Information Fusion 10 (2009) 312–324

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Processing intrusion detection alert aggregates with time series modeling

Jouni Viinikka [a,*,1], Hervé Debar [a,1], Ludovic Mé [b,1], Anssi Lehikoinen [c,2], Mika Tarvainen [c,2]

[a] France Telecom, BP 6243, FR-14066 Caen Cedex, France
[b] Supélec, BP 81127, FR-35511 Cesson Sévigné Cedex, France
[c] University of Kuopio, P.O. Box 1627, FIN-70211 Kuopio, Finland

ARTICLE INFO

Article history:
Received 19 December 2006
Received in revised form 21 January 2008
Accepted 23 January 2009
Available online 4 February 2009

Keywords:
Network security
Intrusion detection
Alert correlation
Time series modeling
Kalman filtering

ABSTRACT

The main use of intrusion detection systems (IDS) is to detect attacks against information systems and networks. Normal use of the network and its functioning can also be monitored with an IDS. It can be used to control, for example, the use of management and signaling protocols, or the network traffic related to some less critical aspects of system policies. These complementary usages can generate large numbers of alerts, but still, in operational environment, the collection of such data may be mandated by the security policy. Processing this type of alerts presents a different problem than correlating alerts directly related to attacks or filtering incorrectly issued alerts.

We aggregate individual alerts to alert flows, and then process the flows instead of individual alerts for two reasons. First, this is necessary to cope with the large quantity of alerts – a common problem among all alert correlation approaches. Second, individual alert's relevancy is often indeterminable, but irrelevant alerts and interesting phenomena can be identified at the flow level. This is the particularity of the alerts created by the complementary uses of IDSes.

Flows consisting of alerts related to normal system behavior can contain strong regularities. We propose to model these regularities using non-stationary autoregressive models. Once modeled, the regularities can be filtered out to relieve the security operator from manual analysis of true, but low impact alerts. We present experimental results using these models to process voluminous alert flows from an operational network.

1. Introduction

Originally intrusion detection systems were designed to detect violations of the monitored system's security policy. Today, complementary uses are more and more common. In some environments the security policy mandates the tracking of all Simple Network Management Protocol (SNMP) and Internet Control Message Protocol (ICMP) traffic and intrusion detection systems (IDSes) typically have generic signatures reacting to the mere occurrence of these protocols. The rationale for such monitoring using today's IDS, with all its limitations (e.g. [1,2]), is in being able to detect large scale problems and anomalies which might manifest themselves indirectly through management and control protocols. The purpose is not in finding attacks contained in one packet nor complex attack scenarios executed by a skillful attacker. Detecting such attacks needs more specific signatures and, depending on the type of the attack, different correlation approaches such as [3,4]. Another motivation can be monitoring the compliance to less critical aspects of system policies, such as instant messaging (IM) usage. We focus on the particularities of the alert flow analysis in the Section 1 and the reader is referred to [5] for a more generic introduction to intrusion detection and alert correlation.

1.1. Alert flow analysis

The complementary uses of IDSes tend to create large numbers of alerts, but those alerts are often of low impact. By low impact we mean that the alert is correctly issued, but does not need immediate reaction from the security operator. Snort [6] *should* trigger an alert `SNMP public access udp` in the presence of SNMP packet using word "public" as the community string. It *is* very likely that the IDS issued this alert on such a network packet. In other words it is very likely that the alert is issued correctly.

Moreover, it can be impossible to determine the significance of a single alert. Instead, the significance depends from the number of similar alerts in recent history with respect to past behavior. We illustrate these two issues with two examples, one with ICMP messages, and another with known, identified malicious traffic.

* Corresponding author. Tel.: +33 2 31 75 97 31; fax: +33 2 31 37 83 43.
E-mail address: jouni.viinikka@orange-ftgroup.com (J. Viinikka).

[1] Supported by the French National Research Agency (Agence Nationale de la Recherche) through the project ACES.
[2] Supported by the Academy of Finland through the Centre of Excellence in Inverse Problems Research programme.

1566-2535/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.inffus.2009.01.003

ICMP messages are part of normal system functioning[3] but they can also be used for information gathering purposes or reflect various network problems. In large networks the security operator does not always know exactly which hosts emit these packets, when and in what quantities. This problem is even more pronounced for a managed security service provider who is processing alerts issued by IDSes in clients' networks. In addition, when the packet contents in normal and abnormal use can be identical, the IDS cannot make the difference between normal and anomalous situation. However, the changes in overall volume of such packets can reveal phenomena worth further investigation.

In the second example, please note that the alerts are not inherently related to complementary sensor usages, but become low impact alerts in this specific situation. A service provider was hosting physical servers for their clients and providing Internet access to those servers. One of the hosted machines was compromised and a distributed denial of service (DDoS) attack tool `Stacheldraht`[4] was installed on the machine. The service provider did not have administrative access to the machine and for contractual reasons the machine had to be maintained connected to Internet. The decision was to block traffic known to be generated by the attack tool at the firewall to monitor the situation and report any changes to the client. The DDoS attack tool communicated with its master, sending a periodic flow of messages, and Snort generated correspondingly a periodic flow of alerts.

After the initial identification of the problem, the alerts were of low impact: they were correctly issued but did not require immediate action from the operator. The aggregated flow revealed (1) the overall regularity and (2) the changes in the behavior of the attack tool, even though the significance of individual alerts was unclear.

Fig. 1 depicts the number of alerts per hour as the function of time over the course of 45 days. The attack tool triggered alerts at rather steady rate around 550 alerts per hour with small periodic variations. The phenomena worth further investigation and reporting to the client were the *drops* in the alert rate i.e. missing alerts, marked with arrows and numbers in the figure. To detect such changes in the rate of low impact alerts, one needs to analyze the aggregated flow of alerts instead of individual alerts.

We have previously proposed to analyze the sequence of low impact alerts meeting given *aggregation criteria* instead of individual alerts [7,8]. The aggregation criterion can be defined using alert attribute values, such as alert name and/or source or destination address. We call the sequence of alerts *alert flow*. We have found significant regularities related to normal system behavior in alert flows aggregated according to the generating signature and IDS[5], and Fig. 1 is one example of such regularities. We model these regularities to both (1) filter out the alerts related to normal system behavior, and (2) highlight interesting phenomena in the alert flows.

We use time series models for modeling, and analyze the series of *alert intensity* observations i.e. the number of alerts in a sampling interval as a time series.

The underlying assumption is that regular flow behavior is related to normal system use, or at least that only changes in any regular behavior are interesting. Thus we aim to detect any abrupt change in the *flow behavior* i.e. the series of alert intensity observations. The basic idea is that once the modeling is done, the output of the model corresponds to normal system behavior, and the difference between observed behavior and model output gives us the anomalous component of the alert flow.

[3] For example, see http://software.sonicwall.com/applications/ips/index.asp?ev=sig&sigid=368.

[4] http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.

[5] See [7,8,5] for more detailed analysis of such regularities encountered in alert data.

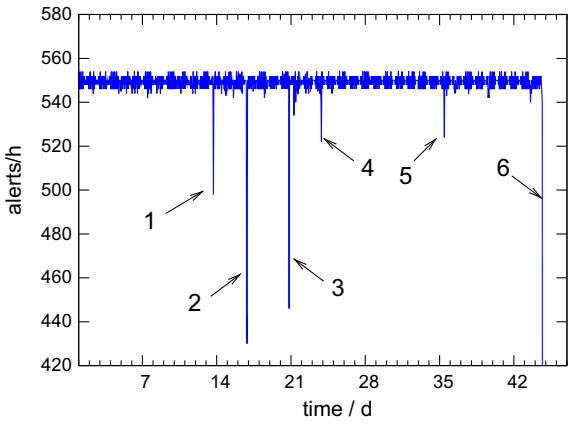

Fig. 1. Flow of alerts triggered by `Stacheldraht` attack tool and known anomalies.

Naturally the underlying assumption does not hold everywhere and we are well aware of that. We have, however, encountered such situations in practice, and thus are trying to provide tools for the operator to help him in such a task.

Please note also that these anomalies could be as well detected by manual inspection of the flow intensity graphs and our aim is to automate this detection. There exists other techniques allowing the detection of anomalies invisible by observing simple measures like flow intensity and variance [9]. Unless the significance of anomalies signaled by such techniques is not well known beforehand, however, it is very difficult for the operator to verify the existence and the significance of the reported anomaly. This is one reason why we focus on automating the detection of visible anomalies.

This alert processing approach is meant as only one component of a larger alert management and correlation platform, with complementary approaches for detecting different types of correlations (e.g. an implementation of [3] for detecting explicitly described attack scenarios).

1.2. Previous approaches and their shortcomings

In [7] we have used an exponentially weighted moving average (EWMA) model to capture normal behavior in alert flows. The EWMA model, even though it worked in some cases, was a very rough projection of the reality. The idea was that short term trends in flow behavior would capture the normal behavior, and abrupt changes would be signaled as anomalies since they would not be predicted by the model.

