Specification-based intrusion detection using Sequence alignment and Data clustering

Djibrilla Amadou Kountché and Sylvain Gombault

Institut Mines-Télécom ; Télécom Bretagne ; IRISA/D2/OCIF RSM, Université européenne de Bretagne, France
{djibrilla.amadoukountche, sylvain.gombault}@telecom-bretagne.eu

Abstract. In this paper, we present our work on specification-based intrusion detection. Our goal is to build a web application firewall which is able to learn the normal behaviour of an application (and/or the user) from the traffic between a client and a server. The model learnt is used to validate future traffic. We will discuss later in this paper, the interactions between the learning phase and the exploitation phase of the generated model expressed as a set of regular expressions. These regular expressions are generated after a process of sequence alignment combined to BRELA (Basic Regular Expression Learning Algorithm) or directly by the later. We also present our multiple sequence alignment algorithm called AMAA (Another multiple Alignment Algorithm) and the usage of data clustering to improve the generated regular expressions. The detection phase is simulated in this paper by generating data which represent a traffic and using a pattern matcher to validate them.

Keywords: *positive security; sequence alignment; data clustering; web application firewall; specification-based ids*

1 Introduction

An *intrusion detection is the process of monitoring the events that occur in a computer system or network and analysing them for signs of possible incidents* [16]. Two major paradigms of intrusion detection methods can be defined based on the way the system determines that an attack took place or is being carried [5] : knowledge-based and behaviour-based. However, another classification of intrusion detection paradigms is given in [7,8,19] : misuse, anomaly and specification. Knowledge-based systems are the most common in computer security and correspond to misuse-based. These systems achieve high performance in term of false positive rate when known attacks are being carried on. But, they require frequent updates of the knowledge base and have difficulties to determine the variant of the same attack. Mostly, the algorithms used by these systems are based on string matching like Aho-Corasick. Many works have been done for automatic learning of the signature based as well on data mining [1] as on sequence alignment [11,13]. Behaviour-based intrusion detection systems design the normal and/or the abnormal behaviour of the user and the system. Therefore, an

attack is defined as an abnormal use of a given application or a deviation from the normal behaviour. However, not all deviations from the model are attacks, thus, this raises the rate of false positive. Behavioural intrusion detection made possible the detection of unknown attacks but has the following drawbacks [5] : i) a high level of false positive due to difficulty to cover the whole behaviour of the target (system or user) at the learning phase; ii) when the behaviour changed, an update of the model is required and iii) some attacks can be introduced in the reference model and considered as normal. Behaviour-based systems are getting more popular in commercial intrusion detection and protection systems (IDPS) because of the potential of positive security. Specification and anomaly-based intrusion detection are part of behaviour paradigm and specification-based is defined in deep in section 2.

These two approaches (knowledge and behaviour) can be combined to perform better results [15]. Indeed, depending on the modelling of the problem, most of the machine learning algorithms can be used for both purposes. In this project, we plan to use the mixed approach and the user of our WAF will be given the possibility to combine many algorithms at the learning and the exploitation phase.

This paper is organized as follow : the next section describes some related works and section 3 develops sequence alignment. The section 4 presents our contributions which are evaluated in the section 5. Finally, the last section concludes this paper and gives the perspectives on our work.

2 Related works

Specification-based intrusion detection made the hypothesis that an intrusion can be detected by observing the deviation from the normal or accepted behaviour [1] of the system or the user. The model can be expressed as : i) dictionaries; ii) finite automata or regular expressions ; iii) statistics, ontologies, etc. Uppuluri and Sekar [19] defined a domain specific language called BMSL (Behaviour modelling specification language) in which a specification can be defined as :

$$pat \rightarrow action \tag{1}$$

Where *pat* is a set of regular expression representing the *history* of normal behaviour observed and *action* corresponds to the services accessed after this normal behaviour. Other DSL have been defined for negative security [20].

A specification-based IDPS comprises two main steps :

1. the learning phase : determination of the model from the data gathered (incomplete and noisy) ;
2. and the exploitation phase : comparison of the observed behaviour to the model.

[1] In the rest of the paper, the term behaviour and specification are considered the same.

