' content is particularly attractive and this makes them one of the most interesting targets

Web applications (in)security

- Delivering a secure software is not easy: e.g. people with different skills contribute in open-source software
- From Symantec's "Internet Security Threat Report" (09/2006):
 - web application vulnerabilities account for 69% of total in Jan-Jun 2006
 - 78% of *easily exploitable vulnerabilities* are related to web applications (e.g. SQL Injection, XSS)
- SecurityFocus' Bugtraq is flooded every day with brand-new web application bugs

Current countermeasures

- NIDSs → Snort has more than 1200 signatures related to web attacks
 - SSL traffic cannot be easily analyzed
 - still playing “*cops and robbers*” with 0-day attacks
- Web Application Firewalls (WAFs) → *ModSecurity* is the most world-wide deployed WAF
 - Apache web server module → overcomes problems with SSL/POST data
 - imports Snort signatures as well as user-defined ones → deployment can become a complex task
 - few anomaly-detection capabilities on HTTP protocol fields only



What are the scientists doing ?

- Tools supporting source-code auditing [JKK06]
 - this is not always easy/possible (e.g. third-party CGIs/Java Servlets)
- Web server logs analysis [RVKA06]
 - recognizes the importance of parameter-based analysis
 - POST data can be missing/logs could be tampered
- Tailored solutions to catch SQL Injection/XSS attacks [VMV05]
 - source code needed/modified DBMS libraries to analyze incoming queries
→ what about, e.g., SQL Server ?
- Long text-based field (e.g. BBS messages/emails) are usually difficult to be analyzed

The definite WIDS (hopefully ;-)

- Let's sketch the desired properties of a hypothetical Web Intrusion Detection System (WIDS)

- Problem-free for :

- SSL/POST data
- unavailable source code
- logs

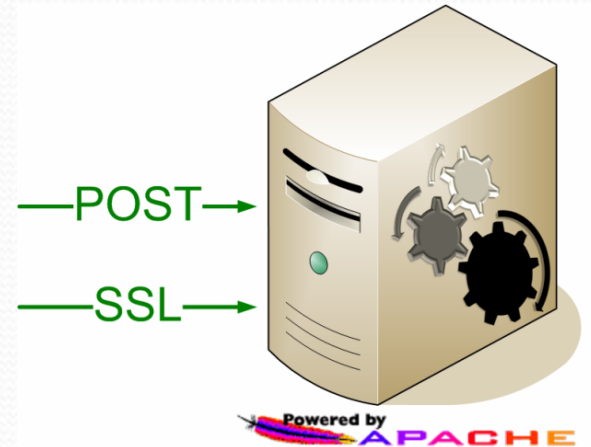


- Anomaly-detection on data and self-instructing, at least partially
 - 0-day attacks could be more easily detected
 - signatures could be automatically generated for certain query parameters to discover attacks → easier deployment

Just a “midsummer night’s dream” ?

Our implementation

- Implemented as an Apache web server module
 - fully anomaly-based
 - analyzes SSL/POST data



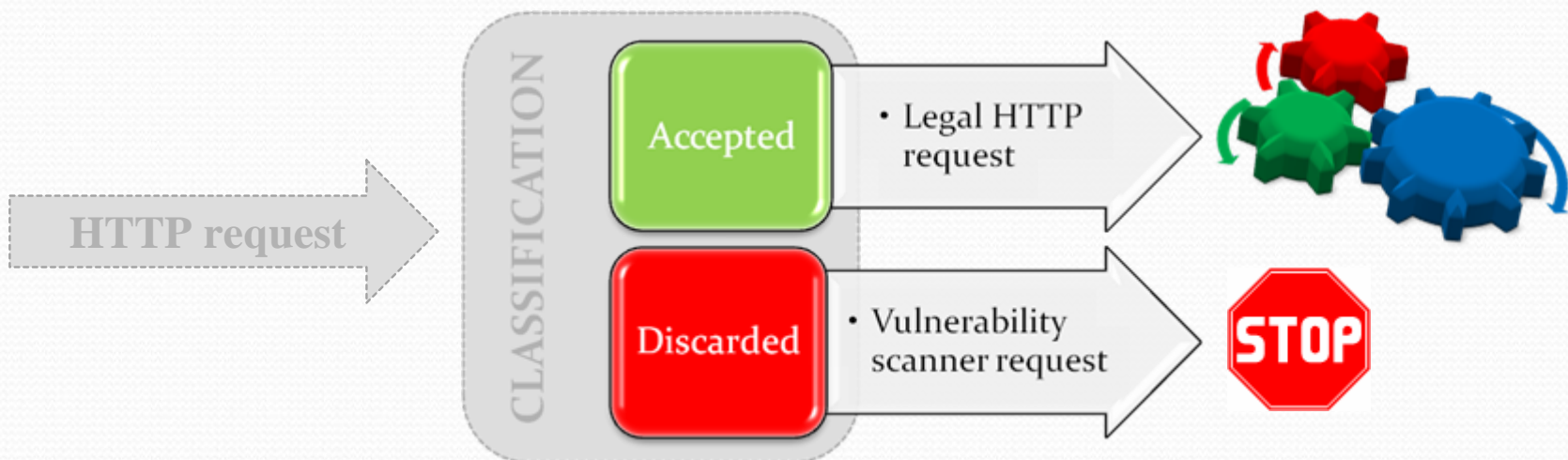
- Different techniques are combined to increase detection and accuracy rates
 - pre-processing phase to clean the training data
 - parameter content analysis based on type → deals with long text content too
- Generates signatures in an automatic way for *ModSecurity*
 - makes deployment easier

