Journal of Computer Security 17 (2009) 945–968
DOI 10.3233/JCS-2009-0350
IOS Press

A cascade architecture for DoS attacks detection based on the wavelet transform

Alberto Dainotti *, Antonio Pescapé and Giorgio Ventre
University of Napoli "Federico II", Napoli, Italy
E-mails: {alberto,pescape,giorgio}@unina.it

In this paper we propose an automated system able to detect volume-based anomalies in network traffic caused by Denial of Service (DoS) attacks. We designed a system with a two-stage architecture that combines more traditional change point detection approaches (Adaptive Threshold and Cumulative Sum) with a novel one based on the Continuous Wavelet Transform. The presented anomaly detection system is able to achieve good results in terms of the trade-off between correct detections and false alarms, estimation of anomaly duration, and ability to distinguish between subsequent anomalies. We test our system using a set of publicly available *attack-free* traffic traces to which we superimpose anomaly profiles obtained both as time series of known common behaviors and by generating traffic with real tools for DoS attacks. Extensive test results show how the proposed system accurately detects a wide range of DoS anomalies and how the performance indicators are affected by anomalies characteristics (i.e. amplitude and duration). Moreover, we separately consider and evaluate some special test-cases.

Keywords: Anomaly, detection, attack, wavelet

1. Introduction

There is a relatively long list of network events that can be considered anomalous from the network operator point of view and that can be associated to variations of the network traffic profile. On one side, there are user-generated events, which can be malicious or not. In the first case different kinds of attacks are conducted by one or more machines and are directed to, or traverse, the network under operation. Some examples are Worms, Scans and Denial of Service (DoS) attacks. However, it is possible that even a legitimate behavior of one of more users, under some specific conditions, could generate an event requiring the attention of network managers. Flashcrowds are a notable example [27]. The posting on a heavily frequented news site of the URL of a not very popular web page is a typical case that may cause such a concentration of visitors in the same time period that the resulting amount of link traffic would lead to unexpected network problems. On the other side, there are anomalous events that are not generated by users, as outages, link failures and

*Corresponding author: Alberto Dainotti, University of Napoli "Federico II", Via Claudio 21, 80125 Napoli, Italy. Tel.: +39 081 7683821; Fax: +39 081 7683816; E-mail: alberto@unina.it.

network device misconfigurations. The efficient operation and management of current large networks depend heavily on the correct analysis of all these anomalies. However, their accurate detection and classification in IP networks is still an open issue due to the intrinsic complex nature of network traffic. Also because isolating anomalous events within traffic is an inherently difficult task. For these reasons, the design and evaluation of a variegate set of anomaly detection systems (ADS), based on very different approaches and techniques, are currently the subject of many research studies.

In particular, a lot of work has been concentrated on the detection of Denial of Service attacks. The motivation behind this, is that these kinds of attacks are very difficult to defend against, and they have caused (and still do) large economical losses. A Denial of Service attack is typically based on the consumption of victim's resources (CPU, bandwidth, memory) to deny legitimate users access to network services. Such attacks are often performed in a highly distributed fashion, making an early and automated detection and the adoption of countermeasures extremely difficult. Even the largest computer-industry companies and high-visibility Internet e-commerce sites have notoriously been victims of Denial of Service attacks since at least year 2000. However, it is known the vast majority of attacks are not publicized, while they include a wide range of victims: from commercial sites, to educational institutions and government organizations [24]. All these summing to considerable damage and economical losses.

In this paper we propose a novel approach – tested in an offline fashion – to the detection of anomalous network events focused on Denial of Service attacks, and centered on the use of the continuous wavelet transform. Network traffic is analyzed by looking at the packet rate signal, processed by a two-stage system that is able not only to raise an alert in case of a detected attack, but also to report to the operator an estimation of the time interval during which the anomaly was present. We evaluate several properties of the proposed system, using a very large set of synthetically generated traffic profiles that reproduce different typologies of DoS attacks. Our first objective is to obtain a high percentage of correct detections (*hits*) with a small amount of false alarms, and we also study the influence on such performance indicators of the shape, the duration and the amplitude of anomalies. Moreover, we evaluate the system capabilities in terms of the accuracy in the estimation of the anomaly time interval, considering also the occurrence of special cases as close anomalies, pulsing attacks, etc. This capability of a network anomaly detection system represents a novel aspect which has not been investigated much in past literature. The results reported in this paper show that the proposed approach presents good performance and interesting features that make it attractive.

The rest of the paper is organized as follows. We discuss related literature in Section 2, commenting on the main differences with other detection approaches in which the wavelet transform was specifically applied. In Section 3 we provide some background analytical information that justifies the techniques adopted. In Section 4 details on the system architecture and algorithms implemented are given. In Sections 5,

6 and 7, we describe, respectively, the traffic traces and anomalies that have been used for the experimental tests, the mother wavelet adopted in such computations, and the detection results obtained in terms of performance. Finally, in Section 8 we draw conclusions and foresee future works.

2. Related works

In past literature, approaches of very different nature for the design of network anomaly detection systems have been proposed. Techniques based on pattern-matching algorithms, finite-state machine models, statistical analysis, and signal processing have shown promising results. In this paper we focus on the last two approaches, which have brought a large amount of interesting literature. Some of these works analyze the aggregate traffic volume on a network link, others consider different flows carried on several links of an ISP, finally others restrict their focus to few typologies of attacks by looking at the time series of specific kinds of packets inside aggregate traffic (e.g. SYN packets). SNMP MIB variables from network devices are another common source of data. A mathematical model based on exponential smoothing and Holt–Winters forecasting was adopted in [4] for performing aberrant behavior detection on time series stored by a network monitoring software. Standard change point detection approaches as the Cumulative Sum algorithm (CUSUM) [25], have been proposed in a lot of different works and they have been applied to different typologies of time series [3,29,33]. In [29], the CUSUM algorithm and an adaptive threshold algorithm were used to detect syn flooding attacks. The proposed system analyzed the time series of the number of TCP SYN packets observed on a network link within a specific time period. Syn flooding attack detection using a CUSUM-based approach was proposed in [33] as well. In [3], instead, the authors propose a multistage sequential procedure with batch processing within individual stages. It represents the combination of a fixed-size batch detection and a sequential change point detection technique. They call it *Batch-Sequential* method and apply it to the detection of network Denial of Service attacks in general. In [31] an auto-regressive (AR) process is used to model abrupt changes in network time series derived from SNMP MIB variables stored in the devices of a network under observation. This work is based on the assumption that in the event of an anomaly, abrupt changes should propagate through the network, and they can be traced as correlated events among the MIB variables of different nodes. This correlation property would allow to discriminate the abrupt changes intrinsic to anomalous situations from the random changes of the variables related to normal network operation. This approach is verified in different contexts with various kinds of network anomalies.

Furthermore, very recent statistical approaches, instead of classical change-point detection techniques, have investigated the adoption of alternative methods as principal component analysis [17] or maximum entropy estimation [11].

As regards approaches more related to the time-frequency domain, Cheng et al. [8] propose the use of spectral analysis to identify legitimate TCP flows, which should exhibit strong periodicity. This is proposed as a complementary approach to existing DoS detection and defense mechanisms that identify attacks. The use of the Fourier transform has also been proposed in [23] and [16]. However, in all these works the focus is more on fingerprinting and on the recognition of different kinds of anomalies once a candidate anomaly inside a specific time interval has already been identified. In [23] the authors state that for strong transients in the attack fingerprint, since such features are not well captured by a Fourier analysis, a wavelet transform was initially applied to the data. The very good time- and scale-localization abilities of the wavelets, indeed, make them ideally suited to detect irregular traffic patterns in traffic traces. For these reasons, in the past few years, we have seen the publication of several works based on the wavelet transform, especially with approaches geared towards automated detection and good time localization of the anomaly. In [2] Barford et al. apply wavelet analysis and synthesis techniques to evaluate the signal of traffic volume filtered only at certain scales, and a thresholding technique is used to detect changes caused by events like outages, attacks and flashcrowds. The authors of [15] show that network problems affecting dominant Round Trip Times can be detected through the analysis of the energy function of the wavelet coefficients, obtained with the discrete *Haar* wavelet transform, at the corresponding scales. In [20] the authors exploit a property of the energy function calculated at a specific set of scales that is related to some network misconfigurations; while, in [18], spikes in the energy function of the wavelet coefficients are connected to DoS attacks.