The step before the learning phase has a great importance. Indeed, the quality of the data from which the specification will be drawn conditioned the results of the learning phase. In web applications, many factors made the collection of data and their accuracy difficult to guarantee. For example, the rapid evolution of frameworks, the coexistence of different version of protocols, applications and the fact that input validation is not always done. However, we consider functional tests as important data source for the learning phase [12]. Also, the specification can be done by "hand" by security experts. The next section reviews some algorithms used for the learning and exploitation phases and presents in depth the regular expression learning problem.

2.1 Algorithms used in intrusion detection

Statistical methods are the most intuitive for characterizing the behaviour which is describe by variables :

$$B = x_i, i \in [1, .., n] \tag{2}$$

Where x_i is a characteristic of the system. The variables may represent a mean or any other central tendency measure or a complex probability distribution like Gaussian Mixture Model. Therefore, methods like statistical tests or expectation maximisation can be used to estimate the parameters of the model [9, 21]. Expert systems and Finite state machines have also been used to model the behaviour of the target [5]. In this project, we focus on using data mining algorithms at all the levels from the data preprocessing to the exploitation phase. Many algorithms are described in [1] for using data analysis methods for computer security. Also, we distinguish detection algorithms used for validation and algorithms used for generation of the model. This model can be deeply tied to the algorithms as in the statistical case but for regular expressions, two different algorithms can be used. In the next section, the problem of regular expression learning is presented followed by a study of sequence alignment.

2.2 Automatic learning of regular expressions

We consider learning a regular expression form positive data. This problem consists of learning a regular expression which corresponds the most to the given dataset and permit to generalize to data expressed in different forms. However, there is an infinite number of regular expression corresponding to the learning dataset. C. De La Higuera gives a detailed description of other related problems [4].

H. Fernau [6] proposed some algorithms for generating a regular expression based on suffix tree and finite automata. These algorithms first build the suffix tree and determine a corresponding regular expression from the automata.

ReLIE was proposed by Yunyao et *al* [10] for automatic generation of regular expressions from text documents in the goal of information retrieval. ReLIE took as input a regular expression R_0 and a document D. It transforms iteratively R_0

in a set of regular expressions and choose the one which maximise the objective function (the *F-Measure*).

A. Bartoli et *al* [2] proposed a genetic algorithm which took as input a couple (s,t) where : s is a string and t the substring for which to construct the the regular expression. The initial population is a set of individual corresponding to valid regular expressions. Every individual is represented as a tree and the fitness function is the sum of Leveinstein distances between the string and other strings which correspond to it toward the regular expression and the length of the regular expression.

Y. Tang et *al* proposed methods based sequence alignment for polymorphic worms signature generation. They proposed: i) a pair alignment algorithm (Contiguous Substring Reward (CSR)) based on Needleman-Wunsch by modifying the score function to favour the alignment of contiguous sequences; ii) and a simplified regular expression (SRE) generation method based on a lookup table [17]. They also proposed the usage of multiple sequence alignment [18] in order to improve the SREs.

In the following, we proposed two algorithms for generating positive rules as basic regular expression which are more strict than the SRE. Also, our algorithm AMAA can take CSR as parameter.

3 Sequence alignment

Sequence alignment determines between many strings if invariant characters are present at the same position in all the strings by insertion of a gap character. It is originated from bioinformatics [3] where it is used for :

- observing patterns of conservation (or variability) between sequences (modelled as a string) ;
- finding commons patterns in the sequences ;
- determining if the two sequences have not evolve from the same sequence.

Definition 1. *Given :*

- $s = s_1 \dots s_n$ *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.*

An alignment of s *and* t *is a pair* (s', t') *of strings of length* $l \geq \max|s|, |t|$ *define on* Σ' *such that the following conditions hold :*

1. $|s'| = |t'|$
2. $h(|s'|) = s$ *and* $h(|t'|) = t$
3. *there can not be at the same position in* s' *and* t' *only gap characters.*

```
Telecom-Bretagne
Kere+++++++val++
```

Fig. 1. Example of an alignment of Kereval and Telecom-Bretagne.

The second condition means that by deleting all the gap, we find back the strings s and t. An example is given by the table 1 for the strings "Telecom-Bretagne" and "Kereval" where, the '+' character is used for the gap.

For an alignment of a pair of sequences, there exist four cases :

- insertion : the gap characters is inserted in the first string;
- deletion : the gap character is inserted in the second string;
- match or correspondence: the 2 characters are identical in the same column;
- mismatch : the two characters are different from each other and different from the gap character.