The approach in [8] was based on the idea that in addition to linear trends, periodic and stationary flow components are caused by normal behavior. We used stationary autoregressive (AR) models for modeling. Compared to the EWMA model we had a more detailed view of the normal behavior instead of relying on short term moving average to capture all regular behavior. However, for the reasons described here below, we have abandoned the stationary AR models, while the EWMA models are still being used. The major inconveniences of stationary AR models were that (1) we needed to remove the trend and periodic components from the flow, (2) the model for remaining normal behavior was stationary i.e. was estimated only once, and (3) the estimation phase required specific training data before entering in the operational phase.

The problem with component removal was, besides the loss of information, the risk of introducing artifacts into the flow. In addition the result interpretation was more difficult since the analyzed signal was very different from the original. Furthermore, we estimated the period of the periodic component using a simple

heuristic based on alert data analyses. The heuristic allowed to decide automatically if the transformation to remove the periodic component is applied, but it was not perfect and again increased the risk of artifacts.

A stationary model was unable to adapt to changes in normal system behavior and had to be re-estimated if the normal system behavior evolved. This introduced an additional phase to the processing system. Moreover, unwanted properties in the training data affected the processing until the next re-estimation.

If the model parameters took into account weekly behavior rhythms, one needed at least one week's worth of data in the estimation phase. In addition, if the periodicity removal transformation was used, even more data were needed. All this delayed the entry to the operational phase.

Lastly, the model errors were still rather large indicating either problems in underlying assumptions, models or in model estimation methods.

In this paper the basic idea is still the same: we first model the normal flow behavior, and then analyze the difference between the observed flow behavior and the model output to detect the deviations from the normal profile. To overcome the above mentioned problems with stationary AR models like the non-adaptivity and the need for re-estimation step, we use non-stationary AR models. In combination with adaptive estimation algorithm we can model the alert flow directly without removing any flow components. At the same time there is no need for training data or explicit model re-estimation steps. Compared to the EWMA model, we gain especially in model accuracy. The accuracy also results in a more stable performance in terms of alert filtering and anomaly detection. The downsides of the approach are more complex algorithms and the risk of incorporating anomalous behavior into the models because of the adaptive estimation algorithm.

The rest of the paper is organized as follows. We start by describing related work in Section 2. Section 3 describes the methods we use in modeling and estimation. More specifically, we define the non-stationary AR models and describe a recursive algorithm called *Kalman filter* we use for estimating time-dependent AR coefficients. Section 4 presents the data used in the experimentation, and the experimental setting in general. Section 5 discusses the results, and we summarize and conclude the paper in Section 6.

2. Related work

In this section we review related work using similar methods in the fields of intrusion detection and network monitoring. We saw already in the Section 1 our previous approaches for filtering irrelevant alerts. In this section we focus essentially on similar approaches from the method point of view. We begin, however, by situating this work with respect to other alert correlation work.

2.1. Relate work in alert correlation

Researchers have proposed various alert correlation approaches for alert processing. However, roughly generalizing, most of these approaches aim to (1) eliminate false alerts (e.g. [10–13]), (2) adjust the alert priority using additional information, such as network topology and inventory data (e.g. [14,15]), and/or (3) find and group alerts to attack or false positive scenarios (e.g. [3,4,16–21]). In other words, existing techniques are designed to process either false or high priority alerts. In the case (1) all the alerts are of no interest for the user, in the case (2) all the alerts are potentially related to attacks and thus potentially of high priority. In the case (3) we have both types of alerts.

In our case, we process low impact alerts (that were correctly issued but of low priority) and the processing is based on the behavior of the alert aggregate and not individual alert's attribute values and eventual additional information. We consider our approach as complementary to these other alert processing techniques.

2.2. Related work from method point of view

Zou et al. detect worms by monitoring scans leaving and entering a network and by looking for exponential growth in the number of scans [22]. They use several worm propagation models, one of which is an autoregressive model. They use Kalman filter for estimating the parameters of all models. On the model side, the differences are that (1) our model is of significantly higher degree, and (2) it is non-stationary. On the estimation side the differences are that (1) Zou et al. estimate a stationary parameter, and (2) their goal is to have a good estimate as early as possible where as we need constantly new estimates of non-stationary parameters.

In the field of network monitoring, Soule et al. [23] use Kalman filtering and state space modeling to estimate the origin–destination flows from link-level measurements like byte counts provided by SNMP. Modeled origin–destination (OD) flows are high level aggregates in Sprint's European backbone. In addition to the application domain, the main difference is that Soule et al. use the known good values of OD flow measurements to calibrate the model when model error increases too much. This allows them to assume the stationarity of some model components. In our case the known good values of model parameters are unavailable.

Kalman filtering has not been used widely in intrusion detection, as far as we know. Bayesian filtering is a more general filtering algorithm than Kalman filtering, and used by Hall et al. [24] to improve MAC address spoofing detection in wireless networks. The detection is based on radio frequency fingerprinting and associating a fingerprint with MAC address. Bayesian filtering is used to reduce the uncertainty of detection by using several observations before the final decision whether the MAC address is considered as spoofed or not.

Bayesian inference or Bayesian networks are used in alert correlation for example by Valdes and Skinner in [25], Goldman et al. in [26], and Qin and Lee in [21]. Apart from the name and some underlying Bayesian principles, the methods are different.

Spectral analysis is used for detecting, classifying, and analyzing distributed denial of service (DDoS) attacks in [27–29]. Compared to our work, the analysis/detection philosophy is different in these approaches. These methods are based on the frequency domain representation of the signal. In addition, at least parts of the signal are assumed stationary, whereas in our case the series are non-stationary. The methods are also focused on DDoS attack detection and analysis, whereas we are interested in anomalies in general.

In [30] Blazek et al. formulate the detection of abrupt changes caused by an DoS attack in the packet size distributions as change-point detection problem. They propose two detection algorithms, one sequential and another batch-sequential that minimize the detection delay for a given false alert threshold. In [31] Tartakovsky et al. further develop the approach. Our work differs in methods and uses a higher level data source, but the goals are, however, very similar: detecting abrupt changes. In [30,31] the algorithms need information on the network traffic distributions, at least for the normal situation, and as mentioned in [31] that information may change dramatically at larger scales like night, day, morning, and afternoon. We are using a non-stationary model especially to cope with such variations. On the other hand, we are not able to provide such analysis and bounds on false alert rates and detection delays as are provided in [30,31].

In [32] the authors propose an adaptive threshold algorithm and a CUSUM-type algorithm for detecting SYN flooding attacks. The adaptive threshold algorithm is very similar to the one we proposed in [7] and the CUSUM-type algorithm uses a EWMA procedure to smooth out the weekly and daily variations in the observed statistics. We have experienced problems with our EWMA-based approach especially with daily and weekly variations and our current work is aiming to overcome these limitations. See [5] for more detailed discussion on the limitations of the EWMA approach.

Barford et al. [33] analyze network traffic using wavelets. They use flow, packet and byte intensities from both SNMP and NetFlow data to form the signal. This signal is analyzed to detect anomalies in mid and high frequency components. The wavelet analysis has the advantage of being a multi-scale technique. At the same time, the detection algorithm works only in batch-mode, where as our approach can be implemented as on-line algorithm.

Dainotti et al. propose an interesting system to detect volume-based anomalies in network traffic in [34]. They use a two-stage system, where the detection algorithms from [32] are used for rough detection and then a component based on continuous wavelet transform (CWT) for finer analysis of the alerts from the first stage. The CWT component allows to localize the anomaly both in time and scale. Our approach lacks the multi-scale property of their approach and is thus more limited in terms of detection capabilities. We can, however, operate on-line whereas the proposed wavelet-based approach works off-line.

To summarize, the existing work in alert correlation is designed for different objective: either to eliminate false alerts or to regroup, verify and re-prioritize alerts related to attacks. We need an approach to process the alerts as an aggregate and taking the time-related flow behavior into account. On the method side, AR models and Kalman filtering have been used in the fields of network security and network traffic analysis. Overall, in those approaches, the modeled phenomena are different, the AR models stationary and of lower degree, and Kalman filter is used in for different estimation tasks. In addition, Kalman smoothers have not been used, to our knowledge, for similar purposes.

3. Non-stationary AR modeling

According to our idea, our model should capture normal flow behavior. Therefore, at instant t, the difference between observations y_t and model output $\hat{y}_t$ represents the abnormal component of the flow $\tilde{y}_t$. The abnormal component is called also the residual series or the model error, and is defined as $\tilde{y}_t = y_t - \hat{y}_t$. In this section we will describe how the modeling is done, and how the anomalies are detected from the abnormal component. The key elements of this process are:

- Normal flow behavior is modeled as a weighted sum of previous observations.
- The weights are re-estimated or updated at every new observation.
- The estimation is done with statistically optimal recursive algorithms.
- Thanks to the update step and the algorithms used, we can analyze alert flows without removing any flow components.
- Signaled anomalies are the most significant differences between forecasts provided by the model and the observations.

An alert flow is a stream of successive alerts meeting the aggregation criteria, which are the generating signature and IDS. Flow intensity measurement taken at fixed, discrete time intervals form a time series $\{y_t\}$. In Section 3.1 we describe the non-stationary autoregressive model we use to capture normal flow behavior. The parameters of the model are time dependent and can be estimated with recursive algorithms such as Kalman filter.

