Performance Evaluation of the Information Sink in a Multi-Probe Statistical Anomaly Detection System

Thomas Zinner, Dirk Staehle, Phuoc Tran-Gia,
Andreas Mäder
University of Wuerzburg, Institute of Computer Science,
Department of Distributed Systems,
Am Hubland, 97074 Würzburg, Germany.
Email:{zinner,staehle,trangia,maeder}@informatik.uni-wuerzburg.de

Kurt Tutschku[1]
National Institute of Information
and Communications Technology (NICT),
Network Virtualization Lab,
1-33-16, Hakusan, Bunkyo-ku, Tokyo 113-0001, Japan.
E-mail: tutschku@nict.go.jp

***Abstract*—Statistical anomaly detection (SAD) becomes an increasingly important tool for the early recognition of potential threats for security-relevant information systems. SAD systems heavily rely on the probing of potentially very large networks. Our contribution is an analysis of the resource requirements on the information sink which constitutes the bottleneck of Client/Server-based SAD systems. In order to dimension the system appropriately, we investigate the trade-off between accumulated and distributed arrival patterns, and the impact of the processing phase of the information sink.**

I. Introduction

Statistical anomaly detection (SAD) systems are designed to recognize threats by analyzing the behavior of potential attackers. The basic principle is to monitor the data flow in a network and to compare the statistical properties with empirical values gained from past measurements. Thus, SAD systems require firstly the permanent monitoring of network nodes, and secondly an information processing instance to perform the required computations. Currently, the most common architecture approach for SAD systems follows a the Client/Server design, where the network probes (the clients) send their measured data to an information sink (the server).

Unfortunately this design is prone to bottlenecks at the server. Such bottlenecks can be avoided by an appropriate dimensioning of the performance parameters. In this paper we investigate analytically the performance of such a system and give guidelines for dimensioning of the information sink.

An example for such a platform, originating from the network security area, is the "Internet Analysis System" (IAS), [4]. This platform is based on a Client/Server architecture with one data sink and many data sources, which are distributed over many different subnetworks. The aim of this system is to derive the network harassment, like worm dispersion or distributed Denial-of-Service attacks, by using statistical anomaly detection based on measured data. In advance, it is also possible to take steps against a threat once it is identified. In order to collect the necessary measurements, each probe monitors the network flow over a node and logs the relevant data used for the evaluation. In order to satisfy the confidentiality regulations, this system counts only anonymous information like the protocol type or the port number of the traffic passing over the link. Personal information, e.g. the IP-Address, are discarded. The measured counters are sent periodically to the server, which performs an evaluation on the coherent data of different network probes. We call this time which is needed to perform this evaluation *reaction time*.

Other SAD systems are *Packet Header Anomaly Detection* (PHAD) [2], the *Application Layer Anomaly Detection* (ALAD) [3] or the *Event Monitoring Enabling Responses to Anomalous Live Disturbances* System (EMERALD), described in [6]. PHAD and ALAD are both models for SAD systems. In contrast to these models, EMERALD is a system which can be compared to the IAS.

EMERALD introduces a hierarchically layered approach to network surveillance. It includes a single analysis, a domain wide analysis and an enterprise-wide analysis. Due to the hierarchical approach of this system measurement data is evaluated and aggregated on the different layers. This means, that the raw data is not available on higher layers. This is the main difference to our specific application, where measurement data are sent from the distributed probes to the evaluation unit. The performance evaluation presented in this paper is developed for the IAS, but can easily be extended to other SAD systems, like e.g. EMERALD.

By using a single server architecture the typical design questions arise. Which buffer size is needed to ensure a fully functional system? What is the blocking probability for a requests for a given buffer size? How long does it take to evaluate the measurement data? In this work we examine the trade-off between the buffer occupation and its impact on the reaction time of the SAD system. In general, the presented results can be used by administrators to design such a system and guarantee policy periods for evaluation and reaction times.

The paper is structured as follows: Section II discusses the investigated application. In Section III we present the model for this application, and in Section IV we show the results of our investigation. Finally, in Section V the paper is concluded and an outlook for further research is given.