Definitions

The score of an alignment (s', t') of length l is defined as follow :

$$\delta(s', t') = \sum_{i=1}^{l} d(s'_i, t'_i) \tag{3}$$

$$d(s'_i, t'_i) = \begin{cases} \text{match} & \text{if } s'_i, t'_i \in \Sigma \text{ and } s'_i = t'_j \\ \text{mismatch} & \text{if } s'_i, t'_j \in \Sigma \text{ and } s'_i \neq t'_j \\ \text{penalty} & \text{if } s'_i \text{ or } t'_j \text{ equals } \beta \end{cases} \tag{4}$$

Where $d(.,.)$ is a function defined on Σ and determine the similarity or the distance of the two characters. The value of *match*, *mismatch* and *penalty* are chosen in accordance to the application domain. Based on this score function, the alignment problem can be seen as an optimisation problem where the goal is to maximize the score function [3].

The score matrix indicates the score between the two sub-strings of s and t at a certain time of the progression of the alignment. This is the basis of the approach of dynamic programming of sequence alignment : determination of partial optimal solutions for the problem. The score matrix of the pair (s, t) having length n and m is a $(n+1) \times (m+1)$ matrix where :

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

$$M(i,j) = \max \begin{cases} M(i-1, j) + penalty & \text{(i)} \\ M(i, j-1) + penalty & \text{(d)} \\ M(i-1, j-1) + d(s_i, t_j) & \text{(m)} \end{cases} \tag{5}$$

The tracing of this matrix gives the results of the alignment (s', t'). Depending on the way the matrix is initialized and backward traced, one can distinguish :

- **Global alignment** : the initialization is as described earlier and the tracing starts at the $(n+1, m+1)$ cell;
- **Local alignment** : the function used to fill the cells became :

$$M(i,j) = \max(\max(i,d,m),0) \tag{6}$$

  And the tracing starts at the cell having the maximum value in the whole matrix.
- the **Semi global alignment** case is describe in detail in [3].

The gap penalty is the parameter which favour the insertion or the deletion. A high value of the gap tends to prevent the gap character to be inserted in both of the sequences. This parameters is mostly chosen in bio-informatics based on an affine function.

3.1 Multiple sequence alignment

Definition 2. *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.*

Multiple sequence alignment problem is NP-Complete, therefore many algorithms have been proposed ranging from exact computation of the best alignment, determination of the consensual alignment to a combination with a pair alignment method [3]. Most of the algorithms were proposed for the purpose of bioinformatics and implementations are available online as servers. In this project, we started an adaptation of T-Coffee [14] for our purpose.

4 The proposed approach

The figure 2 shows the approach used for the learning process. The preprocessing phase consists of cleaning the redundant and noisy data and clustering them before the learning phase. Also, the user can visualise them in order to choose the partitioning appropriate for the learning phase. After the preprocessing phase, when the data are of the same size, BRELA can be directly applied on them or applied after a multiple sequence alignment by AMAA. Then, the results are evaluated. Also, it can be possible, after the evaluation to include certain data in the learning dataset and redo the process. AMAA and BRELA are described in the next section.

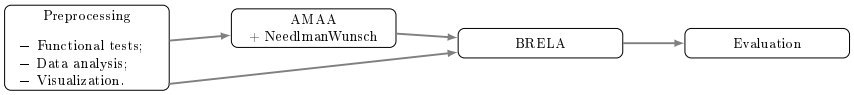

Fig. 2. Description of the learning and evaluation process.

4.1 AMAA : Another multiple alignment algorithm

To illustrate how AMAA works, the GARFIELD dataset [14], described in table 1, is used. The parameters of AMAA are the alignment algorithm and it's parameters. The configuration used is : '+' as gap character, $match = 1.0$, $mismatch = 0.0$ and the $penalty = -1.0$.

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

Table 1. The GARFIELD dataset.

1. AMAA searches the best alignment among all the sequences which is shown in the first column of Table 2.
2. the set $Q = \{$"GARFIELD THE LAST FAT CAT", "GARFIELD THE VERY FAST CAT"$\}$ is suppressed form the input which now contains (GARFIELD THE FAST CAT, THE FAST CAT).
3. The content of the input set is aligned to the result of 1. The best alignments for these sequences are given in the second and the third columns of Table 2.
4. In this example, AMAA did nod pad the sequences, as they are very similar.

The final result is displayed in table 3.