Kalman filter is an adaptive and recursive data processing algorithm that is suited for on-line estimation. We actually use a variation of Kalman filter called *Kalman fixed-lag smoother*, both described in Section 3.2. It provides better estimates than Kalman filter at the cost of a small additional delay.

As neither the underlying idea nor the model are perfect, the abnormal flow component contains some noise in addition to anomalies. Thus we signal only the most significant elements of the residual series as anomalies. The anomaly detection algorithm is described in Section 3.3.

3.1. Non-stationary AR model

The non-stationary AR(p) model of degree p is defined as:

$$y_t = \sum_{k=1}^{p} a_t^k y_{t-k} + e_t. \tag{1}$$

Here y_t is the observation at instant t, a_t^k are the time-varying model parameters, and e_t is observation error. In other words, the model uses a weighted sum of p previous values to estimate the current observation value, the weights being the AR coefficients a_t^k, $k = 1 \ldots p$.

The non-stationarity means that the weights a_t^k are time dependent. The idea is that normal system usage causes sufficiently regular and smooth flow, that the current value can be predicted as a linear combination of p past values. The part of the flow behavior that cannot be predicted in this fashion is sufficiently anomalous to be reported to the user. By using time-varying AR models, we allow model of normal behavior to adapt to changes of the monitored system.

The model degree p needs to be defined. However, its value depends strongly on the application. Thus we will discuss it in the Section 4.3 with the experimental setting.

It could be argued that AR model is too limited for modeling the type of flows we have in our data. For example, errors on rapid changes in observations are one manifestation (cf. Section 5) of the AR model's limitations and non-linear models, for example, could provide better reactivity to rapid changes[6]. We have three reasons to stay with linear models: (1) Given our underlying idea we do not even want to model the alert flow exactly, especially the rapid changes should be left out of the model. (2) This limitation of AR models limits the risk of modeling anomalous behavior into to the model of normal behavior. This is especially the risk with dynamic parameter estimation, and one fundamental problem of any anomaly detection algorithm using some sort of learning. (3) Simple models mean simpler algorithms and this translates to faster and lighter implementations, which is an important consideration for deploying the method.

3.2. Kalman filtering

The model parameters a_t^k in (1) need to be estimated. In estimation theory the term *filtering* is used for real-time data processing method providing parameter estimates from observations. Estimates are obtained using (1) all past observations up to the current instant, and (2) an evolution model describing the time-varying characteristics of the parameters to be estimated. Filtering can be seen as a process where we update incrementally our knowledge of the system as a new observations arrive. In other words, instead

[6] Personal communication, David Mercier, Laboratoire Electronique et Traitement du Signal, Commisariat à l'Energie Atomique, 2006-02-14.

of starting the estimation from scratch for all t, we are just updating the previous estimate each time a new observation is made. The update step is computationally lighter because the algorithms take advantage of certain properties of models and data. Thanks to filtering, it is computationally feasible to use non-stationary models.

To describe the Kalman filtering process, we use a *state space formalism*[7]. In our case, we estimate the AR model parameters from *observed* alert series $\{y_t\}$. The true parameters cannot be observed directly and in state space terminology they are called the *state* θ. Now assume that we have an *observation model* giving the relation between the unobservable state and the observations, and an *evolution model* describing the time-varying nature of the state. Without prior information, the evolution of the state is often described with a random walk model [36, p. 54], and we have [36]

$$y_t = H_t\theta_t + e_t, \tag{2}$$
$$\theta_{t+1} = \theta_t + w_t, \tag{3}$$

where e_t and w_t are observation and state noises. Eqs. (2) and (3) are called, respectively, *observation* and *state equations*.

The non-stationary AR model of (1) can be put in vector form. With time-varying coefficients as

$$\theta_t = \left(a_t^1 \dots a_t^p\right)^T \tag{4}$$

and the past observations as

$$H_t = (y_{t-1} \dots y_{t-p}) \tag{5}$$

we have the non-stationary AR model of (1) as

$$y_t = H_t\theta_t + e_t \tag{6}$$

which corresponds to observation Eq. (2).

For representing the Kalman filter equations, we use $\hat{\theta}$ to denote estimate of θ and $\tilde{\theta}_{t|t-1}$ for estimation error of the state at instant t using observations accumulated at instant $t-1$. In addition, we denote covariance matrices with C. Now the Kalman filter equations[8] for estimating time-dependent AR parameters are

$$C_{\tilde{\theta}_{t|t-1}} = C_{\tilde{\theta}_{t-1}} + C_{w_{t-1}}, \tag{7}$$
$$K_t = C_{\tilde{\theta}_{t|t-1}} H_t^T \left(H_t C_{\tilde{\theta}_{t|t-1}} H_t^T + C_{e_t}\right)^{-1}, \tag{8}$$
$$\epsilon_t = y_t - H_t\hat{\theta}_{t-1}, \tag{9}$$
$$\hat{\theta}_t = \hat{\theta}_{t-1} + K_t\epsilon_t, \tag{10}$$
$$C_{\tilde{\theta}_t} = (I - K_tH_t)C_{\tilde{\theta}_{t|t-1}}. \tag{11}$$

The one step prediction error ϵ_t of observation y_t is used to estimate the unknown and unmeasurable observation noise e_t.

Using Kalman filter in practice requires initial values for state θ_0, error covariance $C_{\tilde{\theta}_0}$, state noise covariance C_w and observation noise covariance C_e. A common approach is to set $\theta_0 = 0$ and $C_{\tilde{\theta}_0} = I$ and run the algorithm on a short segment from observation data backwards. The values obtained in this way for θ and $C_{\tilde{\theta}_0}$ are then used to initialize these values in the actual processing run. Even though we used this initialization step, it is not necessary unless there are events of interest early in the data. For the record, we used this approach with $200 + p$ data points.

The adaptation speed of the Kalman filter is determined by the state noise covariance C_{w_t}. In other words it controls how fast the state adopts the changes in observations. In [36, p. 55] $C_{w_t} = \sigma_w^2 I$ and $C_{e_t} = \sigma_e^2 = 1$ (σ_w^2 is the state noise covariance coefficient and σ_e^2 the observation noise coefficient) were reported as favorable, and were used also here. This means that there is a single coefficient, σ_w^2, to adjust the adaptation. The adaptation speed increases with σ_w^2 and the variance of state estimates is inversely proportional to the value of σ_w^2. Therefore it should be chosen for a desired balance between state estimate variance and filter adaptation speed according to the application. We will discuss the choice in Section 4.3 along with the experimental setting.

The use of future observations for state estimation is called *smoothing*. The objective is to improve the quality of estimates provided by Kalman filter. When we use observations $y_1, \dots, y_{t+L}$ to estimate state θ_t with $L > 0$ the quality of the estimate $\hat{\theta}_t$ is likely to improve compared to estimate obtained with Kalman filtering. There are three classical smoothing algorithms, *fixed-point smoother*, *fixed-interval smoother*, and *fixed-lag smoother*. We use fixed-lag smoother, since it is suitable for on-line processing when a small, fixed delay of L observations is allowed. In other words, the fixed-lag smoother provides estimates $\hat{\theta}_{t|t+L}$ for all time instants $t = 1, \dots, N$ with a fixed lag of L we need to wait for future data points.

To estimate the state θ_t at instant t with a fixed-lag smoother, we will wait to have observations up to instant $t + L$, where $L > 0$. The state and observation equations have now *extended variables*. The Kalman filter equations remain the same, but the extended state and observations variables are

$$\theta_t^* = \begin{bmatrix} \theta_t \\ \theta_{t-1} \\ \vdots \\ \theta_{t-L} \end{bmatrix}, \tag{12}$$

$$H_t^* = [H_t \quad 0 \quad \dots \quad 0], \tag{13}$$

and the simplified state Eq. (3) becomes

$$\begin{bmatrix} \theta_{t+1} \\ \theta_t \\ \theta_{t-1} \\ \vdots \\ \theta_{t-(L-1)} \end{bmatrix} = \begin{bmatrix} \theta_t \\ \theta_{t-1} \\ \theta_{t-2} \\ \vdots \\ \theta_{t-L} \end{bmatrix} + \begin{bmatrix} w_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{14}$$

$$\theta_{t+1}^* = \theta_t^* + w_t^*.$$

The observation Eq. (2) can be written as

$$y_t = [H_t \quad 0 \quad \dots \quad 0] \begin{bmatrix} \theta_t \\ \theta_{t-1} \\ \vdots \\ \theta_{t-L} \end{bmatrix} + e_t, \tag{15}$$

$$y_t = H_t^*\theta_t^* + e_t.$$

3.3. Anomaly detection

We detect anomalies by analyzing the abnormal component. The detection algorithm is an adaptation of EWMA control chart procedure. It was presented in [8], so we provide only a brief overview here.

The EWMA of model error is $\mu_{\tilde{y}_t} = (1-\lambda)\mu_{\tilde{y}_{t-1}} + \lambda\tilde{y}_t$, where $0 < (1-\lambda) < 1$ is called the *smoothing factor*. To obtain the standard deviation, we maintain also the following statistic $\mu_{\tilde{y}_t}^2 = (1-\lambda)\mu_{\tilde{y}_{t-1}}^2 + \lambda\tilde{y}_t^2$. The standard deviation is obtained as $\sigma_{\tilde{y}_t} = \sqrt{\mu_{\tilde{y}_t}^2 - (\mu_{\tilde{y}_t})^2}$.