In this work we propose an approach to network anomaly detection based on the wavelet transform, which we tested against several kinds of DoS attacks. Such approach presents substantial differences with past works. Firstly, we make use of the Continuous Wavelet Transform (CWT), exploiting its interpretation as the cross-correlation function between the input signal and wavelets and its redundancy in terms of available scales and coefficients. All the cited works, instead, are based on the use of the Discrete Wavelet Transform (DWT), which is more oriented to the decomposition of the signal over a finite set of scales, each one with a reduced number of coefficients, in order to make the original signal reconstructible from them. This is typically done in a way that avoids redundancy. Secondly, our detection approach takes explicitly into account – beside *hits* and false alarms – the accuracy in the estimation of the time interval during which the anomalous event happens and the resolution (in terms of ability to distinguish between subsequent anomalies). In the context of security incidents, these aspects can be crucially important, for example, for tracing back the source of an attack, or during forensic analysis. Thirdly, we propose a cascade architecture made of two different systems – the first one based on classical ADS techniques for time series, the second one based on the analysis of wavelet coefficients – which allows more flexibility and performance improvements as regards the *hits*/false alarms trade-off. Finally, as fourth point, we present an experimental analysis of the performance of the system under an extensive set

(≈15,000) of attack – traffic trace combinations. This paper represents an extension of a preliminary work presented at [9]. Aside from extending discussions related to the architecture proposed and related works, here, after a complete revision of the framework and clarifications on several points regarding the analytical basis, we present more details and experimental results.

3. An analytical basis

The Continuous Wavelet Transform (CWT) is defined as:

$$f_{\mathrm{CWT}}(a,b) = \int_{-\infty}^{+\infty} f(t)\psi_{ab}^{*}(t)\,\mathrm{d}t = \langle f(t)|\psi_{ab}(t)\rangle, \tag{1}$$

where:

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \tag{2}$$

$f(\cdot)$ is the signal under analysis, $\psi(\cdot)$ is a function of finite energy whose integral over $\mathbb{R}$ is 0, called *mother wavelet*, and a and b are the scaling and translation factors respectively (that are applied to the mother wavelet to obtain a scaled and translated version $\psi_{ab}(t)$). Each (a,b) pair furnishes a *wavelet coefficient*, which can also be seen as the cross-correlation at lag b between $f(t)$ and the mother wavelet function dilated to scaling factor a. An important difference between the CWT and the DWT is that the former calculates such correlation for each lag at every possible scale, whereas the DWT calculates a number of coefficients that decreases with the scaling factor. More precisely, the CWT differs from the more common DWT because the same number of coefficients is always obtained at each scale, while in the DWT the number of coefficients diminishes as the value of a increases. Moreover, the use of the DWT is more focused to signal decomposition avoiding redundancy, whereas the CWT can be easily interpreted as a cross-correlation function at several scales: at each scale "a" we can see the series of the coefficients as b varies as the cross correlation function between the signal and the mother wavelet (scaled by "a") at lag b.

The scale of the coefficients global maximum, is where the input signal is most similar to the mother wavelet. This function is chosen to be oscillating but with a fast decay from the center to its sides, in order to have good scale (frequency) and time localization properties. This makes the CWT a good tool for analyzing transient signals as network traffic time series. When the CWT is implemented as a numeric algorithm, b can assume a number of values equal to the number of samples N of the input signal and the scaling factor a is expressed by $a = 2^{(-j+m/M)}$, where j is the *octave*, m is the *voice* index ($0 < m < M$), and M is the number of voices

per octave. The number of octaves is given by $J = [\log_2 N] - 1$, where the operator $[\cdot]$ returns the nearest integer to its argument. The parameter j can vary between 2 and $J + 2$. M usually ranges between 0 and 12; the greater M is, the better is the frequency resolution of the wavelet transform obtained.

In the study of wavelets and image processing, it has been proven that the local maxima of a wavelet transform can detect the location of irregular structures in the input signal [21]. According to this, by applying the CWT to network traffic time series, here we propose to use the property of detecting the location of an irregular structure as the indication of an abrupt change in the traffic time series and, therefore, as an indication of an abnormal event.

Let us consider a *smoothing* function $\theta(t)$, that is the impulsive response of a low-pass filter, such that $\theta(t) = \mathrm{O}(1/(1 + t^2))$ and whose integral is not zero (e.g. the Gaussian function). Given $\theta_a(t) = (1/a)\theta(t/a)$, let $f(t)$ be a real square-summable (over $\mathbb{R}$) function. The edges of $f(t)$ at scale a can be defined as the points of rapid local changes of $f(t)$ filtered by $\theta_a(t)$.

Given two mother wavelets defined as:

$$\psi^1(t) = \frac{\mathrm{d}\theta(t)}{\mathrm{d}t} \quad \text{and} \quad \psi^2(t) = \frac{\mathrm{d}^2\theta(t)}{\mathrm{d}t^2}, \tag{3}$$

the corresponding CWTs are:

$$f^1_{\mathrm{CWT}}(a,t) = f * \psi^1_a(t) \quad \text{and} \quad f^2_{\mathrm{CWT}}(a,t) = f * \psi^2_a(t), \tag{4}$$

where:

$$\psi^1_a(t) = \frac{1}{a}\psi^1(t/a) = a\frac{\mathrm{d}\theta_a(t)}{\mathrm{d}t}, \tag{5}$$

$$\psi^2_a(t) = \frac{1}{a}\psi^2(t/a) = a^2\frac{\mathrm{d}^2\theta_a(t)}{\mathrm{d}t^2}. \tag{6}$$

Substituting in (4), we obtain:

$$f^1_{\mathrm{CWT}}(a,t) = f * \left(a\frac{\mathrm{d}\theta_a}{\mathrm{d}t}\right)(t) = a\frac{\mathrm{d}}{\mathrm{d}t}(f * \theta_a)(t), \tag{7}$$

$$f^2_{\mathrm{CWT}}(a,t) = f * \left(a^2\frac{\mathrm{d}^2\theta_a}{\mathrm{d}t^2}\right)(t) = a^2\frac{\mathrm{d}^2}{\mathrm{d}t^2}(f * \theta_a)(t). \tag{8}$$

Thus, $f^1_{\mathrm{CWT}}(a,t)$ and $f^2_{\mathrm{CWT}}(a,t)$ are proportional to the first-order and second-order $f(t)$ derivative respectively, filtered by $\theta_a(t)$. Such properties are obviously maintained by derivatives of greater order. It follows that, for a fixed scale a, the local extrema of $f^1_{\mathrm{CWT}}(a,t)$ along t correspond to the zero-crossings of $f^2_{\mathrm{CWT}}(a,t)$ and to the inflection points of $f * \theta_a(t)$. Thus, using the derivative of a smoothing function

as a mother wavelet (e.g., derivatives of the Gaussian function), the zero-crossings or the local extrema of the wavelet transform applied to a signal indicate the locations of its sharp variation points and singularities. The CWT coefficient redundancy, allows to identify these points at every scale with the same time-resolution of the input signal.

4. The cascade architecture

In Fig. 1 a block diagram representing the two-stage architecture of the proposed ADS is shown. The ADS takes as input a time series of samples representing the packet rate and outputs an ON–OFF signal reporting the presence of an anomaly for each sample. The first stage, which we called *rough detection*, can be implemented using statistical anomaly detection techniques previously presented in literature, and it is just responsible to detect any suspicious change in the traffic trend and to report an alarm to the second stage. Its output is equal to 0 or 1 for each input sample. Here we impose a high sensitivity aiming at catching as much anomalies as possible, whereas the second stage, which we called *fine detection*, is designed to reduce the number of false alarms. For each detected anomaly, this stage also estimates the time interval during which it is present.

