Institut
Mines-Télécom

Specification-Based Intrusion Detection using Sequence Alignment and Data Clustering

Djibrilla Amadou Kountché Sylvain Gombault

Outline

Web application Firewall

Definition

A WAF is an IDPS capable of :

- Analysing a traffic between a client and a server ;
- Validating further traffic given a set of rules (the model) ;
- Learn this model automatically.

Detection models

Based on three detection models [2] :

1. Knownledge-based ;
2. Behaviour-based ;
3. Hydrydation of the two other paradigms.

Specification-based Intrusion detection

Hypothesis

An intrusion can be detected by observing the deviation from the normal or accepted behaviour of the system or the user [4].

Types of specification

An specification can be define as [3] :

- Dictionaries ;
- Statistics, ontologies, etc. ;
- Finite automata or regular expressions ;

Automatic generation of regular expressions

Examples of algorithms

- Grammatical inference ;
- Genetic programming ;
- ReLIE, etc.

Sequence Alignment

- Observing patterns of conservation between sequences ;
- Finding commons patterns in the sequences ;
- Determining if the two sequences have not evolve from the same sequence.

Sequence Alignment

Definition : Pair alignment

Given [1] :

- $s = s_1 \dots s_n$ and $t = t_1 \dots t_m$ two strings defined on an alphabet Σ
- β the gap character, $\beta \ni \Sigma$
- $\Sigma' = \Sigma \cup \beta$
- $h : (\Sigma')^* \Longrightarrow \Sigma^*$ an homomorphism defined by $h(\alpha) = \alpha, \bigvee \alpha \in \Sigma$ and $h(\beta) = \lambda$. λ is the empty character.

Summarizing alignment

- Insertion : the gap characters in the first string ;
- Deletion : the gap character in the second string ;
- Match or correspondence : the 2 characters are identical
- Mismatch : the two characters are different from each other and gap

Score matrix

Dynamic programming

- Dynamic programming approach

Score matrix

- The first column and the first row are filled with $(j \times penalty)$ and $(i \times penalty)$
- Any other cell is filled with :

$$M(i,j) = \max \begin{cases} M(i-1,j) + penalty & \text{(i)} \\ M(i,j-1) + penalty & \text{(d)} \\ M(i-1,j-1) + d(s_i, t_j) & \text{(m)} \end{cases} \quad (1)$$

Sequence alignment

Definition : Score

$$\delta(s', t') = \sum_{i=1}^{l} d(s'_i, t'_i) \tag{2}$$

$$d(s'_i, t'_i) = \begin{cases} \text{match} & \text{if } s'_i, t'_i \in \Sigma \text{ and } s'_i = t'_j \\ \text{mismatch} & \text{if } s'_i, t'_j \in \Sigma \text{ and } s'_i \neq t'_j \\ \text{penalty} & \text{if } s'_i \text{ or } t'_j \text{ equals } \beta \end{cases} \tag{3}$$

Multiple sequence alignment

Multiple sequence alignment

Given n sequences $s_1, \ldots, s_n$ defined on an alphabet Σ having different lengths. β is the gap character and $h(\alpha) = \alpha$ an homomorphism.
A multiple alignment of $s_1, \ldots, s_n$ is an $n - uplet(s'_1, \ldots, s'_n)$ of length $l \geq |s_i|, i \in [1 \ldots n]$ on the alphabet Σ' where the following conditions hold :

1. $|s_i| = |s_j|, \bigvee i, j \in [1 \ldots n]$
2. $h(s'_i) = s_i, \bigvee, i \in [1 \ldots n]$
3. there is no row in the $(n \times l)$-matrix where there is only gap characters.

The proposed approach

Data acquisition

- Blackbox approach : no access to application code ;

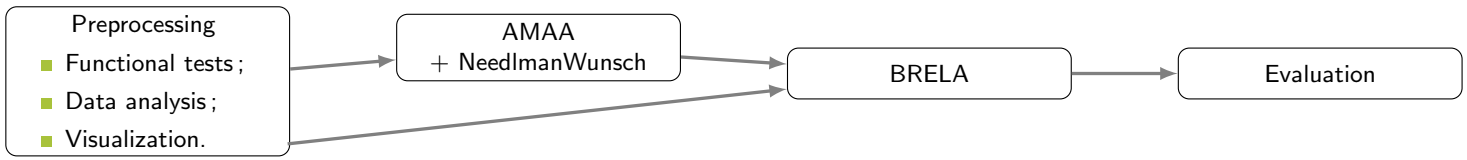

FIGURE: Description of the learning and evaluation process.