We signal an anomaly when the current residual value $\tilde{y}_t$ surpasses the control limits

$$\hat{\mu}_{\tilde{y}_{t-1}} \pm n\hat{\sigma}_{\tilde{y}_{t-1}}, \tag{16}$$

[7] Please note that the representation is tailored for estimating AR model parameters. For more general representation, see e.g. [35–37].

[8] Please note that these are not the general Kalman filter equations, since the use of the random walk model simplifies the situation, see [36] for more details.

where $n \in \mathbb{R}_+$ adjusts the width of the control limits. We will discuss the choice of n in the Section 4.3 along with the experimental setting.

4. Data and experimental setting

This work has been inspired by alert data from operational information systems. In this section we present the data used in experiments, how it was collected and its characteristics. We will finish this section by presenting the experimental setting used for modeling and filtering the alert data.

Please keep in mind that given the subset of alerts we are analyzing, the anomalies are not necessarily attacks, but more likely more general problems. We aim by selectively filtering out alert noise to free time and resources for both operators and other correlation approaches to focus on those more relevant alerts. Thus the filtering is considered as more important objective than the anomaly detection.

The functioning of this type of filtering can only be tested with real data containing alert noise generated by the normal system functioning. Lack of good testing data is a common problem in intrusion detection community since obtaining the ground truth about the observed events in a non-trivial real system is very difficult, especially over a longer period. To address this problem, there have been efforts to generate data sets and the ground truth using artificial environments. The Lincoln Laboratory work being probably the best-known [38,39]. However, the quality of those data sets has been criticized [40] and nowadays they are widely considered as unsuitable.

4.1. Data collection

Alerts were generated by a Snort IDS using the default set of signatures and few additional rules from Bleeding Snort[9]. The IDS was placed in one of the computing centers of a division of a large organization. The computing center provided mostly services for local users, but was accessible also from other sites of the division, and from other divisions of the organization. The services hosted in the computing center included backup applications, network disks, and collaboration software. The users of the information system were mostly normal office workers.

The IDS had generated approximately 36 M alerts in the course of 43 days. Alerts were stored in a database and analyzed in post mortem. Table 1 shows the most prolific signatures in the data set.

The flows were obtained by aggregating all alerts generated by the same signature and the same IDS as an alert flow. In other words, these are high level flows containing alert from all different hosts triggering the given signature. The flow granularity is a compromise between the precision with which the detected anomalies can be attributed to a cause and the number of flows that need to be analyzed. In addition, in order to find out the system level behaviors, a high level aggregation is often needed. This is a choice we have made in our case, and we can also use finer grained flows according to the needs of the operator.

4.2. Data characteristics

This data set supports the arguments stated in the Section 1, as well as did the alerts analyzed in [7,8]. First, from Table 1 we can see that a large proportion of alerts are low impact alerts, triggered by SNMP, ICMP and NETBIOS related signatures.

Table 1
Signatures having generated over 1 M alerts. Five first ones count for 69% and the whole table for 78% of all alerts. Most significant noise sources are signatures alerting on administration traffic (SNMP, ICMP). Signatures triggering on system usage (NETBIOS) are the second most important source.

Signature name	Alerts
SNMP request udp	8 065 524
SNMP public access udp	6 895 768
ICMP L3retriever Ping	6 270 314
(http_inspect) BARE BYTE UNICODE ENCODING	2 310 954
NETBIOS SMB-DS DCERPC NTLMSSP asn1 oflow att	1 936 139
NETBIOS SMB-DS IPC$ share unicode access	1 901 695
NETBIOS SMB IPC$ share unicode access	1 362 219
Sum	28 742 613
All alerts	36 862 451

We will use the flow generated by the signature `ICMP L3Retriever Ping`[10] (later `L3R`) for more detailed comparisons between non-stationary and EWMA models in Section 5, so we describe it in more detail. The signature alerts on ICMP echo packets containing a string `ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI` in the payload. These packets known to be used by L3Retriever scanner. However, Windows Domain Controllers as well as Windows 2000 and Windows XP machine generate ICMP echo messages with the same string in the payload[11].

In our case a large majority of the alerts are generated by Windows machines. Strictly speaking, these alert could be considered as false positives since the ICMP echo messages are not generated by the L3Retriever scanner as described by the alert. However, since these alerts are issued on the ICMP messages and they reflect the normal behavior of the system, the aggregate potentially carries useful information.

Secondly, alert flows contain significant regularities. Fig. 2 depicts the first 13 k observations of the flow generated by the signature `L3R`. The alert intensity is plotted at 1 min sampling interval and we can see clear daily and weekly rhythms. The figure shows flow intensity starting on a Friday where we can see an increase in the activity during the day zero. During the following weekend the activity stays at lower level. On the working days, starting from day three, the alert intensity increases during the working hours, and decreases during the night time. The regular behavior is caused by normal functioning of Windows hosts on the network, and thus the alerts are not interesting for the operator.

The flow contains several abrupt deviations from the normal behavior. We have marked these deviations with arrows and numbers in the Fig. 2. We do not know the exact nature of the phenomena behind these deviations, but they should be signaled to the operator for further investigation.

4.3. Experimental setting

Next we will describe the experimental setting. More specifically, we will discuss the effect and choice of sampling interval, the preprocessing of the analyzed series with low-pass filtering, and the used model parameters. The tool used in experimentation was written in Matlab.

4.3.1. Sampling interval

The choice of sampling interval t_s has an effect on (1) the timeliness of the detection, (2) the real-time-reach of the AR models, and (3) the visibility and clarity of phenomena.

Forgetting all processing delays, anomaly detection will be delayed at least to the end of the current interval. In the case of

[9] http://www.bleedingsnort.com.

[10] http://www.snort.org/pub-bin/sigs.cgi?sid=466.

[11] http://software.sonicwall.com/applications/ips/index.asp?ev=sig&sigid=368.

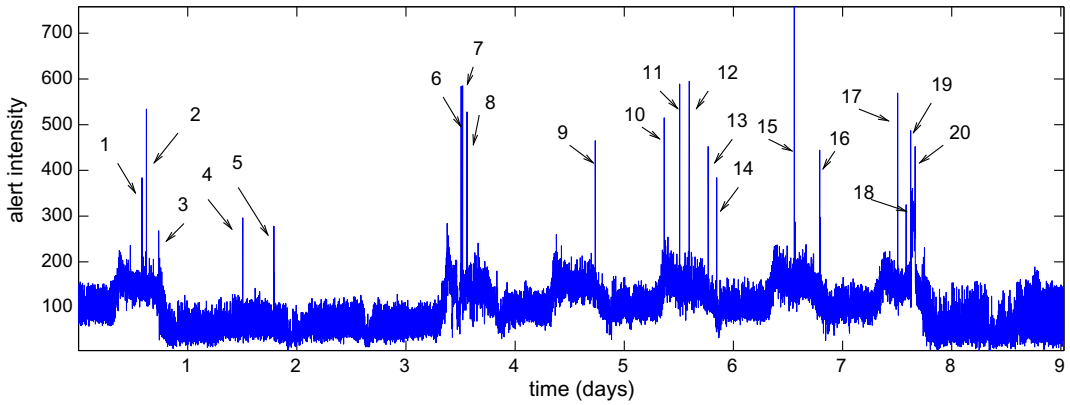

Fig. 2. Observations and known anomalies in the flow `ICMP L3Retriever Ping`.

Kalman fixed-lag smoother, the delay is longer, L sampling intervals. This suggests shorter sampling intervals for shorter detection delays. However, the low impact nature of analyzed alerts means that the timeliness is less important than for some other alert processing methods.

An AR(p) model uses data that is at most $p \cdot t_s$ old. If the product is sufficiently large, the real-time-reach of the model may be sufficiently large to cover the period of some regularities. For example, with $t_s = 60$ min and $p = 24$, the real-time reach corresponding to one day might allow the model to better take into account daily rhythms. However, in our case we are using smaller t_s, and such real-time reach would require high and computationally expensive model degree. Thus we leave this aspect out of consideration.