Before going into the details of each component of the proposed architecture, to provide a first look at the entire system we point the attention of the reader to Fig. 2, anticipating some of the concepts that will be explained in the next sections of the paper. Figure 2 shows a test on a sample attack-free trace from Darpa 1 dataset (see description of Table 1), to which we superimposed the profile of a *TCP ACK flood* anomaly. The anomaly profile, which is highlighted by a frame in the upper diagram, starts from sample 1800, is of 200 samples length, and has been scaled so that its maximum peak is equal to 1.5 times the root mean square of the attack-free

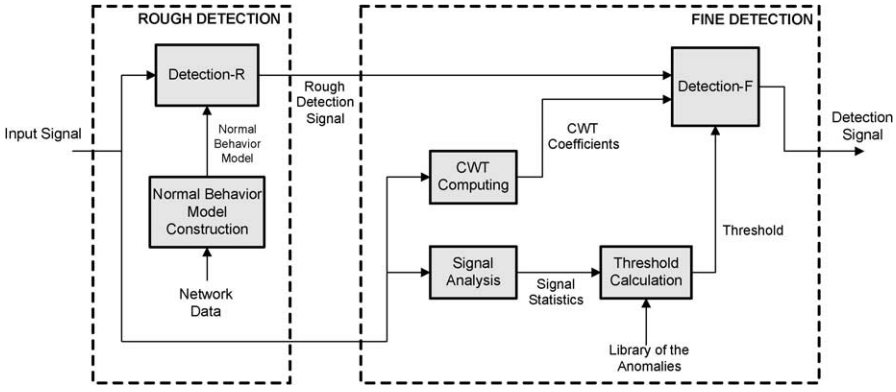


Fig. 1. Anomaly detection system: proposed architecture.

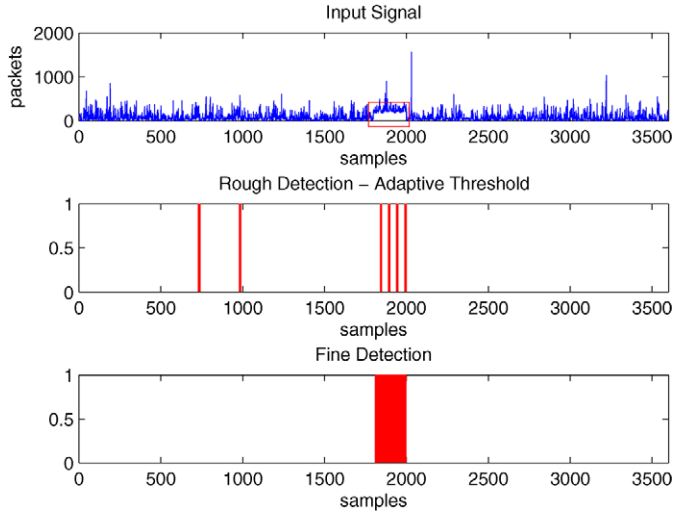


Fig. 2. Exemplifying test on a single trace (Darpa 1 with StachelDraht TCP ACK flood): anomaly correctly detected.

Table 1
Traffic traces

Data set	Year	T_s (s)	# Traces	Mean	Std
Darpa 1	1999	2	5	80 pkt	90 pkt
Darpa 2	1999	5	5	20 pkt	40 pkt
Darpa 3	1999	5	5	12 pkt	30 pkt
UCLA	2001	2	4	20 pkt	15 pkt
UNINA	2004	2	3	8 10E3 pkt	1.3 10E3 pkt

trace (refer to Section 7). The output of the first stage, the *rough detection* module, is shown in the diagram in the middle. Here, an adaptive threshold algorithm has been used for this module. Whereas the diagram at the bottom of the figure shows the final output of the system (the output of the *fine detection* module).

Following the definitions which will be given in the next sections, the output parameters of an automated run of this sample test are the following:

- test_result $= 1$ (the test was successful: the anomaly has been detected);
- FP $= 0$ (there have not been false positives);
- FN $= 0$ (there have not been false negatives);
- left_lag $= +12$ (the begin of the anomaly has been estimated with a delay of 12 samples);
- right_lag $= 0$ (the end of the anomaly has been estimated correctly);
- fragments $= 1$ (the detection of the anomaly led to a single alarm, that is the anomaly interval estimated is only one, no fragments).

4.1. First stage: Rough detection

As for the rough detection module, we adopted the two alternative techniques proposed in [29] to detect SYN flooding attacks (an adaptive threshold algorithm and the CUSUM algorithm) and we applied them to generic traffic traces. A similar implementation of the CUSUM algorithm has also been proposed in [3] to detect different DoS attacks. The system proposed in [29] analyzed the time series of samples representing the number of TCP SYN packets observed on a network link. Such algorithms can be applied in the same way to time series representing the number of IP packets observed on a link, as we do in our approach.

The adaptive threshold (AT) algorithm generates an alarm when the value of a sample is greater than a threshold that adaptively changes with the traffic trend. Such threshold is adaptive, indeed it changes depending on recent measures. This is good because of the typical high non-stationarity of network traffic. Let x_n be the number of packets during the nth time interval and let $\bar{\mu}_{n-1}$ be the mean rate estimated from measurements prior to n, an alarm at time n is signaled if:

$$\sum_{i=n-k+1}^{n} 1_{[x_i \geqslant (\alpha+1)\bar{\mu}_{n-1}]} \geqslant k, \tag{9}$$

where α determines the threshold sensitivity, and $1_{[x \geqslant y]}$ is equal to 1 if $x \geqslant y$, to 0 otherwise. The average value $\bar{\mu}_n$ is calculated using the Exponentially Weighted Moving Average (EWMA) on the previous estimates:

$$\bar{\mu}_n = \beta\bar{\mu}_{n-1} + (1-\beta)x_n, \tag{10}$$

where β is the EWMA factor. The configurable parameters of the algorithm are: α, β and k.

The CUSUM algorithm is based on the change-point detection theory, and uses the *log-likelihood ratio*:

$$S_n = \sum_{i=1}^{n} s_i, \tag{11}$$

where $s_i = \ln \frac{p_{\theta_1}(y_i)}{p_{\theta_0}(y_i)}$ and $\{y_i\}$ are random variables. The θ_0 and θ_1 hypotheses represent the statistical distributions prior and after a change respectively. The *log-likelihood ratio* guarantees a negative drift before a change and a positive drift after the change. Therefore, let $m_n = \min_{1 \leqslant j \leqslant n} S_j$, an alarm is signaled when $g_n = S_n - m_n \geqslant h$, where h represents the threshold. After some calculations [29], an expression of g_n based on the mean and variance of θ_0 and θ_1 can be derived (we assume the two distributions have different mean, μ_1 and μ_2, respectively, but same

variance σ). However, $\{y_i\}$ are assumed as independent Gaussian variables. Because this is generally not true for network traffic, algorithms to remove trends and time correlations should be applied to the input signal. A common and simpler approach is to subtract from the considered time series its EWMA. We therefore apply the CUSUM algorithm to $\tilde{x}_n = x_n - \bar{\mu}_{n-1}$, where x_n is the number of packets in the nth time interval and $\bar{\mu}_n$ is an estimate of the mean rate at time n (calculated using the same EWMA as in the adaptive threshold algorithm). Taking into account that the mean value of $\tilde{x}_n$ prior to a change is 0, and approximating the mean traffic rate after the change with $\alpha\bar{\mu}_n$, g_n can be expressed as:

$$g_n = \left[g_{n-1} + \frac{\alpha\bar{\mu}_{n-1}}{\sigma^2} + \left(x_n - \bar{\mu}_{n-1} - \frac{\alpha\bar{\mu}_{n-1}}{2} \right) \right]^+. \tag{12}$$

The algorithm configurable parameters are: α, β and h.