$i = 0$	$i = 1$	$i = 2$
GARFIELD THE LAST FA+T CAT	GARFIELD THE+++++ FAST CAT	+++++++++THE+++++ FAST CAT
GARFIELD THE VERY FAST CAT	GARFIELD THE LAST FA+T CAT	GARFIELD THE VERY FAST CAT

Table 2. Description of the steps of AMAA.

4.2 BRELA : Basic regular expression learning algorithm

BRELA will consider an array $n \times m$ like the result of AMAA in Table 3. It determines the character frequencies for each column generate the regular expression given in the last line of Table 3 as follow : in the first column, the character 'G' and '+' are encountered, so G can be considered as a fixed field. The

Algorithm 1 AMAA : Another Multiple Alignment Algorithm

Input : $S = \{s_i, i = 1, \ldots, n\}$ set of sequences to be aligned.
$S' = \emptyset$
And a pair alignment algorithm (Needleman-Wunsch [3])
Output : $S' = \{a(s_i), i = 1, \ldots, n\}$ the aligned sequences (with the same length). Where $a(s_i)$ is the alignment of s_i produce by the algorithm.
Initialization :
Determine $n \times (n-1)$ alignments of all sequences and choose the best alignment :
$A^* = (s'_k, s'_p)$ such that $score(s'_k, s'_p) > score(s'_i, s'_j)$
Insert the best pair (A^*) of sequences in S' and delete (s_k, s_p) form S
while S is not empty **do**
    $s \leftarrow pop(S)$
    Align s with all the sequences in S' and choose the best sequences
    for All the sequences $s' \in S'$ **do**
        if s' has been modified by the alignment **then**
            Replace s' in S' by the new sequences s''
        else if the length of the a sequence changed **then**
            Insert padding in the position where an insertion or a deletion took place
        end if
    end for
    Add the new best alignment for the sequence s in S'
end while

	Result of AMAA
	GARFIELD THE VERY FAST CAT
	+++++++++THE+++++ FAST CAT
	GARFIELD THE+++++ FAST CAT
	GARFIELD THE LAST FA+T CAT
Result of BRELA	[GARFIELD]{0,1}THE[LAST]{0,1} FA[S]{0,1}T CAT

Table 3. The final result of AMAA on the GARFIELD dataset.

Data sets					
	Date	Phone	Curr.	Password	GUID
Samples	15/12/2024	01 97 74 34 19	$8,803	e?DEm8Wo	6E6BA93F-5960-BA96-4572
	23/12/2016	05 27 24 82 61	$6,794	.ZyE-FKv	BF54713E-E402-7515-763D
	13/11/2003	04 96 03 64 98	$9,067	@'@H"z'[	20E2E738-0AF1-F7B7-3EF1
	01/09/2230	05 99 44 53 73	$7,030	<2iEo3dN	47016986-1837-55F6-C019
Size	89	100	100	89	100

Table 4. Examples of datasets used for the evaluation.

Algorithm 2 Basic Regular Expression Learning Algorithm

Input : a set $I = \{s_i, i = 1, \ldots, n\}$ with $len(s_i) = len(s_j) = m, i \neq j, 1 <= i, j <= n$
Output : The corresponding regular expression RE
for every column $c_k, 0 <= k < m$ **do**
  Determine the characters distribution for c_k
  if c_k contains only one type of character α **then**
    α is an invariant character and append α in RE
  else
    while c_k is not a fixed character **do**
      Create a set (V) and insert the characters of c_k in it
      Determine the min and max length (stop at the next fixed character)
      $k \leftarrow k + 1$
    end while
    Sort the characters of V and append the character to RE as $[ordered(V)]\{\min, \max\}$
  end if
end for

same holds until the character 'D'. So the sequence of characters [GARFIELD] is inserted in the regular expression followed by an interval indicating that this word was observed as is 0 time or once. The process is continued until the last column. The words 'THE' and 'CAT', for example, have been identified as invariant fields

5 Experimentations

The flow of evaluations is describe as follows :

1. we generate data from `generatedata.com`;
2. BRELA is applied on data with the same size then is combined with AMAA;
3. in this scenario, the result of AMAA/BRELA are evaluated using the : true positive rate, false positive rate, the accuracy and the F-Measure [2].
4. for this case, the datasets are mixed and data clustering is used to discriminate them and a regular expression is generated for each cluster.