Another Multiple Alignment Algorithm (AMAA)

Parameters

- Pair alignment algorithm : Needleman Wunsch
- Gap character : $+$, $match = 1.0$, $mismatch = 0.0$, $penalty = -1.0$

Learning Data Set

```
GARFIELD THE LAST FAT CAT
GARFIELD THE FAST CAT
GARFIELD THE VERY FAST CAT
THE FAST CAT
```

TABLE: The GARFIELD dataset.

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. ```
   GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
   ```
   ```

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. `GARFIELD THE LAST FA+T CAT`
   `GARFIELD THE VERY FAST CAT`
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. ```
   GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
   ```
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
   ```

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. `GARFIELD THE LAST FA+T CAT`
   `GARFIELD THE VERY FAST CAT`
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
5. Realign 4 to 2. Choose the best :

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. ```
   GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
   ```
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
5. Realign 4 to 2. Choose the best :
6. ```
   GARFIELD THE+++++ FAST CAT
   GARFIELD THE LAST FA+T CAT
   ```

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. `GARFIELD THE LAST FA+T CAT`
   `GARFIELD THE VERY FAST CAT`
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
5. Realign 4 to 2. Choose the best :
6. `GARFIELD THE+++++ FAST CAT`
   `GARFIELD THE LAST FA+T CAT`
7. Suppress GARFIELD THE FAST CAT from 4 and add : GARFIELD THE+++++ FAST CAT :

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
5. Realign 4 to 2. Choose the best :
6. GARFIELD THE+++++ FAST CAT
   GARFIELD THE LAST FA+T CAT
7. Suppress GARFIELD THE FAST CAT from 4 and add : GARFIELD THE+++++ FAST CAT :
8. GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
   GARFIELD THE+++++ FAST CAT

Another Multiple Alignment Algorithm (AMAA)

1. AMAA searches the best alignment among all the input sequences :
2. GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
3. Suppress GARFIELD THE LAST FAT CAT, GARFIELD THE VERY FAST CAT :
4. Modified input set : GARFIELD THE FAST CAT, THE FAST CAT
5. Realign 4 to 2. Choose the best :
6. GARFIELD THE+++++ FAST CAT
   GARFIELD THE LAST FA+T CAT
7. Suppress GARFIELD THE FAST CAT from 4 and add : GARFIELD THE+++++ FAST CAT :
8. GARFIELD THE LAST FA+T CAT
   GARFIELD THE VERY FAST CAT
   GARFIELD THE+++++ FAST CAT
9. Repeat the process for THE FAST CAT.

BRELA

Learning the regular expression

- Alignment does not generate regular expression ;
- Sequences of same size ;
- Learn the corresponding regular expression.

Result of AMAA																								
G	A	R	F	I	E	L	D		T	H	E		V	E	R	Y		F	A	S	T		C	A	T
+	+	+	+	+	+	+	+	+	T	H	E	+	+	+	+	+		F	A	S	T		C	A	T
G	A	R	F	I	E	L	D		T	H	E	+	+	+	+	+		F	A	S	T		C	A	T
G	A	R	F	I	E	L	D		T	H	E		L	A	S	T		F	A	+	T		C	A	T
[GARFIELD]{0,1}THE[LAST]{0,1} FA[S]{0,1}T CAT																								

TABLE: The final result of AMAA on the GARFIELD dataset.

Evaluation : Learning phase

1. Generate data from `generatedata.com` ;
2. Apply BRELA and AMAA/BRELA on data of same size ;
3. Evaluate results of AMAA/BRELA using : true positive rate, false positive rate, the accuracy and the F-Measure ;
4. Use data clustering.