To give an example of the visibility and clarity of phenomena, Fig. 3 shows the flow `NETBIOS SMB IPC$ share unicode access` at sampling intervals of 1, 5, 10, 15 and 20 min. We can see that (1)

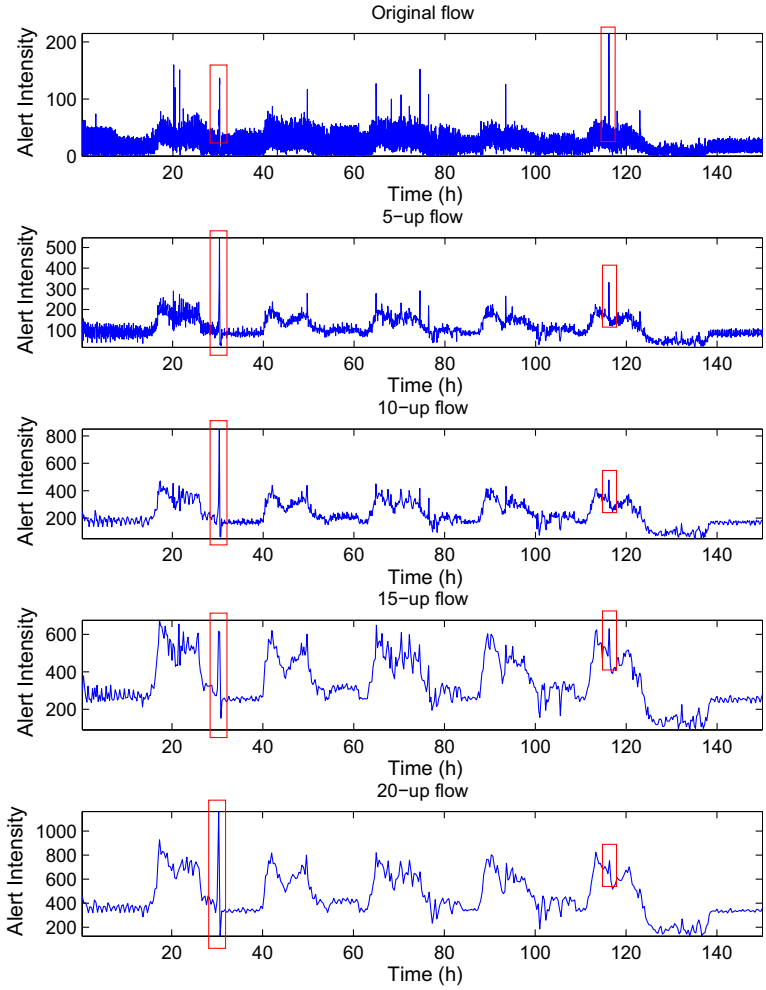

Fig. 3. Effect of sampling interval on visibility of phenomena in the flow `NETBIOS SMB IPC$ share unicode access`. The original flow intensity was measured once a minute, and following flows were obtained by summing up 5, 10, 15 and 20 samples. The normal flow behavior is visible at all time scales, but the visibility of the anomalies can vary from one time scale to another. The left hand rectangle points out how a larger time scale helps detecting an increase in alert intensity. The right hand rectangle highlights how a rapid change is averaged out at larger time scales.

the normal behavior is visible at all time scales, but (2) the visibility of anomalies may vary from one time scale to another. Overall, we have analyzed alert flows with different sampling intervals ranging from 1 min to 1 h. We have found out that increasing t_s results in smoother flows since the high frequency fluctuations are averaged out. The normal behavior was found to be visible at all used sampling intervals, but only anomalies whose time scale is close to sampling interval are clearly visible in the flow. The form of the anomalies, however, was similar throughout the time scales. Thus, the choice of sampling interval plays a larger role in defining the time scale of anomalies that stand out than in the visibility of the normal flow behavior. We chose to use $t_s = 1$ min for following reasons:

- Timely detection of flow anomalies even when using Kalman fixed-lag smoother.
- Flows are more difficult to process than with larger sampling intervals.
- If we can filter out normal behavior with such flows, the method should work also with larger sampling intervals, if the interest is in anomalies of longer time-scale.

4.3.2. Smoothing out high frequency fluctuations

The low intensity and high frequency fluctuations in the signal are considered as "noise". We can reduce the effect of the noise by smoothing the observations with EWMA before the analysis. In this way we do not need to increase the sampling interval to average out the noise. We use a smoothing factor 0.6, and this smoothing corresponds to low-pass filtering the observations.

4.3.3. Model parameters

We wanted to explore the possibility of having a general set of parameters (p, L, n, σ_w^2) that would work for most of the flows. We fixed the parameters by trial and error using a small part of the flow `SNMP public access udp`. We then used the parameters for all flows of Table 1. We verified later that it would be possible to find a better set of parameters for each flow, if abandoning the general approach and working on a flow-by-flow basis. However, from the system deployment point of view, a general set of parameters is more interesting.

According to [36], in practice the model degree is often fixed using some prior knowledge or guidelines. We had identified daily and weekly rhythms, but because of small t_s we could not use large enough p to cover even daily rhythms. In addition, the number of flows may be large, and with small t_s the update steps are frequent. Thus we want to limit the model degree to ease the computational load. We settled to $p = 20$ as a degree which allowed the model to capture sufficiently well the normal flow behavior.

We experimented with different smoother delays L, and noticed a significant increase in model accuracy when switching from Kalman filter to Kalman fixed-lag smoother with $L = 1$. The increase in accuracy was then slower when further increasing L. As larger L means also longer delay in detection, we chose to use $L = 1$.

Similarly, we chose to use $\sigma_w^2 = 0.000025$ for the adaptivity of the filter, and $n = 4$ as parameters allowing us to filter out the normal flow behavior from the flow `SNMP public access udp`.

If we had labeled data for testing and training, or clean normal data for building the model of normal flow behavior, we could do a more thorough analysis of the effect of different algorithm parameters:

The model degree p could be set using some algorithms like AIC or BIC [41, pp. 200–202]. The algorithms try to optimize the goodness of fit vs. model complexity ratio. Thus they need clean data for which a perfect fit would not mean incorporating anomalies in the model, and in our current situation such algorithms are unfeasible.

Control limit width n could be set to some acceptable false positive rate. The processing method could be run on a known normal data set to find value n such that normal variations do not cause more false positives than acceptable.

State noise covariance factor σ_w^2 could be set so that it gives a suitable balance in adapting to normal behavior and avoiding to incorporate anomalous behavior in to the model.

There are also other adaptive algorithms that could be used instead of Kalman filter and fixed-lag smoother. For example, in biosignal applications least mean square (LMS) and recursive least squares (RLS) estimates are popular [36]. We do not claim that Kalman filter/smoother has the best cost/quality ratio, just that it is good enough to show that it is overall possible to model normal alert flow behavior with linear, non-stationary time series models and adaptive algorithms. We also argue that as we lack good testing data it is (1) very difficult, and (2) rather useless to spend time optimizing such factors. The optimal values may vary from one environment to another and depending on the user's needs. The parameters should be fixed in the deployment phase.

5. Results: Applying alert processing to flows

In this section we present the results obtained by applying the non-stationary AR (NAR) method on alert flows. As was mentioned in Section 1.2, we have found our previous EWMA approach [7] to work in some situations, but the stationary AR approach [8] to be impractical. Thus we will be comparing the non-stationary AR approach with EWMA models in 5.1. We will then present less detailed results on alert reduction capability in 5.2, and finish the section by discussion in 5.3.

5.1. Comparing non-stationary AR and EWMA approaches

We will use two criteria to compare the two approaches: (1) the anomalies signaled by the methods and the known anomalies in the analyzed flows, and (2) the model error $\tilde{y}_t = y_t - \hat{y}_t$. The model error is a limited metric since the alert flows contain also anomalies we do not want the model to capture. However, it gives us an idea of the difference in the accuracy of the models, especially given the two facts. First, after the manual inspection of the alert flows, it is reasonable to claim that the proportion of anomalies with respect to normal flow behavior is small. Second, as we shall see, the models signal a majority of the known anomalies i.e. the anomalous behavior has not been modeled. Thus the models have captured mainly the normal behavior, and the model error describes mainly the difference between the model and the normal behavior.

We start with the flow `Stacheldraht` shown in Fig. 1. The Figs. 4 and 5 depict the anomalies signaled by EWMA and NAR approaches, respectively. The black line is the alert intensity observed once in hour over 45 days. The signaled anomalies are marked with dotted vertical lines and diamond markers.

The EWMA model used parameters found suitable for hourly measurements, $(1 - \lambda) = 0.92$ and $n = 3$ in [7]. For the non-stationary AR model we used the parameters described in Section 4.3.3 with smaller detection threshold $(n = 3)$ given the larger sampling interval i.e. smoother signal than considered in Section 4.3.3.

With such a stable flow the two models are almost equal, EWMA even slightly better when judging by the signaled anomalies. Both detect all six known anomalies indicated in Fig. 1, and EWMA model signals additional small drop in alert intensity in

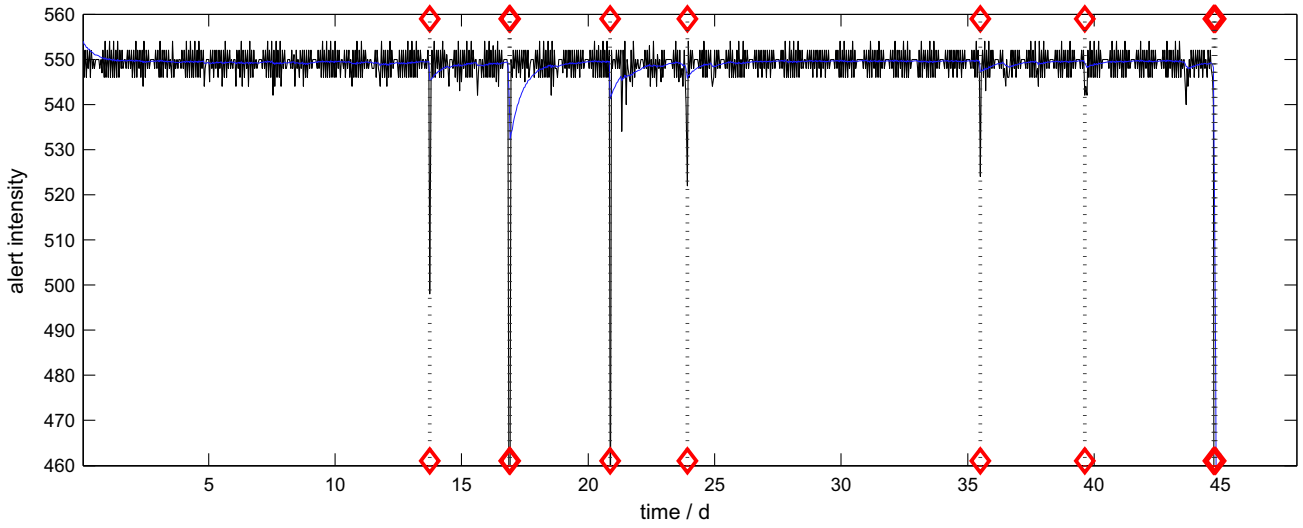

Fig. 4. The anomalies signaled by EWMA model from the flow `Stacheldraht`.