4.2. Second stage: Fine detection

The *CWT computing* block (Fig. 1) computes the continuous wavelet transform of the whole input signal. We used the Wavelab [14] set of routines under the Matlab environment. The block output is a matrix W of M rows and N columns, where N is the number of samples of the input trace. Each row reports the wavelet coefficients at a different scale. The number of available scales M is given by the number of octaves, $J = [\log_2 N] - 1$ times the number of voices per octave. The CWT function implemented under Wavelab allowed us to work with 12 voices per octave (this to obtain a good frequency resolution). This matrix is fed as an input to the *Detection-F* block, which receives as inputs also a *threshold* level (that will be explained in the following) and the *Rough Detection Signal*. For each alert reported in the *Rough Detection Signal*, the *Detection-F* block operates as follows:

- In the column of W that corresponds to the instant of the alert, the maximum value is found. The row index j_1 of this value represents a first estimate of the scale at which the anomaly is present.
- Looking at all the coefficients at the scale j_1, the zero-crossings (starting from the left and right of the maximum value) are determined. Their distance represents a first estimation of the anomaly interval.
- A sub-matrix of W, obtained by considering only the columns related to this interval, is used for a new search. In this sub-matrix, a new maximum coefficient is found. The index j_2 of its row represents the final estimated scale.
- An anomaly is found if the maximum coefficient results greater than the threshold level. Otherwise the rough detection alarm is ignored and no other operations need to be performed.
- The final estimation of the anomaly interval is made by looking at the coefficients at the scale j_2. Again, the interval boundaries are identified by searching for the zero-crossings at the left and right of the maximum value.

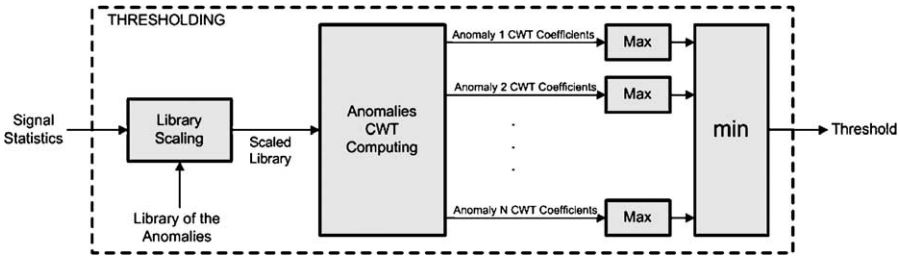


Fig. 3. Threshold calculation block.

Basically, starting from the alert of the rough detection stage, we look for the scale at which the coefficients reach the maximum variation. The use of the CWT guarantees that we have a coefficient for each input sample at every scale – differently from the DWT, where typically the number of coefficients decreases as the scale grows. This way, if an anomaly is recognized, we can identify with good precision the zero-crossing points of the wavelet coefficients at the scale where the anomaly is present.

The choice of the threshold level for the wavelet coefficients (*Threshold Calculation* block) is based on the mean and standard deviation of the traffic trace (*Signal Analysis* block) and on the *Library of Anomalies*, which is a collection of signals representing some traffic anomalies (see Section 5.2). Inside the *Threshold Calculation* block (Fig. 3), in the sub-block named *Library Scaling*, all the anomaly signals are scaled to a maximum peak value of $p_{\max}$. This value is given by the standard deviation of the input trace multiplied by a factor, for which we have chosen three possible values corresponding to different ranges of the mean/standard deviation ratio of the input trace. This is because we want to make the threshold calculation adaptive with respect to the trace characteristics. After that all the anomalies have been scaled, for each anomaly k in the library the CWT of the scaled anomaly signal is computed, and the maximum m_k among all the coefficients is found. Finally, the threshold is obtained as min(m_k), that is the smallest of the maximum coefficients of each anomaly.

5. Traffic traces and anomalies

To study and develop our ADS, we made several experiments under a broad range of situations. Our approach was to generate traffic signals by superimposing anomaly profiles to real traffic traces in which no anomalies were present, an approach commonly used in literature [5,10,29]. This choice is partly due to the scarce availability of traffic traces containing classified anomalies along with all the necessary details. For example, the lack of information on the exact beginning and end of each anomaly would not allow us to evaluate the temporal precision of the detection system. On the other hand, being able to generate different traces containing anomalies

allowed us to dispose of much more test cases than those practically obtainable by capturing real traffic traces with real anomalies. In the following subsections we give some details on the data we used.

5.1. Traces

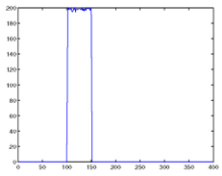
We considered real traffic traces that were known not to contain any anomalies, obtaining a large and heterogeneous set of traces. In Table 1 the data sets we used are summarized. The first three groups of traces in Table 1 were derived from the DARPA/MIT Lincoln Laboratory off-line intrusion detection evaluation data set [19], which has been widely used for testing intrusion detection systems and has been referred in many papers (e.g. [28,32]). We used only traces from the weeks in which no attacks were present. The dataset marked in Table 1 as *UCLA* refers to packet traces collected during August 2001 at the border router of Computer Science Department of University of California Los Angeles [12]. They have been collected in the context of the D-WARD project [22]. Finally, the *UNINA* data set refers to traffic traces we captured by passively monitoring ingoing traffic at the WAN access router at *University of Napoli "Federico II"*. We make the time series representing the sampled packet rate publicly available at [13]. Table 1 contains details about the data sets, as the number of traces for each group and the sampling period T_s used to calculate the packet rate time series. Also, indicative values of mean and standard deviation (*std*) for the traces of the same set are shown. All traces are composed of 3600 samples.

5.2. Anomalies

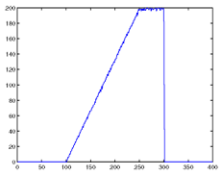
Anomalies in network traffic can be of different nature and can be originated by different kinds of events. It is possible to distinguish among network performance problems and failures (temporary or permanent problems on nodes or links), non-malicious but unordinary events (e.g. flashcrowds) and malicious events (e.g. DoS attacks). These events tend to determine an abrupt change in the time series representing the traffic rate. In this work, several kinds of anomaly profiles related to DoS attacks have been synthetically generated. We assigned labels to each anomaly we used (see Table 2 for the complete list, and Fig. 4 for understanding the envelopes of some of them). Some anomaly profiles were obtained by generating traffic with real

Table 2
Tested anomalies

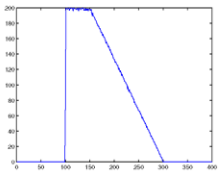
Tools	Matlab	TFN2K	Stacheldraht
Anomalies	Constant rate, increasing rate, decreasing rate	ICMP Ping flood, TCP SYN flood, UDP flood, mix flood, Targa3 flood	TCP ACK flood, TCP ACK NUL flood, TCP random header attach, Mstream (TCP DUP ACK), DOS flood, mass ICMP bombing, IP header attack, SYN flood, UDP flood



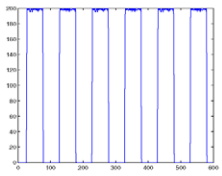
(a) Constant Rate generated with Matlab



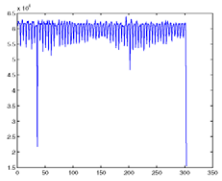
(b) Increasing Rate generated with Matlab



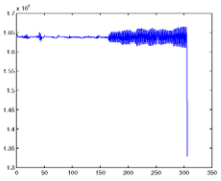
(c) Decreasing Rate generated with Matlab



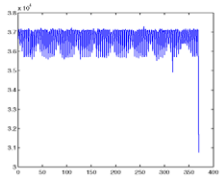
(d) Pulsing Attack



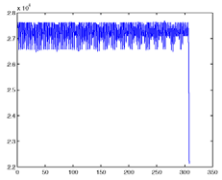
(e) ICMP Ping flood generated with TFN2k



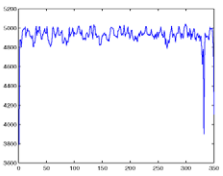
(f) TCP SYN flood generated with TFN2k



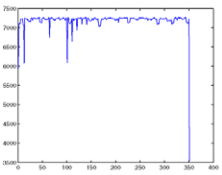
(g) UDP flood generated with TFN2k



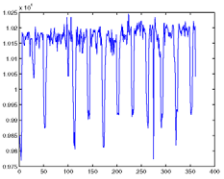
(h) Mix flood (ICMP, TCP, UDP) generated with TFN2k



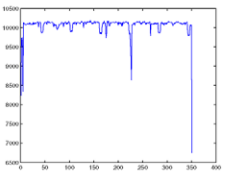
(i) TCP ACK flood generated with StachelDraht



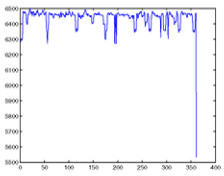
(j) TCP ACK NUL generated with StachelDraht



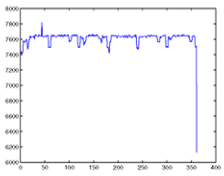
(k) TCP random header attack generated with StachelDraht



(l) Mstream generated with StachelDraht



(m) ICMP ping flood generated with StachelDraht



(n) IP header attack (IP storm) generated with StachelDraht

Fig. 4. Packet rate profiles of some of the anomalies used.