Comparing BRELA and AMAA/BRELA

Configuration

- Random parameter generation ;

Regex	Parameters (*match*, *mismatch*, *penalty*)
Date	
`[/0-9]{10}`	`9.65,-0.15,-92.46`
Currency	
`\$[5-9]{1},[0-9]{3}`	`66.21,0.0,-87.29`
`[$,0-9]{6}`	`2.06,-0.0015,-73.60`
GUID	
`[-0-9A-F]{24}`	`52.80,-0.008,-98.55`
`[0-9A-F]{8}-[0-9A-F]{4}-[0-9A-F]{4}-[0-9A-F]{4}`	`28.51,0.0,-90.87`
Phone	
`[0-9]{14}`	`83.54,-0.31,-12.71`
`0[1-9]{1}\[0-9]{2}\[0-9]{2}\[0-9]{2}[0-9]{2}`	`79.79,0.0,-92.36`
Password	
`[!"#$%&'()*+,-./0-9:;\<=\>\?@A-Z[\]^_‘a-z{|}~]{8}`	`21.60,-0.35,-39.41`

TABLE: This table summarised the results obtained for 100 runs. Up to two kinds of regular expressions are generated.

Evaluation

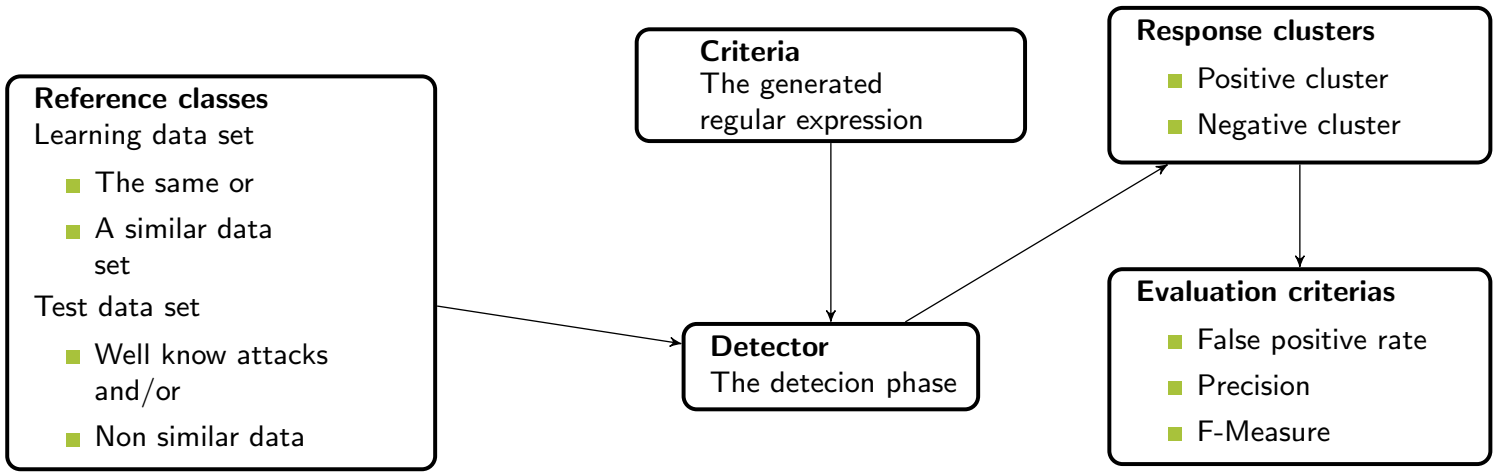

FIGURE: The detection phase.

Case 1 : The negative test dataset

Configuration

- Leanring datasets : password, date, phone and currency
- Test dataset : XSS or SQL injections

Results

Results for the first case				
	Date	Phone	Currency	Password
F-Measure	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)

TABLE: Results where the negative test dataset is composed only of XSS or SQL attacks.

Case 2 : The test dataset contains confusing data

Configuration

- Confusing data ;
- Other evaluation criteria.

Results

Results for the second case				
	Date	Phone	Currency	Password
TPR	0.63	0.77	1.0	0.52
FPR	0.36	0.23	0.0	0.47
Accurary	0.58	0.79	1.0	0.53
F-Measure	0.62	0.81	1.0	0.67

TABLE: Results where the test dataset is contains confusing data and XSS or SQL injections.

Clustering

Why clustering

- Cases where, at the learning phase, different types are present ;
- Determining the appropriate partitioning of the data ;
- Generate the corresponding regular expression can improve the performances.