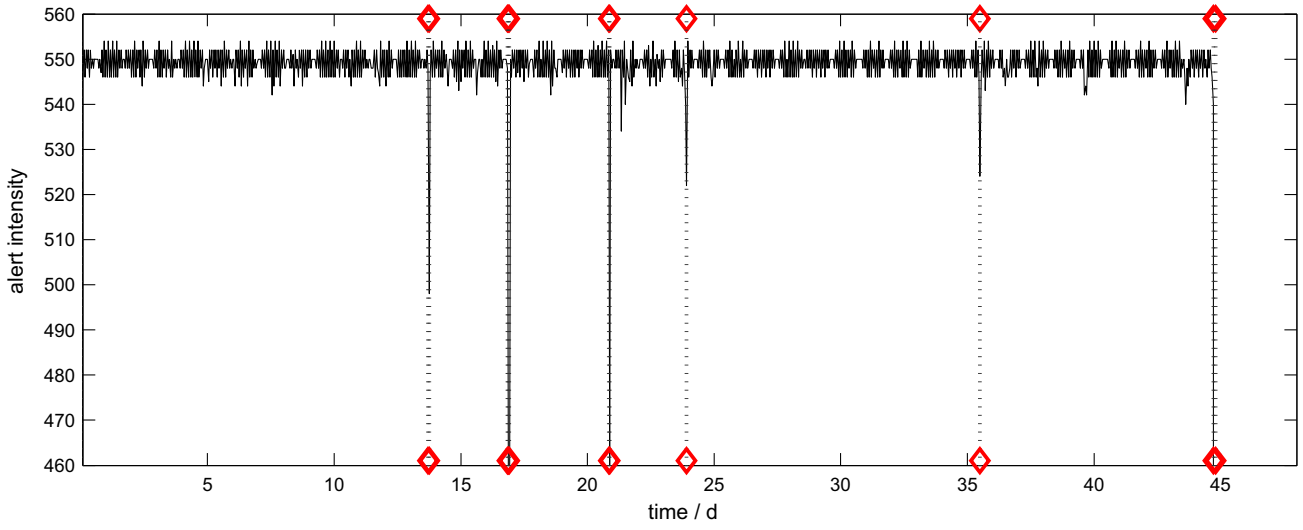

Fig. 5. The anomalies signaled by NAR model from the flow `Stacheldraht`.

the end of day 39, which can be considered as an interesting anomaly.

EWMA model signaled in total 10 anomalies, some of which being issued in successive intervals, resulting to 7 distinct anomalies. The NAR model signaled in total 11 anomalies in 6 groups. Neither method signaled uninteresting anomalies.

The model error, however, shows NAR model being significantly more accurate. The sum of squared model errors for EWMA model was 1.6×10^6 whereas for NAR model only 37. The EWMA models slower adaptation penalizes it in terms of model error because of the sharp drop in the end of the day 44. But even when discarding the error caused by the drop, the remaining error is of the order 10^4.

To give an example of a more complex flow, we analyze the first 13 k observations of the flow `L3R` using both methods. To determine the parameter $(1-\lambda)$ controlling the adaptation speed for EWMA method, we used values $\{0.92, 0.94, 0.96, 0.98, 0.99, 0.995, 0.998\}$. Table 2 shows the sum of squared errors (SS), the sum of absolute errors (SA), and the number of alerts. Based on these values, we chose to use $(1-\lambda) = 0.98$.

Table 2
The model errors, sum of squares (SS) and sum of absolute values (SA), and the number of anomalies issued with different smoothing factors.

$(1-\lambda)$	SS	SA	Alerts
0.92	9.5×10^6	$2.4 \cdot 10^5$	106
0.94	1.0×10^7	$2.5 \cdot 10^5$	96
0.96	1.1×10^7	$2.6 \cdot 10^5$	88
0.98	1.2×10^7	$2.7 \cdot 10^5$	82
0.99	1.4×10^7	$2.9 \cdot 10^5$	82
0.995	1.6×10^7	$3.3 \cdot 10^5$	101
0.998	2.1×10^7	$3.8 \cdot 10^5$	109

This gives an advantage for the EWMA method, since we fitted the parameters for this flow, whereas the NAR method parameters were fixed using another alert flow.

The anomalies of interest were given in Fig. 2. Figs. 6 and 7 depict the analysis results using EWMA and NAR approaches, respectively. The observed alert intensity is drawn a solid black line and the signaled anomalies are marked with vertical dotted lines and diamond markers.

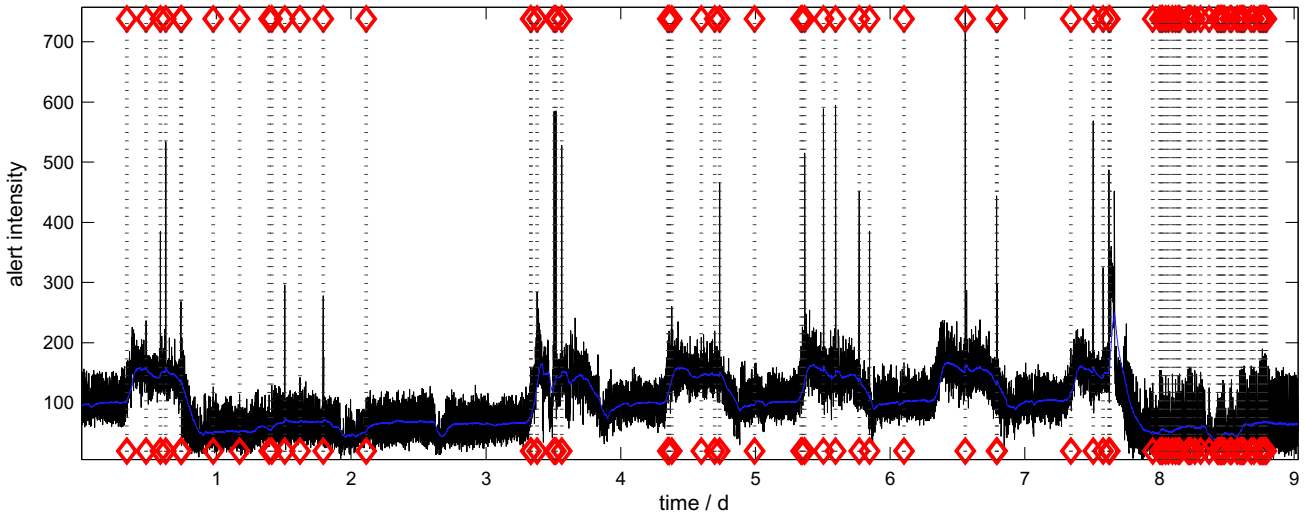

Fig. 6. The anomalies signaled by EWMA model from the flow L3R.

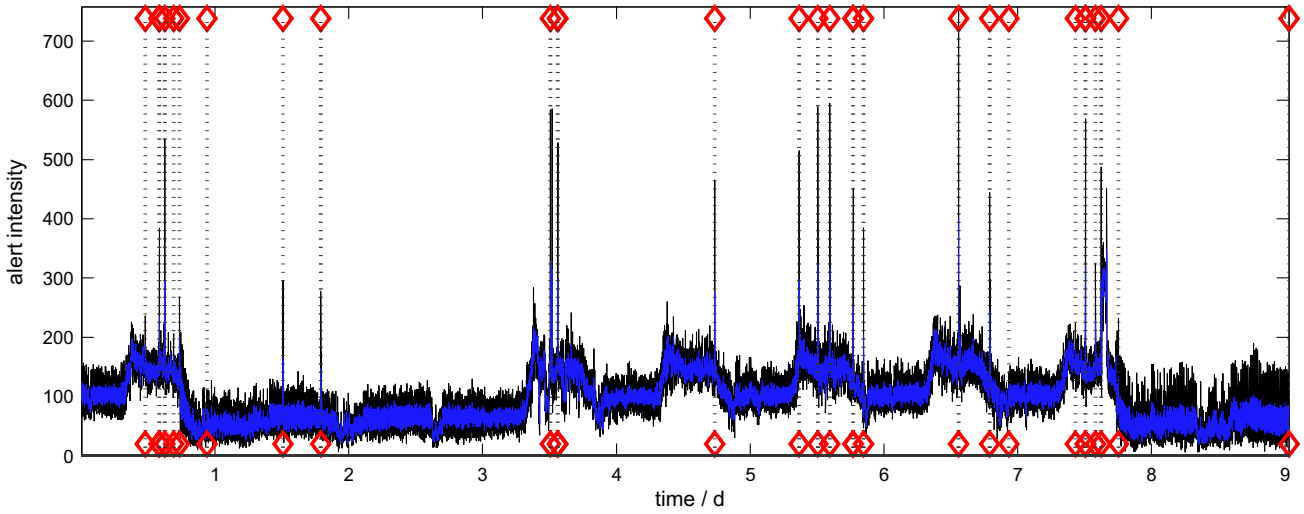

Fig. 7. The anomalies signaled by NAR model from the flow L3R.