Data pre-processing - 1

- Since the engine is anomaly-based, we need to train the WIDS before being able to detect attacks
- Nowadays a huge number of BOTnets and automatic scanners are constantly “monitoring” subnets, searching for well-known vulnerabilities
- This “noise” should be somehow avoided, since it affects the detection and the accuracy rates
 - in this situation, some information can be extracted to detect scanning activities

Data pre-processing - 2

- Pre-processing the training data makes possible to improve the performances (and size) of the anomaly engine models:
 - few parameters (received/sent bytes, response, duration)
 - low detection rate, but it suits our *pre-processing* purpose
 - typically discovers web scanner probes



What are we going to analyze ?

- A query has a typical structure (from the CMS PostNuke)

GET /modules.php?name=News&file=article&sid=25

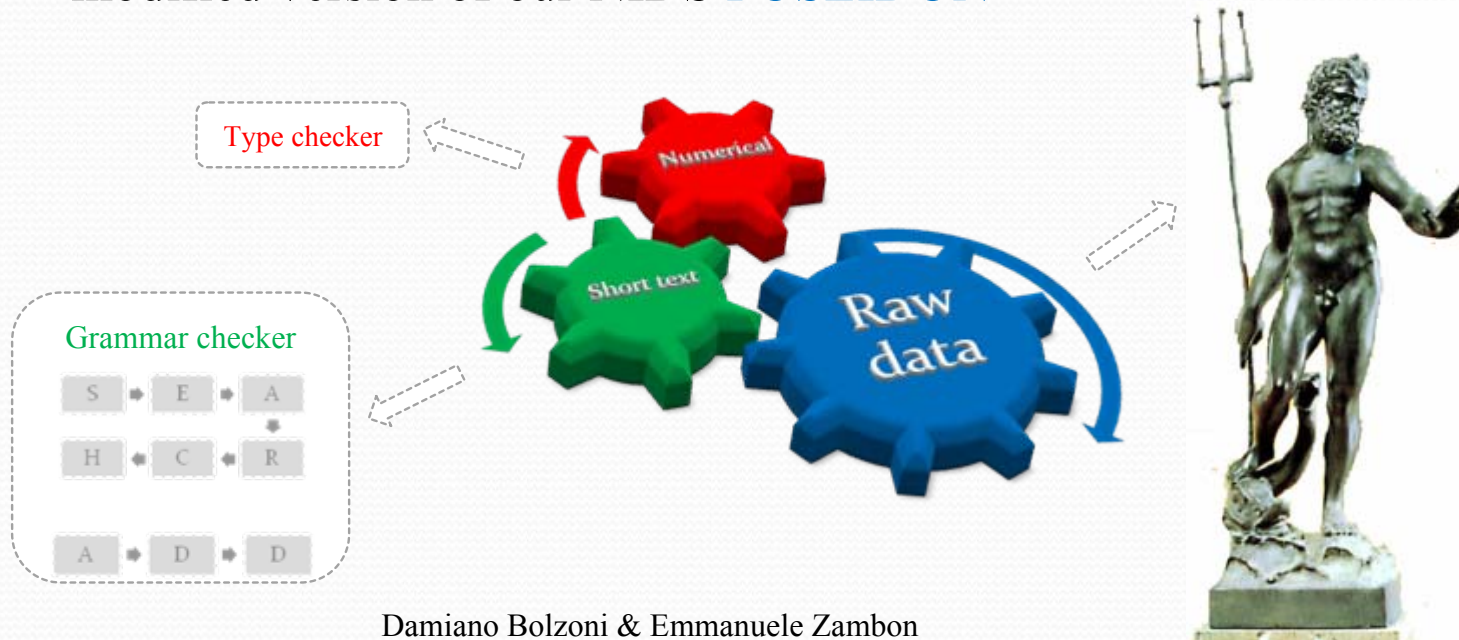
text-based
parameter

numerical-
based
parameter

- The content analysis of a certain parameter gives the best information to detect attacks
 - if we observe integer values and later some text, that is likely to be an attack (e.g. SQL Injection or XSS)