5.1 Comparison of BRELA and AMAA/BRELA

For this case, the experimentations are conducted as follow :

1. for every dataset, a regular expression is generated with BRELA;
2. first, the parameters of AMAA had constant values : $(match = 1.0, mismatch = 0.0, penalty = -1.0)$
3. then, uniform distributions are applied to generate values for the parameters.

[2] These criteria are explained on `http://en.wikipedia.org/wiki/F1_score`

The algorithms are implemented in Java and the experimentations where run on a Intel $core^{TM}i5$ computer with $4GB$ of RAM. The colt [3] implementation of the MersenneTwister was exploited for the uniform distributions. The experimentations where run 100 times for each dataset.

Data set	Regex
Dates	`[0-9]{2}/[0-9]{2}/2[0-9]{3}`
Phone	`0[1-9]{1} [0-9]{2} [0-9]{2} [0-9]{2} [0-9]{2}`
Currency	`$[56789]{1},[0123456789]{3}`
Password	`[!"#$%&'()*+,-./[0-9]:;<=>?@[A-Z][]^_‘[a-z]{|}~]{8}`
GUID	`[0-9A-F]{8}-[0-9A-F]{4}-[0-9A-F]{4}-[0-9A-F]{4}`

Table 5. Results obtained for both BRELA and the combination of AMAA and BRELA. In the second case, the parameters are $match = 1.0, mismatch = 0.0, penalty = -1.0$

The random seed generator provided in colt was used for each parameters. The configuration of the generators is described in the table 6.

Parameter	Interval	Seed
$match$	$[1.0, 100.0]$	9876
$mismatch$	$[-1.0, 0.0]$	1299961164
$penalty$	$[-100.0, -1.0]$	669708517

Table 6. The configuration of the uniform random 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 7. This table summarised the results obtained for 100 runs. Up to two kinds of regular expressions are generated.

[3] https://dst.lbl.gov/ACSSoftware/colt/

Discussions The results in Table 5 show that BRELA and AMAA/BRELA produce the same regular expression. These regular expressions preserved the invariant fields (which are identifiable without alignment). But in the case of the password dataset, any invariant field is present. Therefore, BRELA produced a *pattern* which is an ordering of the characters followed by the total length of the strings. In the third scenario, the results are illustrated in Table 7. The regular expression generated depends on the parameters of the alignment algorithm. While a high value of match allows the alignment of the same characters in the same column, a high value of mismatch will change the natural alignment of the characters. When the data are similar and the invariant are distinguishable (even if they are different), BRELA is faster for determining the regular expression. Aligning the sequences before applying BRELA can produce the same result or will help to identify some field which are hidden in the dataset. We will study this case in the next section with datasets composed by different string of different lengths.

5.2 Detection performances analysis

The process of this experimentation is describe by the figure 3.

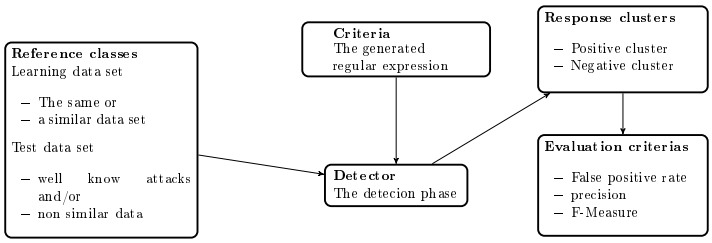

Fig. 3. The evaluation process.

The dataset represents a *specification* of the system captured at a certain time. These data are employed to generate a regular expression which in turn is use to evaluate a mixed dataset containing the same and/or similar data and know XSS and SQL injections. After the evaluation, two classes are generated : the positive cluster which contains the data considered as normal and the negative cluster contains the data considered as not conform to the specification.

Case 1 : The negative test dataset for password, date, phone and currency. contains only XSS or SQL injections while the positive contains corresponding data. The results are summarized in the Table 9.

Discussion When the positive and negative datasets are significantly different, all the data in the negative class will be considered as attacks. Even for the

Regular expression learning datasets				
	Date	Phone	Currency	Password
samples	29/11/2104	01 97 74 34 19	$8,803	=o9_(om
	06.30.22	(+33)0408017767	£12.63	}x5"C
	10-14-55	0298623840	107,774€	@'@H"z'[
	10/22/43	+332 40 77 54 65	€0,266	%|Y}:A&5eAb
Total size	103	99	57	74

Table 8. Examples of datasets used for the evaluation.