For the known anomalies, both approaches perform in similar manner. EWMA approach detected all but #20, and NAR approach all but #7 and #20. The EWMA model signals several anomalies close to #6, so it is impossible to say whether the known anomaly #7 was really detected or only covered by the burst of anomalies issued around #6.

The overall number of signaled anomalies is, however, much higher for the EWMA model, 82 against the 39 for NAR model. In addition, from the Fig. 7 we can see that for some known anomalies NAR model issues two successive anomalies, resulting to 25 groups. This is more convenient for the operator than the dispersed anomalies issued by the EWMA model seen in the Fig. 6. Furthermore, NAR model signals anomalies only on abrupt changes whereas the EWMA model flags also regular behavior, for example during the day 8.

The NAR model is also more accurate than the EWMA model. The sum of squared model errors is 2.7×10^5 for NAR model, and 1.2×10^7.

To conclude the comparison based on these examples, we can say that for simple flows the two models perform in similar manner, and that the NAR model is more suitable for complex flows. The NAR model is more accurate all the time, and the differences can be quite significant. The NAR model behaves in more constant manner than the EWMA model, the former effectively highlighting only abrupt shifts in the flow intensity.

5.2. Alert reduction via filtering

Knowing that the NAR method highlights most abrupt changes i.e. the phenomena of interest in the flow L3R, we can look at the alert reduction obtained through this processing.

The example contains 1 269 529 alerts. After processing, the method signaled 39 intervals as anomalous. The proportion of intervals necessitating operator's attention is reduced to 0.30%. Alerts related to the weekly and daily rhythms present in the flow were filtered out. All abrupt alert bursts are signaled as anomalous, some more than once.

Due to space constraints we are not presenting such detailed analysis for the remaining flows. We will look only at the alert reduction obtained with the other flows created by the signatures from the Table 1. The NAR model behavior is similar with all other flows: the model signals a large majority of abrupt changes. Thus

Table 3
Flow volumes and signaled anomalies.

Flow	Observ (10^3)	Alerts (10^6)	An	Prop (10^{-3})	OK
SNMP request	13	3.0	362	28	NOK
SNMP public	13	2.8	16	1.2	OK
L3retriever	13	1.3	39	3.0	OK
(http_inspect)	13	0.4	147	11	NOK
DCERPC 1	13	0.4	15	1.2	OK
DCERPC 2	5	0.1	14	2.8	OK
SMB-DS IPC$ 1	13	0.4	24	1.8	OK
SMB-DS IPC$ 2	4	0.2	9	2.3	OK
SMB IPC$	13	0.3	22	1.7	OK

the alert reduction tells us the overall proportion of the normal behavior in these flows.

As each alert flow contains over 60 k observations obtained over 43 days with $t_s = 1$ min, the number of observations makes plotting infeasible. In addition, the flow behavior is rather stable over time. For these reasons we present filtering results using only selected parts of the flows.

Table 3 summarizes the numbers for these flows. In column *Flow* we have the flow names. The number of observations and alerts in the flows is given respectively in columns *Observ* and *Alerts*. The method signals *An* anomalies and the proportion of anomalous intervals is given in column *Prop*. The last column, *OK*, shows whether the non-stationary AR method with used set of parameters is suitable for processing the flow. We consider the NAR method unsuitable when the proportion of intervals requiring operator's attention is larger than 1%, in which case either method parameters should be tuned or alternative approaches used.

For all flows, the number of intervals with zero alerts was small. The number was highest for flow `(http_inspect) BARE BYTE UNICODE ENCODING`, 190 times. Most often the alert intensity was zero less than ten times. This means that without automated processing, the operator would constantly need to check the state of the flows. The proportion of intervals that can be discarded with the processing method is significant, and the saved time can be spent on other tasks.

5.3. Discussion

Overall, our alert processing method allows us to filter out a large majority of alerts related to normal flow behavior. This is the most important property of the method from our point of view. The method also highlights a large majority of abrupt changes in alert intensity. The abrupt changes missed by the detection algorithm are still often visible in the residual series. In other words, the detection algorithm is the weakest point of the processing method, but the modeling capacity is good.

Some of the abrupt changes may be uninteresting, but given the significant overall reduction in intervals needing operator's attention, we do not consider this as a major weakness. The method allows the operator to focus on a small set of potentially interesting phenomena that can be brought to his attention automatically.

Even though we lose the visibility to past events further than 20 min away with non-stationary AR method, $p = 20$ and $t_s = 1$ min, we are able to filter the daily and weekly behaviors remarkably well. Non-stationarity of the model is one and adaptivity of the estimation algorithm is another reason for this. Also the short sampling interval makes the situation easier, since the shifts are dispersed over several observations, and the model is able to adapt to those smooth-enough changes. On the other hand, this means that we are likely to miss anomalies causing intensity changes at a similar rate of change.

Out of the seven flows analyzed here, the used algorithm parameters worked in five cases. Of the two remaining, in the case of `SNMP request udp`, it was easy to find a suitable set of algorithm parameters so that the problematic high frequency component is not signaled as anomaly anymore. In the case of `(http_inspect) BARE BYTE UNICODE ENCODING` the problem is the nature of the alert flow. The flow is full of alert bursts i.e. the bursts that NAR model detects are not flow anomalies, but its normal behavior. Our method, indeed, is not suited for processing that type of alert flow.

Also for the other flows it is possible to fit the method parameters better for the particularities of each flow. These particularities arise from (1) the type and the amplitude of the high frequency fluctuations, and (2) the strength and the stability of the weekly and daily rhythms. There is, however, always a compromise to make. If we wish to flag less oddities of the normal behavior in a ragged and unstable flow, we increase the risk of missing some interesting phenomena at the same time. Typically preceding strong variations – normal or anomalous – can mask interesting flow behaviors from the detection algorithm. The risk increases when we need to accommodate more and stronger variations into the normal behavior.

In all cases, the adjustments were made using just a few days sample covering both weekend and weekday behavior. Thus it would be possible to do this in practice by collecting training data from the monitored system. We consider, however, more important the possibility of using general parameters that work rather well with most of the flows. This generality means more straightforward deployment in practice.

5.4. Changing the time scale of analysis

One limitation of the method is its single-scale nature. The time scale of detected anomalies is close to t_s. Shorter term anomalies risk being averaged out, and longer term anomalies lost in the noise. As was discussed in Section 4, the normal behavior is visible at all scales. We have also used three different sampling intervals, $t_s = 1, 20, 60$ min for analyzing the same alert flow. The model can be used to filter out effectively the normal flow behavior at all three time scales, and the abrupt changes at each time scale are detected. Thus a different t_s or a combination of models with different t_s can be used to cover the time scale(s) of interest.

6. Conclusions

In this paper we have presented an approach for modeling and filtering irrelevant alerts from alert flows. The basic assumption is that regularities and smooth changes in the alert intensity are considered as echoes of normal system use. This normal behavior is not observable at alert level, and thus we monitor alert flows.

Anomalies are detected from residual series obtained as the difference between observations and predictions given by the model. We signal errors that are large with respect to recent average and variance in the error series by using a modified exponentially weighted moving average (EWMA) control chart.

Previously we have used exponentially weighted moving average (EWMA) and stationary autoregressive (AR) models for this task. Here we use a non-stationary AR (NAR) model and estimate its parameters with a Kalman fixed-lag smoother algorithm. This combination provides an adaptive model which is a clear advantage over the stationary AR models.

Compared to the adaptive EWMA model, the NAR provides a significant increase in model accuracy, and more consistent filtering and detection behavior. In the analyzed examples NAR model signaled essentially the same anomalies of interest as the EWMA model, with significantly less noise.

We used the approach to analyze a larger set of alerts from operational information system. The method filters out effectively alerts related to normal flow behavior, which was our main objective. The flows can contain significant variations, part of which are normal flow behavior. Therefore anomalies signaled by the method are to be considered as interesting phenomena worth *further investigation* i.e. verification by a human is required before any further action. With the used data, the method leaves 0.2–0.3% of intervals for the operator. This is a significant improvement compared to manual handling of alerts – be it individually or as flows.

The adaptivity of the Kalman smoother increases the risk of modeling abnormal behavior into the model compared to the previous approaches. Based on the observed data, we claim that anomalies in alert flows are very often abrupt changes in alert intensity. The linear AR model does not capture well abrupt changes, and this limits the risk of including unwanted behavior into the model of normal behavior.