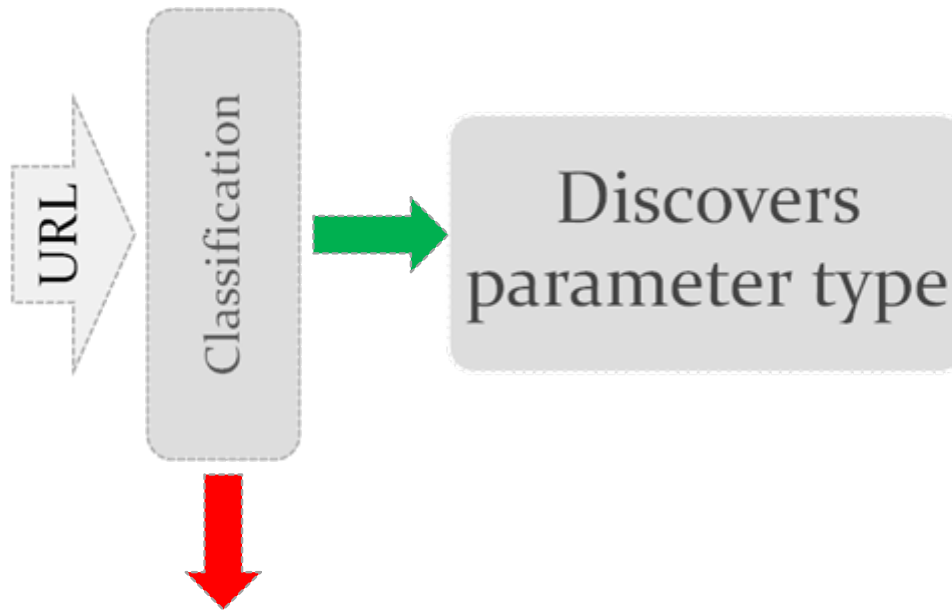
The detection engine

- The detection engine “learns” the parameters type
 - numerical (mainly integer, e.g. DBMS table entries) → **type checker**
 - short text (almost fixed length/content) → **grammar checker**
 - raw data(variable length/content, e.g. email messages/executables) → modified version of our NIDS **POSEIDON** [BZEH06]