DoS attack tools, TFN2K [6] and Stacheldraht [7]. Both tools can be configured to perform DoS attacks using several known techniques. We launched such tools with several different options and we captured the traffic generated by them. The anomaly profiles obtained were stored and labeled depending on the adopted attacking technique. Another group of anomalies have been obtained by synthetically generating the corresponding time series with Matlab, according to known profiles that were considered in [34]. We considered 'Constant Rate', 'Increasing Rate' and 'Decreasing Rate' anomalies. Even if in this paper we focus our attention only on DoS attacks, it is worth noticing that the envelope of some of the considered anomalies due to DoS attacks may be also observed in other kind of anomalies (e.g., due to worms, misconfigurations, flashcrowds, links failures, ...).

6. Fine detection block: Choice of the mother wavelet

According to the description given in Section 3, it is clear the central role of the selected mother wavelet. After several trials, in our tests we computed the CWT using the *Morlet* mother wavelet (see Fig. 5), which has the following expression:

$$\psi(t) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\mathrm{j}w_0 t} \mathrm{e}^{-t^2/2}. \tag{13}$$

The Morlet mother wavelet is one of the most used in signal processing because of its good properties as symmetry and a narrow and rapidly decreasing central lobe. Usually $w_0 = 5$ is chosen, to have the second lobe half of the first one. Such properties translate into good time and scale localization capabilities.

We found a strong similarity with even-order derivatives of a Gaussian, for which there is a strong analytical basis for their use in the field of singularity detection

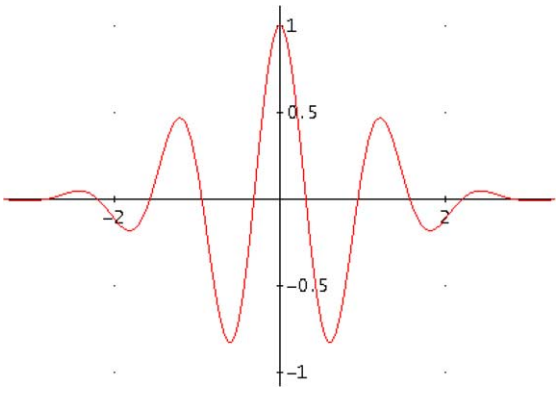


Fig. 5. Real Morlet Mother wavelet ($\omega_0 = 5$).

(see Section 3). We verified such similarities also by calculating the cross-correlation function between the coefficients of the wavelet transform of the tested anomalies using even-order Gaussian derivatives and Morlet mother wavelets. We found a cross-correlation value of 0.96 between coefficients obtained with the 24th order Gaussian derivative and the Morlet mother wavelet with $w_0 = 5$. Experimental tests with even-order derivatives of the Gaussian function of smaller orders – e.g., the *Mexican Hat* mother wavelet [1] which is the 2nd order derivative – showed a remarkable loss of accuracy in finding the start and the end of each anomaly. As regards odd-order derivatives, we did not take them into account because singularity detection using odd-order derivatives of a *smoothing* function is based on the identification of local maxima (see Section 3). A search for local maxima is more difficult to implement rather than a search for zero-crossing points, which is the case for even-order derivatives.

7. Experimental results

The experimental results shown in the following have been obtained by performing a large set of automated tests. The results have been summarized and the following performance metrics have been calculated:

1. The hit rate, $HR = \frac{number\ of\ test\ hits}{number\ tests} \times 100$;
2. The false alarms ratio, $FAR = \frac{number\ of\ false\ alarms}{total\ number\ of\ alarms} \times 100$;
3. The estimation errors in the identification of the beginning and the end of the anomaly;
4. The number of fragments when a single anomaly is recognized as several ones.

Our scripts generated traces containing anomalies with various combinations of parameters and ran the ADS on each of them. In order to test the ADS under more complicated situations (i.e. obfuscating the anomalies in the traces), when a trace and an anomaly profile are selected, the amplitude and the duration of the signal representing the anomaly are modified. Then the signal is superimposed to the traffic trace at a randomly selected point – at $1/4$, $1/2$ or $3/4$ of the trace – and the detection system is executed. For a specific trace, the amplitude of an anomaly was scaled in order to make its maximum peak proportional to the root mean square of the original traffic trace. The choice of the proportionality factor varies from 0.5 to 2.00 with a step of 0.25. Anomaly durations range from 50 to 300 samples with a step of 50. Sampling and interpolation of the anomaly profiles were performed for expansion and shortening respectively. Thus we performed a number of tests given by the product (*traces* $\times$ *anomalies* $\times$ *intensities* $\times$ *durations*). With 22 traces and 16 anomalies, we performed about 15,000 tests, each time we tested a system configuration (i.e. with CUSUM, with AT).

7.1. Hit rate (HR) and false alarm ratio (FAR)

In Table 3 we show the system performance, in terms of *HR* and *FAR*, when the rough detection block is implemented with AT and CUSUM algorithms. We report results obtained separately for each of the 5 trace data sets, and in the last row, we show global results obtained working with all the traces. The columns labeled *FD*(*AT*) and *FD*(*CUSUM*) report performance indicators derived from the output of the fine detection stage when the rough detection stage are *AT* and *CUSUM*, respectively. Instead, the performance results related only to the output of the rough detection stages are reported in columns labeled with *RD*(*AT*) and *RD*(*CUSUM*). This is to show how we tuned the rough detection stage with a very high sensitivity in order to catch as much anomalies as possible at the expense of a high *FAR*. We tuned the α, β, and k (h in the case of the CUSUM algorithm) accordingly. For all the traces β was set to 0.98, k to 4 or 5; h was set to 5 except for UCLA and UNINA traces, where $h = 12$ and $h = 15$ were respectively chosen; the α parameter was chosen with values between 0.1 and 0.4 depending on the traces.

Passing from the rough detection output to the fine detection output, while *HR* remains almost the same, *FAR* decreases dramatically. This happens for all the sets of traces, and for both AT and CUSUM, and it represents one of the most important features of the proposed ADS.

In order to sketch a comparison between the proposed two-stage ADS and AT or CUSUM used as standalone algorithms, in the columns labeled as *AT-sa* and *CUSUM-sa* we show how they perform in terms of *HR* when tuned with approximately the same *FAR* of the proposed ADS. We see that, in the case of AT, the introduction of the second stage, improves *HR* of about 10% for 3 out of 5 trace sets, as for AT, while for CUSUM the improvements range from about 12%, for the fifth trace set, to almost 50%, for the first one.