The detection algorithm is the weakest point of the approach as it can miss an anomaly closely preceded by another anomaly. Apart from this problem, the abrupt changes are detected in a consistent manner.

With respect to existing approaches capable for on-line processing, the advantage is that we do not need as much knowledge of flow behavior such as packet size/alert intensity distributions.

The off-line approaches based either on discrete or continuous wavelets have better detection capabilities thanks to the multi-scale nature of wavelets. So our method trades the multi-scale detection against on-line processing capability.

In future, we plan to improve the detection algorithm or to study the possibility of using time–frequency representations suitable for on-line processing. The intuition is that regularities in alert flows could show up clearly at different frequencies than anomalies. Alternatively, applicability of approaches such as [42] could be studied.

References

[1] T.H. Ptacek, T.N. Newsham, Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, Technical Report, Secure Networks, Inc., January 1998.

[2] H. Debar, B. Morin, Evaluation of the diagnostic capabilities of commercial intrusion detection systems, in: Wespi et al. [43].

[3] B. Morin, H. Debar, Correlation of intrusion symptoms: an application of chronicles, in: Vigna et al. [44], pp. 94–112.

[4] P. Ning, Y. Cui, D. Reeves, Constructing Attack Scenarios Through Correlation of Intrusion Alerts, Full Version of Paper in CCS'03, 2002.

[5] J. Viinikka, Intrusion Detection Alert Flow Processing Using Time Series Analysis Methods, Ph.D. Thesis, University of Caen, Caen, France, November 2006.

[6] M. Roesch, Snort-lightweight intrusion detection for networks, in: Proceedings of LISA'99, Seattle, Washington, USA, 1999.

[7] J. Viinikka, H. Debar, Monitoring IDS background noise using EWMA control charts and alert information, in: Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID), vol. 3224 of Lecture Notes in Computer Science, Springer-Verlag, 2004.

[8] J. Viinikka, H. Debar, L. Mé, R. Séguier, Time series modeling for ids alert management, in: Proceedings of the ACM Symposium on Information, Computer and Communications Security(AsiaCCS'06), Taipei, Taiwan, 2006, pp. 102–113.

[9] A. Scherrer, N. Larrieu, P. Borgnat, P. Owezarski, P. Abry, Non gaussian and long memory statistical modeling of internet traffic, in: Proceedings of the Internet Performance, Simulation, Monitoring and Measurements, Salzburg, Austria, 2006.

[10] S. Manganaris, M. Christensen, D. Zerkle, K. Hermiz, A Data Mining Analysis of RTID Alarms, 2nd International Symposium on Recent Advances in Intrusion Detection (RAID 1999), 1999.

[11] K. Julisch, Mining alarm clusters to improve alarm handling efficiency, in: Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC2001), 2001.

[12] K. Julisch, M. Dacier, Mining intrusion detection alarms for actionable knowledge, in: Proceedings of Knowledge Discovery in Data and Data Mining (SIGKDD), 2002.

[13] K. Julisch, Clustering intrusion detection alarms to support root cause analysis, ACM Transactions on Information and System Security 6 (4).

[14] B. Morin, L. Mé, H. Debar, M. Ducassé, M2D2: a formal data model for ids alert correlation, in: Wespi et al. [43], pp. 115–137.

[15] C. Kruegel, W. Robertson, Alert verification: determining the success of intrusion attempts, in: Proceedings of the 1st Workshop on the Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), Dortmund, Germany, 2004.

[16] F. Cuppens, A. Miége, Alert correlation in a cooperative intrusion detection framework, in: Proceedings of the 2002 IEEE Symposium on Security and Privacy, 2002.

[17] S. Cheung, U. Lindqvist, M.W. Fong, Modeling multistep cyber attacks for scenario recognition, in: Proceedings of the third DARPA Information Survivability Conference and Exposition(DISCEX III), Washington DC, USA, 2003.

[18] S.J. Templeton, L. Karl, A requires/provides model for computer attacks, in: Proceedings of the ACM New Security Paradigms Workshop, Cork Ireland, 2000, pp. 31–38.

[19] A. Valdes, K. Skinner, Probabilistic alert correlation, in: W. Lee, L. Mé, A. Wespi (Eds.), Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2001), vol. 2212 of Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, Germany, 2001.

[20] O. Dain, R.K. Cunningham, Building scenarios from a heterogeneous alert stream, IEEE Transactions on Systems, Man and Cybernetics.

[21] X. Qin, W. Lee, Discovering novel attack strategies from infosec alerts, in: Proceedings of The 9th European Symposium on Research in Computer Security (ESORICS 2004), Sophia-Antipolis, France, 2004.

[22] C.C. Zou, W. Gong, D. Towsley, L. Gao, The monitoring and early detection of internet worms, IEEE/ACM Transaction on Networking 13 (5) (2005) 961–974.

[23] A. Soule, K. Salamatian, A. Nucci, N. Taft, Traffic matrix tracking using Kalman filters, ACM Sigmetrics Performance Evaluation Review 33 (3) (2005) 24–31.

[24] J. Hall, M. Barbeau, K.E., Enhancing intrusion detection in wireless network using radio frequency fingerprinting, Extended Abstract in Communications, Internet and Information Technology, 2004.

[25] A. Valdes, K. Skinner, Adaptive, model-based monitoring for cyber attack detection, in: Debar et al. [45], pp. 80–93.

[26] R.P. Goldman, W. Heimerdinger, S.A. Harp, C.W. Geib, V. Thomas, R.L. Carter, Information modeling for intrusion report aggregation, in: Proceedings of the DARPA Information Survivability Conference and Exposition, 2001, pp. 329–342.

[27] C.-M. Cheng, H.T. Kung, K.-S. Tan, Use of spectral analysis in defense against dos attacks, in: Proceedings of IEEE Globecom 2002, 2002.

[28] A. Hussain, J. Heidemann, C. Papadopoulos, A Framework for Classifying Denial of Service Attacks, Technical Report ISI-TR-2003-569b, USC/Information Sciences Institute, August 2003.

[29] A. Hussain, J. Heidemann, C. Papadopoulos, Identification of Repeated DoS Attacks using Network Traffic Forensics, Technical Report ISI-TR-2003-577, USC/Information Sciences Institute, August 2003.

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

[31] A.G. Tartakovsky, B.L. Rozovskii, R.B. Blazek, H. Kim, A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods, IEEE Transactions on Signal Processing 54 (9) (2006) 3372–3382.

[32] V.A. Siris, F. Papagalou, Application of anomaly detection algorithms for detecting syn flooding attacks, in: Proceedings of IEEE Global Telecommunications Conference (Globecom 2004), Dallas, USA, 2004, pp. 2050–2054.

[33] P. Barford, J. Kline, D. Plonka, A. Ron, A signal analysis of network traffic anomalies, in: Proceedings of ACM SIGCOMM Internet Measurement Workshop, 2002.

[34] A. Dainotti, A. Pescapé, G. Ventre, Wavelet-based detection of dos attacks, in: Proceedings of IEEE Global Telecommunications Conference (Globecom 2006), 2006.

[35] J. Durbin, S.J. Koopman, Time Series Analysis by State Space Methods, Oxford University Press, Ofxord, Great Britain, 2001.

[36] M. Tarvainen, Estimation Methods for Nonstationary Biosignals, Ph.D. Thesis, Department of Applied Physics, University of Kuopio, Kuopio, Finland, 2004.

[37] J.P. Kaipio, E. Somersalo, Statistical and Computational Inverse Problems, vol. 160 of Applied Mathematical Sciences, Springer-Verlag, Heidelberg, Germany, 2005.

[38] R.P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. McClung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, M.A. Zissman, Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation, in: Proceedings of the 2000 DARPA Information Survivability Conference and Exposition, vol. 2, 2000.

[39] R. Lippmann, J.W. Haines, D.J. Fried, J. Korba, K. Das, Analysis and results of the 1999 darpa off-line intrusion detection evaluation, in: Debar et al. [45], pp. 162–182.

[40] J. McHugh, The 1998 lincoln laboratory IDS evaluation – a critique, in: Debar et al. [45], pp. 145–161.

[41] G.E.P. Box, G.M. Jenkins, G.C. Reinsel, Time Series Analysis: Forecasting and Control, third ed., Prentice-Hall International, Inc., Upper Saddle River, New Jersey, 1994.

[42] A. Scherrer, N. Larrieu, P. Owezarski, P. Borgnat, P. Abry, Une caractérisation non gaussienne et é longue mémoire du trafic internet et de ses anomalies, in: Proceedings of the 5th Conference on Security and Network Architectures (SAR 2006), Seignosse, France, 2006, pp. 29–50.
[43] A. Wespi, G. Vigna, L. Deri (Eds.), Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), vol. 2516 of Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, Germany, 2002.
[44] G. Vigna, E. Jonsson, C. Kruegel (Eds.), Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003), vol. 2820 of Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, Germany, 2003.
[45] H. Debar, L. Mé, S.F. Wu (Eds.), Proceedings of the 3rd International Symposium on Recent Advances in Intrusion Detection (RAID 2000), vol. 1907 of Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, Germany, 2000.