Configuration

1. 16 clusters ;
2. For each clusters a regular expression using AMAA/BRELA ;
3. Evaluate for 4 datasets. .

Clustering

Results

N	Phone				Password				Date				Curr.			
	FM	AC	TP	FP	FM	AC	TP	FP	FM	AC	TP	FP	FM	AC	TP	FP
	0.66	0.52	0.60	0.40	0.65	0.52	0.60	0.40	0.70	0.58	0.64	0.36	0.76	0.67	0.69	0.31
	0.66	0.61	0.74	0.26	0.57	0.51	0.64	0.36	0.64	0.60	0.72	0.28	0.56	0.50	0.62	0.38
	0.76	**0.61**	**0.63**	**0.37**	**0.78**	**0.65**	**0.64**	**0.36**	**0.75**	**0.61**	**0.62**	**0.38**	**0.75**	**0.60**	**0.62**	**0.38**
	0.76	0.62	0.63	0.37	0.77	0.64	0.63	0.37	0.76	0.62	0.62	0.38	0.76	0.61	0.62	0.38
	0.60	0.50	0.61	0.39	0.61	0.51	0.61	0.39	0.86	0.82	0.86	0.14	0.60	0.50	0.60	0.40
	0.76	0.62	0.63	0.37	0.77	0.64	0.64	0.36	0.76	0.61	0.62	0.38	0.75	0.61	0.62	0.38
	0.59	0.50	0.61	0.39	0.59	0.50	0.60	0.40	**0.95**	**0.94**	**0.97**	**0.03**	0.59	0.50	0.60	0.40
	0.75	0.60	0.62	0.38	0.77	0.64	0.64	0.36	0.75	0.62	0.63	0.37	0.74	0.59	0.61	0.39
16	**0.85**	**0.82**	**0.91**	**0.09**	0.59	0.52	0.63	0.37	0.59	0.52	0.63	0.37	0.59	0.52	0.63	0.37
	0.75	0.61	0.62	0.38	0.78	0.65	0.65	0.35	0.75	0.60	0.62	0.38	0.74	0.60	0.61	0.39
	0.70	0.56	0.62	0.38	0.70	0.55	0.61	0.39	0.70	0.56	0.61	0.39	0.79	0.69	0.68	0.32
	0.69	0.55	0.61	0.39	0.69	0.55	0.60	0.40	0.69	0.55	0.60	0.40	**0.82**	**0.74**	**0.72**	**0.28**
	0.69	0.56	0.62	0.38	0.67	0.53	0.60	0.40	0.80	0.72	0.72	0.28	0.67	0.53	0.60	0.40
	0.75	0.60	0.62	0.38	0.78	0.66	0.65	0.35	0.74	0.60	0.62	0.38	0.74	0.59	0.61	0.39
	0.59	0.50	0.61	0.39	0.59	0.50	0.60	0.40	**0.95**	**0.94**	**0.97**	**0.03**	0.59	0.50	0.60	0.40

TABLE: 16 clusters, FM (F-Measure), AC (Accuracy), TP (True positive rate), FP(false positive rate).

Conclusions and Perspectives

- AMAA with other sequence alignment algorithm ;
- Comparison of AMAA/BRELA with other regular expression learning algorithms ;
- An evaluation of the clustering using statistical tests and partition evaluation criteria ;
- Seeking collaborations on the project.

Questions

Thank you for your attention.

This work is financed by the DGA-MI as a RAPID Project and is a cooperation with Kereval (SME).

Bibliography


H.J. Böckenhauer and D. Bongartz.
Algorithmic Aspects of Bioinformatics.
Natural Computing Series. Springer, 2007.


Hervé Debar, Marc Dacier, and Andreas Wespi.
Towards a taxonomy of intrusion-detection systems.
Computer Networks, 31(8) :805–822, 1999.


Pedro Garcia-Teodoro, J Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez.
Anomaly-based network intrusion detection : Techniques, systems and challenges.
computers & security, 28(1) :18–28, 2009.


Prem Uppuluri and R Sekar.
Experiences with specification-based intrusion detection.
In *Recent Advances in Intrusion Detection*, pages 172–189. Springer, 2001.