In Fig. 6 we show how *HR* and *FAR* are influenced by the relative amplitude (left figures) and the duration (right) of the anomalies. Top and bottom figures refer to the system with AT and CUSUM rough detection, respectively. We evaluated performance separately for each anomaly profile. We observed that the *increasing rate* and *decreasing rate* anomalies are more difficult to be detected, compared to the other

Table 3
HR/FAR trade-off results

Dataset	*RD*(*AT*)		*FD*(*AT*)		*RD*(*CUSUM*)		*FD*(*CUSUM*)		AT-sa		CUSUM-sa	
	HR	*FAR*	*HR*	*FAR*	*HR*	*FAR*	*HR*	*FAR*	*HR*	*FAR*	*HR*	*FAR*
Darpa 1	95.9	72.8	89.5	34.9	84.0	68.6	82.4	1.56	79.0	35.3	35.1	6.7
Darpa 2	93.7	68.2	84.9	38.0	85.7	83.6	84.8	38.9	74.1	36.4	49.4	32.6
Darpa 3	92.1	81.1	83.8	50.1	88.3	77.9	84.7	28.1	71.6	51.0	62.7	25.0
UCLA	90.9	17.7	86.0	14.0	91.5	89.6	86.2	39.8	85.7	15.8	56.3	44.4
UNINA	99.6	69.7	98.0	7.4	99.6	77.3	98.0	12.1	86.4	7.0	78.6	13.1
All	94.2	70.9	87.7	34.1	83.7	86.2	86.3	27.2	79.4	33.1	49.2	33.9

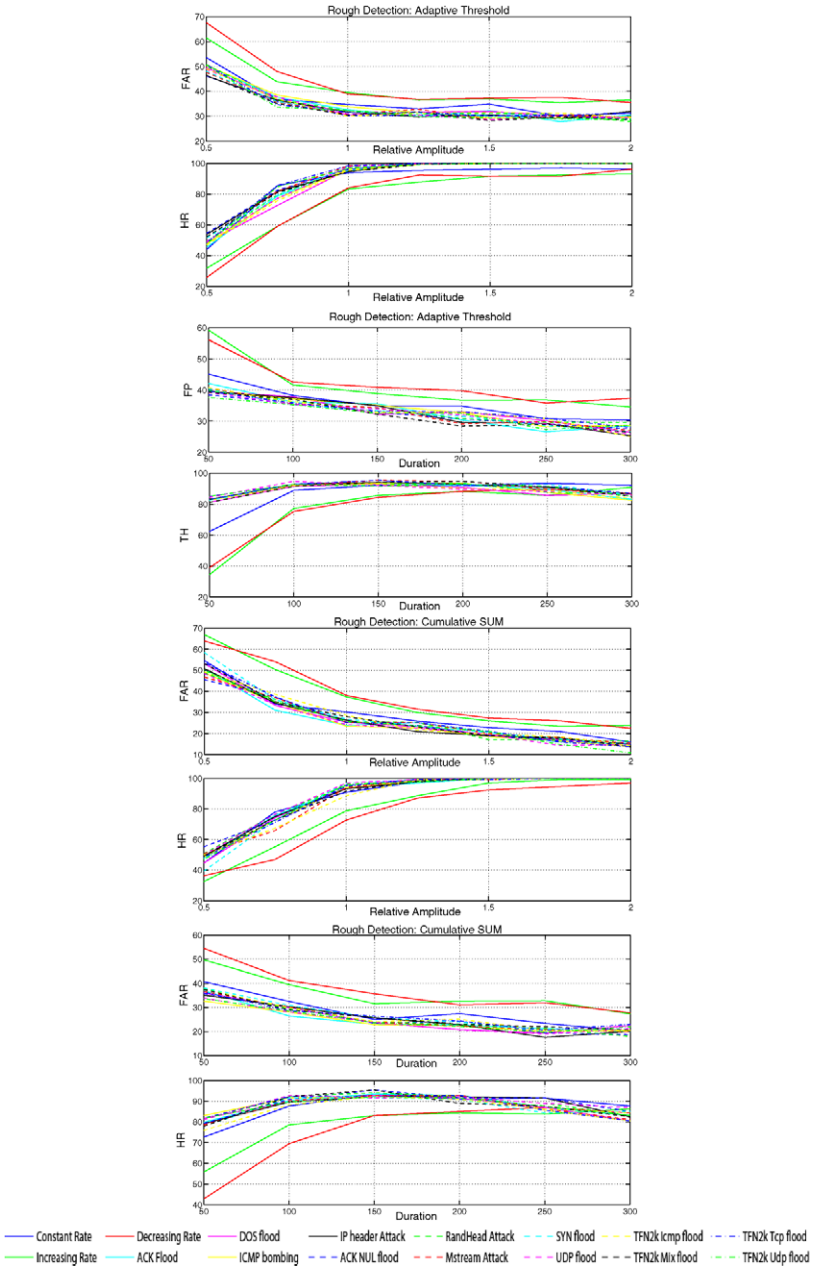


Fig. 6. HR and FAR as functions of attacks' relative amplitude and duration.

anomalies. However, it is interesting to note that the curves related to all the anomaly profiles follow approximately the same trends. The relative amplitude has more influence on *HR* and *FAR* than the anomaly duration. But, when the anomaly amplitude is tuned for peak values greater than the RMS of the trace (relative amplitude $\geqslant 1$) *HR* does not increase anymore. A similar behavior happens for *FAR* in the AT case, while as for the CUSUM implementation *FAR* tends to slowly decrease even after the relative amplitude is higher than 1. As regards the anomaly duration, while *FAR* always decreases when the anomaly lasts longer, *HR* inverts this trend after a certain duration. This behavior is accentuated in the CUSUM case.

7.2. Accuracy in the detection of the anomaly time interval

The diagrams in Fig. 7 show the percentage of correct estimates of the start and the end time of the anomalies, when the width of the confidence interval (expressed

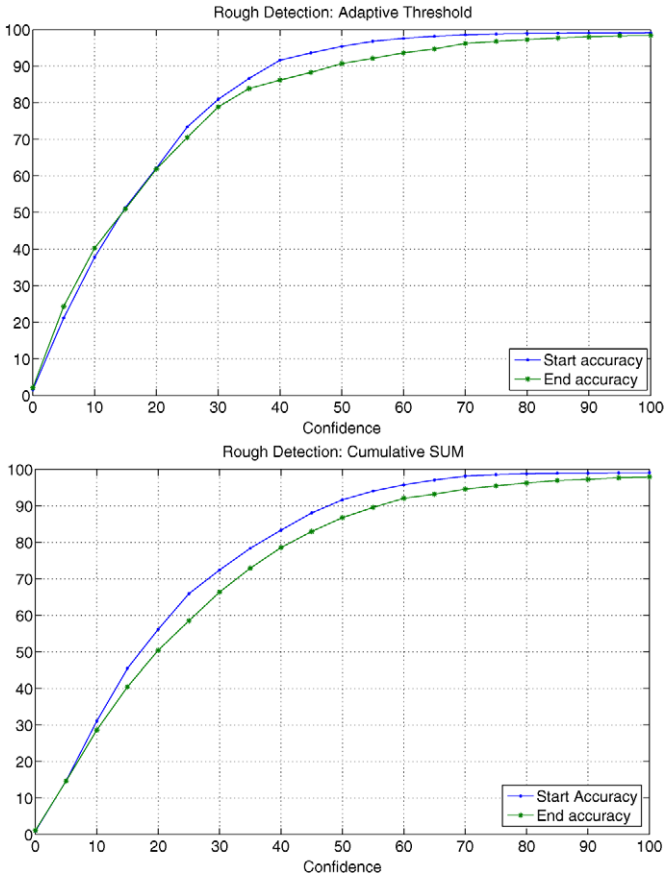


Fig. 7. ADS accuracy.

in number of samples) increases. We consider the estimate to be correct when the start/end time falls into the confidence interval. For a confidence interval of 30 samples, 70% of the start and end times are correctly identified. In general, we note a slightly better performance in the estimation of the start time compared to the end time. We also evaluated when the system did not correctly estimate the anomaly duration because the anomaly was recognized as several different anomalous events. This occurred rarely: for only 4.62% of the detections with the AT rough detection block, and 1.62% with CUSUM.

7.3. Resolution

With the term *resolution* we mean the minimum distance at which two anomalous events can be placed for the system to detect them as distinct anomalies. We made several tests by superimposing two anomalies to the same trace. We varied their distance, duration and amplitude. The system seems to perform very well, detecting two separate anomalies even at small distances. In Fig. 8 we show two examples. In the top diagrams, we used a trace from the DARPA 2 set, to which we superimposed an *UDP flood* and an *IP header* attack at the distance of 5 samples. The rough detection block here is implemented using the AT algorithm. In the bottom diagrams a *constant rate* anomaly and a *stacheldracht TCP ACK flood* at the distance of 1 sample have been correctly detected (with a CUSUM rough detection) when they were superimposed to a trace from the DARPA 1 set. In both cases it can be seen how the system correctly identifies two distinct anomalies, whereas the rough detection stage fails to make this distinction: in the first test, the AT block reports several alerts all at the same distance, while in the second test the CUSUM block reports a series of alerts from the start of the first attack to the end of the second one (plus a false alert nearby).