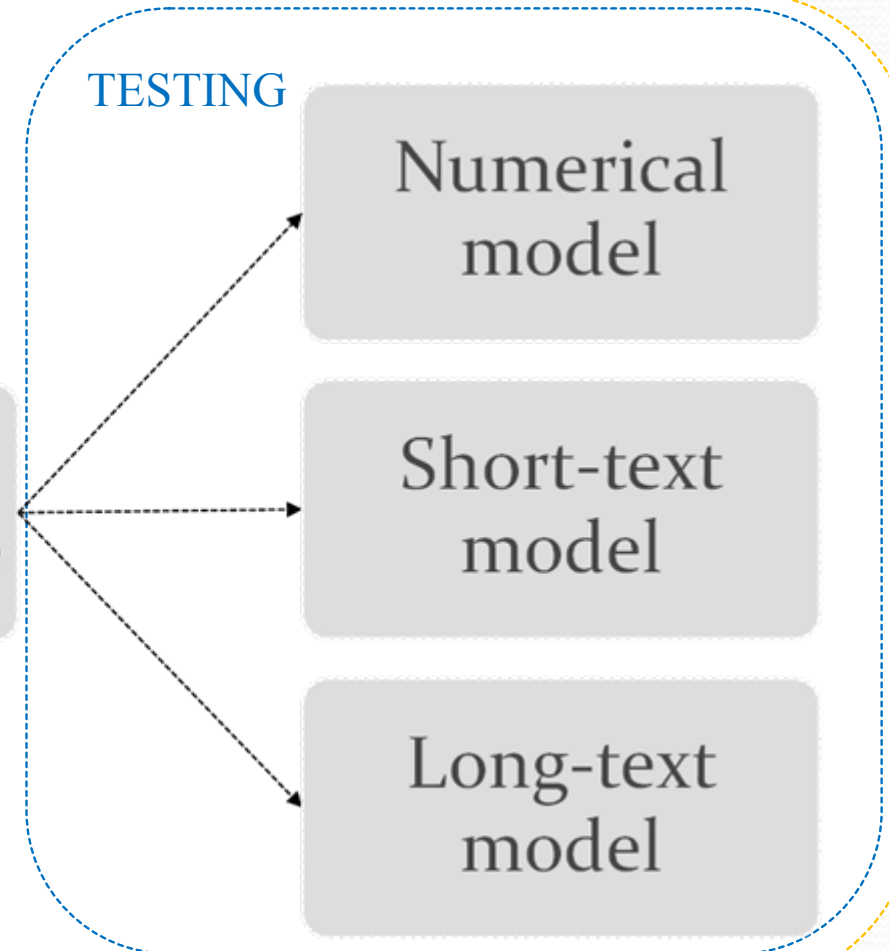


Data flow

TRAINING



TESTING



Numerical and short-text models

- Numerical model
 - discovers the numeric type (integer/decimal)
 - verifies that the following numbers are always of the same type
- Short-text model (fixed length or with slight variations)
 - an automaton is built to describe the character sequence
 - automata are merged in case of common sub-sequences
 - *my_short_text_1* and *my_short_text_2* → *my_short_text_(1|2)*
 - they can be generalized to reduce the size
 - e.g. session cookies: **abcd1234defg** → $a^* d^* a^*$ (but not @,+/,...)
 - if we have too many automata (e.g. > 10), or too “long”, for the same parameter, the detection becomes unfeasible → length information

`search=(%(5(E(a|u|n|t|v|s|p|r|w|k|o|g|b|m|l|y|h|i|f|q|x|j|z|c|e))s) | Go | plugin | external | None)`

Raw data

- Our NIDS POSEIDON uses the single network packets to detect attacks
 - a neural network pre-classifies the traffic
 - an anomaly engine based on n-gram analysis detect attacks
- This time we have a stream of characters...
 - we divide the stream in blocks of N characters
 - each block is then passed to the detection engine → n-gram analysis
 - when we detect X anomalies in a single stream, we consider it anomalous

Threshold setting

- One of the most common problem with anomaly-based systems is setting properly the threshold(s)
 - it influences directly detection and false positive rates
- The long-text model has some values to be set
 - length of n-grams to be analyzed (N)
 - number of anomalous n-grams (X)
- These values have been found experimentally
 - $N \rightarrow 5$
 - $X \rightarrow 5$

Benchmarks – DARPA 1999

- Unfortunately, we do not have large (and complex!) public data set for testing WAFs
 - we can compare current results with the previous ones of our NIDS POSEIDON

DARPA 1999		
	DR	FP
POSEIDON	100%	0.0016%
WIDS	100%	0%

Easy...let's for something serious now!



Benchmarks – Private data set

- We collected the traffic directed to the web server hosting the web sites of our department
 - ~1000 requests per hour
- We found evidences of SQL Injection and XSS attacks (manually crafted), few buffer overflows and a lot of scan attempts for well-known paths

Private data set		
	DR	FP
POSEIDON	100%	2.85%
WIDS	100%	0.2%

Generating signatures - 1

- Anomaly-based systems are more powerful than signature-based ones but the latter are easier to be deployed...when we have the right signatures...
 - if we are deploying an *ad hoc* web application, most probably we need to spend a lot of time on writing signatures...
- We do **NOT** generate signatures from detected attacks (although it could be possible): our signatures define what is **ALLOWED**
- Our system can generate in an automatic way signatures for *ModProxy* for numerical-based parameters and most of the short text-based ones
 - SecRule QUERY_STRING “!^(id=\d+)” → accepts only digits for *id*
 - SecRule QUERY_STRING “!^(func=(add|delete|view))” → accepts only certain values for *func*

Generating signatures - 2

DEMO

Conclusion

- Our WIDS is the first totally anomaly-based solution
 - infers the parameter types (we define 3 basic types)
 - uses a different detection model for each parameter type
 - generates signatures in an automatic way to define what is allowed to pass → ModSecurity deployment made easier
- Currently, we are going to deploy it in a data center of an important Dutch outsourcer, offering security services (IDSs, log analysis and forensics)
 - customers are mainly e-commerce sites and banks
 - we want to evaluate especially the on-line capabilities

Q&A



(be my Sphinx!)

References

- [BZEH06] D. Bolzoni, E. Zambon, S. Etalle and P. Hartel. **POSEIDON: a 2-tier Anomaly-based Network Intrusion Detection System**. In *IWIA '06: Proc. 4th IEEE International Workshop on Information Assurance*, 2006.
- [JKK06] N. Jovanovic, C. Kruegel, and E. Kirda. **Pixy: A Static Analysis Tool for Detecting Web Application Vulnerabilities**. In *S&P '06 : Proc. 27th IEEE Symposium on Security and Privacy*, 2006.
- [RVKA06] W. Robertson, G. Vigna, C. Krueger, and R. A. Kemmerer. **Using generalization and characterization techniques in the anomaly-based detection of web attacks**. In *NDSS '06: Proc. 17th ISOC Symposium on Network and Distributed Systems Security*, 2006.
- [VMV05] F. Valeur, D. Mutz, and G. Vigna. **A Learning-Based Approach to the Detection of SQL Attacks**. In *DIMVA '05: Proc. 2nd Conference on Detection of Intrusions and Malware and Vulnerability Assessment*, 2005.
- [ModSecurity] <http://www.modsecurity.org>