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

passwords dataset, which contains characters used for XSS and SQL injections. In this case, the length of the strings are inferior to the length of an XSS or SQL injections attacks.

Case 2 : The test dataset contains confusing data The lengths, certain fields of the data are varied so that the negative dataset will contain data which are close to learning dataset but must be classified as attacks. Other criteria for the evaluation will be used : the accuracy, the true positive rate, the false negative rate.

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

Discussion In Table 10, the currency dataset has the best values for all the criteria. The regular expression discriminates well-formed currencies expressed as prefixed Dollar, Euro and Pounds and others from others numeric values. However, in the case of the password, some XSS and SQL attacks were considered as normal. As stated before, this is due to the presence of some ambiguous characters. Also the specification is defined strictly so that the date is considered different than password. But, in real use cases, some users consider a date or

phone numbers as a password. For the date dataset, the format use for the generation was $day : sep : month : sep : year$. Where : sep : is a separator. Therefore, the validator has to reject other date format until the user decides to use them in future. However, permutations of the field like year and day, are accepted. In the case of the phone dataset, some plain number are also accepted as normal phone number. However, for all these datasets, the validator continues to consider the injections dataset as abnormal.

5.3 The data clustering

Clustering is used for the cases where, at the learning phase, different types are present. Thus, we believed that determining the appropriate partitioning of the data and generate the corresponding regular expression can improve the performances. At the exploitation phase, the validator will also cluster the data or use a hierarchy of the regular expressions to determine for a data which regular expression is most appropriate for validating it. However, in this paper, only the clustering at the learning phase is studied. The data from each of the four datasets are mixed and each corresponds to 25%. The total length of the clustering dataset is 200. We use the LingPipe Java Library [4].

Case 1 : One regular expression the dataset We generate one regular expression without clustering. The result is given in the Figure 4. This regu-

```
[ !"#$%&'()*+,-./0-9:;<=>?@A-Z[\]^_‘a-z{|}~£€]{5,103}
```

Fig. 4. Regular expression generated with default values for the alignment algorithm.

lar expression will consider validate any sequence of characters having a length between 5 and 103. Therefore, it is not accurate to discriminate inappropriate inputs. In the table 11, we show the result for the evaluation criteria by considering only a dataset where the positive data are the mixed from the four datasets and the negative dataset consists of XSS and SQL attacks.

One regular expression	
TPR	0.86
FPR	0.14
Accurary	0.86
F-Measure	0.92

Table 11. Results where the test dataset is composed only of XSS or SQL attacks.

[4] http://alias-i.com/lingpipe/index.html

In this case, the accuracy, F-measure and true positive rate are higher and corresponds to the analysis on the regular expression.

Case 2 : Determining the appropriate number of regulars expressions In this experimentation, we present result with hierarchical clustering with Jaro and Winkler distance although we tried K-Means. We also tried edit distance, but in our case, it tends to produce a higher number of sparse clusters than Jaro Winkler distance. In order to evaluate the clustering :

1. we choose a partitioning of 16 clusters ;
2. for each clusters we generate a regular expression using AMAA/BRELA ;
3. then we determine the values of the criteria for the 4 datasets used to build the clustering dataset. Therefore, we consider the three others as negative.

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 12. A cut of the dendrogram to have a number of clusters $N = 16$. FM (F-Measure), AC (Accuracy), TP (True positive rate), FP(false positive rate).

Discussion For each of the dataset, the combination of the clustering and regular expression learning is able to association a cluster. The bold results in the Table 12 show which cluster can be associated to the dataset. These clusters have for example the highest F-measure and the lowest false positive rate. However, some clusters can also correspond to all the datasets at the same time (Bold and italic results). One particular case is the password dataset which summarises the case where it is difficult to determine an appropriate cluster. We show that the clustering phase can allow the user to determine which data to gather together as similar and generate a regular expression. Therefore, when he goes through the dendrogram, he can choose the number of partitions or decide to merge some of them. An important remark is regarding the number of clusters. We choose 16 to illustrate the potentialities of the clustering but there are many other possible partitioning in the dendrogram. Therefore, we will applied statistical criteria to determine the number of clusters and visualisation.