These results, along with those related to fragmentation and accuracy in the previous subsection, show that the proposed ADS is reliable also in the identification of anomalies intervals. Such feature is even not considered by most of the other ADSs, which only report an alarm for each input sample that is recognized as anomalous (e.g. see the fragmented alerts from the AT rough detection stage in top Fig. 8).

7.4. Detection of pulsing attacks

Pulsing attacks represent a special case of Denial of Service attacks. They usually are used to cause a significant reduction of quality (RoQ attack) of a network service while trying to pass undetected because of their specific profile [26]. In a pulsing attack a single attacking node sends to the victim packets with a certain time period. The packet rate is not very high, so that the average value of the traffic generated can deceive common volume-based detection systems, but the peaks are still high enough that their succession will cause a service disruption. We separately tested the ability of the proposed anomaly detection system to properly detect this special

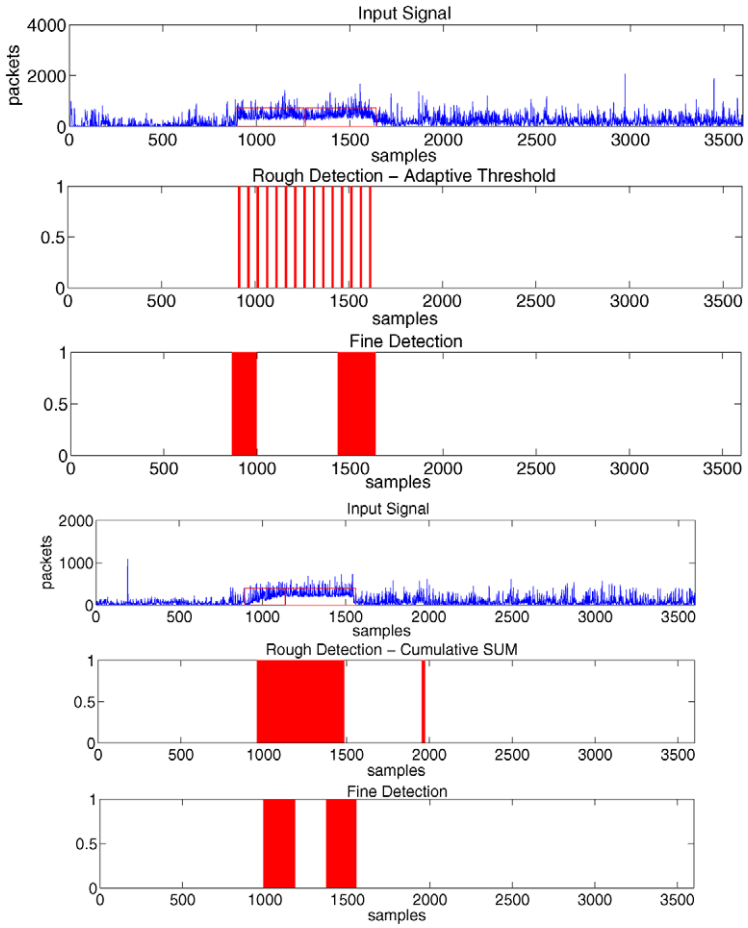


Fig. 8. ADS resolution: two examples.

typology of attacks by identifying the single pulses without being confused by their rapid succession and thus allowing a correct analysis of the attack. Figure 9 shows an example in which a pulsing attack is superimposed to a Darpa 1 trace and the rough detection module is the AT. We found that the system is able to correctly recognize all pulses and to accurately identify starting and ending sample points of each of them.

7.5. Considerations on computational constraints for an "online" implementation

To study and evaluate the proposed technique, in this paper we implemented a prototype of the algorithm, under the Matlab environment, working in a *offline* mode.

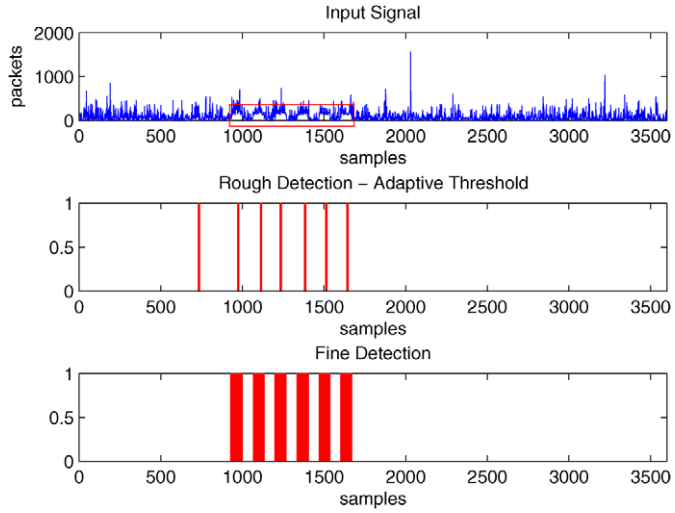


Fig. 9. Detection of a pulsing attack of 800 samples and relative amplitude 1 on a Darpa 1 trace.

In other words our implementation works with a full trace of a limited size and analyzes it *ex post*. Even if this working mode may be used for a real-world practical implementation, for example by executing the algorithm each x hours or on a daily basis, an important application of this technique would be to see it implemented in a *online* version. That is, a version of our technique able to continuously monitor network links and rapidly raise automatic alerts when anomalous events occur. Even if considering and evaluating an online implementation of the proposed technique falls out of the scope of the present paper, in this section, to allow a more complete evaluation of the new approach here proposed, we report qualitative information useful to understand under which conditions an online implementation would be feasible. These are mostly bound to the computational load associated to the different blocks composing the architecture.

Firstly, it is important to understand that implementing an online version of the algorithm of our detection technique would require slight modifications to the specific architecture presented here. Indeed, while the rough detection stages (both AT or CUSUM) are natively conceived as online algorithms (they can be run for each new sample) the operations of the second stage are designed to be applied to an interval, that is, a set of samples. A possible implementation in an online fashion would be to use a sliding window of time t_{w} over which the algorithms of the second stage could be run. Considering a sampling period for the packet rate signal equal to T we have a window of t_{w}/T samples that may slide for each sample. The arrival of each new input sample would cause the calculation of the first stage, plus the sliding of the samples window of 1 step, and the calculations of the second stage applied to this window.

Moreover, it is to be considered that some calculations of the second stage do not need to be run each time the window slides. This is the case of the CWT computation of the anomaly library. This operation can be done only once. This way the calculation of the threshold for the fine detection stage would require only few simple operations.

Under these conditions, a design requirement would be that the total time for the overall (first plus second stage) algorithm execution, each time a new input sample arrives, should be less than the sampling period T. The total computation time thus poses a constraint on the minimum sampling time to be used, a parameter related to the responsiveness of the online anomaly detection system. If we call t_1, t_2, t_3, t_4 the times necessary to respectively execute the rough detection stage, the CWT computation of the current window, the detection-F block, and the signal analysis plus threshold calculation blocks, then we have $t_1 + t_2 + t_3 + t_4 < T$. Where only t_2 and t_4 are functions of the window size t_w/T, and t_2 represents by far the highest computational cost.

In our prototype implementation under Matlab, running on a machine equipped with a Pentium 4 3.6 GHz, we verified that: t_1 is less 10 μs, t_3 is about 0.01 s, whereas, for a 3600 samples window, t_2 is around 0.15 s, and t_4 (considering the modified version cited above) is 0.02 s. These values show that the most intensive operation is the CWT computation of the current window. However, even in our prototype implementation the times observed are small enough to allow a sampling period of 1 second when a sliding window of 3600 samples (i.e. 1 hour) is considered. Moreover, an implementation for a production environment would be written in a more performing language or it would make use of dedicated hardware (e.g., ASICs or FPGAs), achieving much lower computation times.

8. Discussion and conclusion

This paper proposed a cascade architecture, working in an offline fashion and mainly based on the Continuous Wavelet Transform, to detect volume-based network anomalies caused by DoS attacks. We showed how the proposed schema is able to improve the trade-off existing between *HR* and *FAR*, and at the same time to provide insights on anomaly duration, by estimating starting and ending time intervals. Testing the proposed approach showed good results also under special cases as in the presence of subsequent close anomalies and with pulsing attacks. Our current work is focused on investigating several improvements, additions, and evaluating more properties under different conditions. E.g. extending the considered network anomalies to not only Denial of Service attacks, and evaluating our detection technique "on the field", testing it in a real network environment.

As for the rough-detection stage we are working both on testing other algorithms as well as trying to redesign the whole system to work without a first stage. We are also considering multi-channel change point detection approaches where the input is

split into several series, e.g. by separately considering the rate of packets of different nature (e.g., by transport protocol, by packet size, etc.), as proposed in [30]. Regarding the fine-detection stage we are working on more sophisticated approaches for the final detection decision, using correlation among wavelet coefficients and also trying to use this information for anomaly classification purposes. Moreover, we plan to further investigate and stress the ability of the system to work on the detection of anomalies at very different time scales. This would also allow to infer properties useful for the classification of the detected network anomalies.

Acknowledgements

This work has been partially funded by PRIN RECIPE Project, by the CONTENT EU Network of Excellence (IST-FP6-038423), and finally by European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 216585 (INTERSECTION Project).

References

[1] F. Argeso, J. Gonzalez-Nuevo, J.L. Sanz, L. Toffolatti, P. Vielva, D. Herranz and M. Lopez-Caniego, The Mexican hat wavelet family: application to point-source detection in cosmic microwave background maps, in: *EUSIPCO 2005, The 13th EURASIP European Signal Processing Conference*, Antalya, September 4–8, 2005.

[2] P. Barford, J. Kline, D. Plonka and A. Ron, A signal analysis of network traffic anomalies, in: *ACM SIGCOMM Internet Measurement Workshop*, Marseille, France, 2002.

[3] R.B. Blazek, H. Kim, B. Rozovskii and A. Tartakovsky, A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods, in: *IEEE Workshop Information Assurance and Security*, West Point, NY, USA, 2001, pp. 220–226.

[4] J. Brutlag, Aberrant behavior detection in time series for network monitoring, in: *USENIX Fourteenth System Administration Conference LISA XIV*, New Orleans, LA, USA, December 2000.

[5] G. Carl, R.R. Brooks and S. Rai, Wavelet based denial-of-service detection, *Computers & Security* **25**(8) (2006), 600–615.

[6] CERT Coordination Center, Denial-of-service tools – Advisory CA-1999-17, http://www.cert.org/advisories/CA-1999-17.html, December 1999.

[7] CERT Coordination Center, DoS Developments – Advisory CA-2000-01, http://www.cert.org/advisories/CA-2000-01.html, January 2000.

[8] C.-M. Cheng, H.T. Kung and K.-S. Tan, Use of spectral analysis in defense against DoS attacks, in: *IEEE GLOBECOM 2002*, 2002, pp. 2143–2148.

[9] A. Dainotti, A. Pescapé and G. Ventre, Wavelet-based detection of DoS attacks, in: *2006 IEEE GLOBECOM Conference*, Network Security Systems Symposium, San Francisco, CA, USA, 2006, pp. 1–6.

[10] L. Feinstein, D. Schnackenberg, R. Balupari and D. Kindred, Statistical approaches to DDoS attack detection and response, in: *DARPA Information Survivability Conference and Exposition*, Washington, DC, USA, April 2003, Vol. 1, 2003, pp. 303–314.

[11] Y. Gu, A. McCallum and D. Towsley, Detecting anomalies in network traffic using maximum entropy estimation, in: *IMC 2005*, Berkeley, CA, USA, 2005.

[12] http://lever.cs.ucla.edu/ddos/traces [Online], 2008.

[13] http://www.grid.unina.it/Traffic [Online], 2008.

[14] http://www-stat.stanford.edu/∼wavelab/ [Online], 2008.

[15] P. Huang, A. Feldmann and W. Willinger, A non-intrusive, wavelet-based approach to detecting network performance problems, in: *ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, USA, November 2001.

[16] A. Hussain, J. Heidemann and C. Papadopoulos, Identification of repeated denial of service attacks, in: *Proceedings of IEEE INFOCOM 2006*, Barcelona, Spain, 2006.

[17] A. Lakhina, M. Crovella and C. Diot, Diagnosing network-wide traffic anomalies, in: *ACM SIGCOMM*, Portland, OR, USA, 2004.

[18] L. Li and G. Lee, DDos attack detection and wavelets, in: *IEEE ICCCN'03*, Dallas, TX, USA, October 2003, pp. 421–427.

[19] R. Lippmann et al., The 1999 DARPA off-line intrusion detection evaluation, *Computer Networks* **34**(4) (2000), 579–595 (data is available at: http://www.ll.mit.edu/IST/ideval/).

[20] A. Magnaghi, T. Hamada and T. Katsuyama, A wavelet-based framework for proactive detection of network misconfigurations, in: *ACM SIGCOMM'04 Workshops*, Portland, OR, USA, 2004.

[21] S. Mallat and W.L. Hwang, Singularity detection and processing with wavelets, *IEEE Transactions on Information Theory* **38**(2) (1992), 617–643.

[22] J. Mirkovic, G. Prier and P. Reiher, Attacking DDoS at the source, in: *ICNP 2002*, Paris, France, November 2002, pp. 312–321.

[23] U. Mitra, J. Heidemann, A. Ortega and C. Papadopoulos, Detecting and identifying malware: a new signal processing goal, *IEEE Signal Processing Magazine* **23**(5) (2006), 107–111.

[24] D. Moore, C. Shannon, D. Brown, G.M. Voelker and S. Savage, Inferring Internet denial-of-service activity, *ACM Transactions on Computer Systems* **24**(2) (2006), 115–139.

[25] E.S. Page, Continuous inspection schemes, *Biometrika* **41** (1954), 100–115.

[26] W. Ren, D.Y. Yeung, H. Jin and M. Yang, Pulsing RoQ DDoS attack and defense scheme in mobile ad hoc networks, *International Journal of Network Security* **4**(2) (2007), 227–234.

[27] A. Scherrer, N. Larrieu, P. Owezarski, P. Borgnat and P. Abry, Non Gaussian and long memory statistical characterization of Internet traffic with anomalies, *IEEE Transaction on Dependable and Secure Computing* **4**(1) (2007), 56–70.

[28] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, S. Zhou, A. Tiwari and H. Yang, Specification based anomaly detection: A new approach for detecting network intrusions, in: *ACM CCS*, Washington, DC, USA, 2002.

[29] V.A. Siris and F. Papagalou, Application of anomaly detection algorithms for detecting SYN flooding attacks, in: *IEEE GLOBECOM 2004*, Dallas, TX, USA, November 2004, pp. 2050–2054.

[30] A.G. Tartakovsky, B.L. Rozovskii, R. Blazek and H. Kim, Detection of intrusions in information systems by sequential change-point methods, *Statistical Methodology* **3**(3) (2006), 252–340.

[31] M. Thottan and C. Ji, Anomaly detection in IP networks, *IEEE Transactions on Signal Processing* **51**(8) (2003), 2191–2204.

[32] G. Vigna and R. Kemmerer, NetSTAT: A network-based intrusion detection system, *Journal of Computer Security* **7**(1) (1999), 37–71.

[33] H. Wang, D. Zhang and K.G. Shin, Detecting syn flooding attacks, in: *Proceedings IEEE INFOCOM*, New York, NY, USA, 2002.

[34] J. Yuan and K. Mills, Monitoring the macroscopic effect of DDos flooding attacks, *IEEE Transactions on Dependable and Secure Computing* **2**(4) (2005), 324–335.