6 Conclusion

Specification-based intrusion detection is a promising model. In this paper, we presented our algorithms AMAA and BRELA for automatic learning of the specification and we combine them to data clustering. When the data are of the same length, BRELA can be directly use but it will not be able to determine hidden invariant fields. AMAA can determine these fields before the generation of the regular expression. In the case of web forms for example, many data types can be mixed. Thus, the clustering will help the user of our WAF to organise the data in clusters corresponding to the needs. Also, we are working to integrate other string clustering metrics and data clustering algorithms. Also a tree levels of comparison is under the way : i) AMAA with other sequence alignment algorithm; ii) comparison of AMAA/BRELA with other regular expression learning algorithms and iii) an evaluation of the clustering using statistical tests and partition evaluation criteria.

Acknowledgements

This work is a part of the RoCaWeb project carried at Kereval and Telecom-Bretagne and financed as a RAPID project by the DGA-MI. We would like to thank Alain Ribault, Constant Chartier, Frédéric Majorczyk and Yacine Tamoudi.

References

1. Niall Adams and Nicholas Heard. *Data Analysis for Network Cyber-Security*. World Scientific, 2014.
2. Alberto Bartoli, Giorgio Davanzo, Andrea De Lorenzo, Marco Mauri, Eric Medvet, and Enrico Sorio. Automatic generation of regular expressions from examples with genetic programming. In *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation*, pages 1477–1478. ACM, 2012.
3. H.J. Böckenhauer and D. Bongartz. *Algorithmic Aspects of Bioinformatics*. Natural Computing Series. Springer, 2007.
4. Colin De La Higuera. A bibliographical study of grammatical inference. *Pattern recognition*, 38(9):1332–1348, 2005.
5. Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, 1999.
6. Henning Fernau. Algorithms for learning regular expressions from positive data. *Information and Computation*, 207(4):521–541, 2009.
7. 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.
8. Paria Jokar, Hasen Nicanfar, and Victor CM Leung. Specification-based intrusion detection for home area networks in smart grids. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 208–213. IEEE, 2011.

9. Christopher Kruegel, Giovanni Vigna, and William Robertson. A multi-model approach to the detection of web-based attacks. *Computer Networks*, 48(5):717–738, 2005.
10. Yunyao Li, Rajasekar Krishnamurthy, Sriram Raghavan, Shivakumar Vaithyanathan, and HV Jagadish. Regular expression learning for information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 21–30. Association for Computational Linguistics, 2008.
11. Zhichun Li, Manan Sanghi, Yan Chen, Ming-Yang Kao, and Brian Chavez. Hamsa: Fast signature generation for zero-day polymorphic worms with provable attack resilience. In *Security and Privacy, 2006 IEEE Symposium on*, pages 15–pp. IEEE, 2006.
12. Tejeddine Mouelhi. *Testing and Modeling Security Mechanisms in Web Applications*. Theses, Institut National des Télécommunications, September 2010.
13. James Newsome, Brad Karp, and Dawn Song. Polygraph: Automatically generating signatures for polymorphic worms. In *Security and Privacy, 2005 IEEE Symposium on*, pages 226–241. IEEE, 2005.
14. Cédric Notredame, Desmond G Higgins, and Jaap Heringa. T-coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, 302(1):205–217, 2000.
15. Jerome H Saltzer and Michael D Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.
16. Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (idps). *NIST special publication*, 800(2007):94, 2007.
17. Yong Tang, Xicheng Lu, and Bin Xiao. Generating simplified regular expression signatures for polymorphic worms. In *Autonomic and Trusted Computing*, pages 478–488. Springer, 2007.
18. Yong Tang, Bin Xiao, and Xicheng Lu. Using a bioinformatics approach to generate accurate exploit-based signatures for polymorphic worms. *computers & security*, 28(8):827–842, 2009.
19. Prem Uppuluri and R Sekar. Experiences with specification-based intrusion detection. In *Recent Advances in Intrusion Detection*, pages 172–189. Springer, 2001.
20. Giovanni Vigna, Fredrik Valeur, and Richard A Kemmerer. Designing and implementing a family of intrusion detection systems. In *ACM SIGSOFT Software Engineering Notes*, volume 28, pages 88–97. ACM, 2003.
21. Nong Ye, Xiangyang Li, Qiang Chen, Syed Masum Emran, and Mingming Xu. Probabilistic techniques for intrusion detection based on computer audit data. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(4):266–274, 2001.