[1] Work was started when K. Tutschku was with the University of Wuerzburg

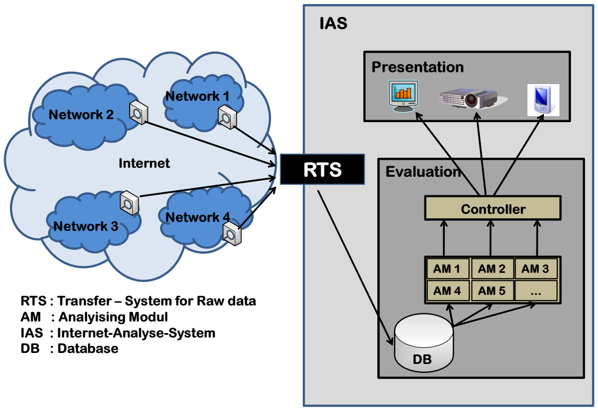


Fig. 1. Architecture of the IAS-SAD-system.

II. System Description

The IAS is an intrusion and malware identification system based on statistical anomaly detection. This kind of technique is described in [5]. The architecture of the IAS, which is illustrated in Figure 1, is separated into three parts: various network probes a central information sink, the transfer system for raw data and an evaluation and presenting system.

The network probes are spread over the investigated network and serve as information providers. They extract and keep counters for predefined communication parameters. This could be, for instance, the number of TCP and UDP packets passing over the observed link. Its also possible to capture e.g. the number of packets belonging to a special IP address range used as bot-nets or special ports which are known to be used for controlling trojan horses. The number of observed communication parameters can reach up to tens of thousands.

In periodic intervals, counter data are sent to the Raw Data Transfer-System, where they are collected and forwarded to the evaluation. The measurements are sent as <*identifier–counter*>tuples with a size of 8 byte. Since counters are reset after transmission, they describe the difference of the observed parameters with respect to the previous counters. In order to reduce the amount of data, counters which have not changed in the current interval are not transmitted. This leads to a variation of the number of counters which are sent to the server in each interval.

The collected data is stored into a database and then made available for the anomaly detection algorithms. These algorithms perform a comparison of the current counters with predefined thresholds in order to identify local threats. To achieve additionally a superior view of the network, chronologically correlated data from several network probes are evaluated. For that reason, the data of each sensor concerning one measurement interval must be available at the RTS as fast as possible to allow a fast and effective evaluation. The time which is lost during the analysis of the measurement data is time which is also lost for the protection of the observed system.

In order to achieve a fast overview over the network harassment, it would be the best to synchronize the probes and get all the measurements at once. However, due to buffer restrictions, this could lead to buffer overflows, data loss and delay. One the other hand, by distributing the arrival times over the whole interval the buffer occupation would be kept to a minimum. But this would slow down the examination of the connected data sets. The aim of this work is to discuss the trade off between buffer size and examination delay, and to determine the impact of different arrival patterns on the design of the system. In the investigated case we assume, that the server and not the network is the bottleneck of the SAD system.

III. Model

A. Abstract Server Model

We consider scenarios with multiple clients and one server. In particular the clients are network probes which send a packet consisting of measurement counters in periodic intervals to the server. The server stores the data in a database. Data arriving at the server while it is busy is stored into a queue, which we assume to be infinite throughout this work. This assumption simplifies the analysis of the system and allows us to discuss overall processing times and buffer sizes.

We assume that the server processes each counter within a constant processing time. Since the number of counters arriving in each packet is varying, the processing time of a packet is also varying.

B. Definitions and Performance Metrics

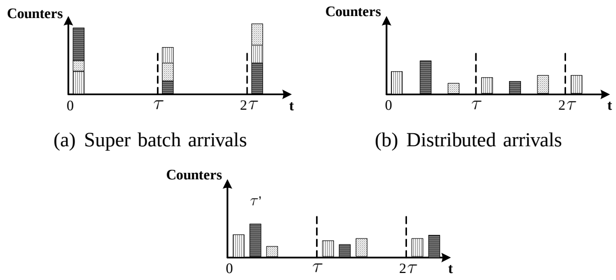
We assume that the network probes transmit their observed counter measurements in equally spaced time slices of constant length τ. There are m network probes in the system which send their data to the server. The number of counters C_i sent by sensor node i to the server per time slice is variable and follows a probability distribution $c_i(x)$ with mean $E[C_i]$ and standard deviation STD$[C_i]$. The processing time t_c for a counter at the information sink is constant, such that the service time for probe data i follows as $B_i = t_c \cdot C_i$ with distribution $b_i(t) = \lceil c_i(t/t_c) \rceil$. In this work we assume the counter distribution to be independent and identically distributed, i.e. $c_i(x) = c(x), i \in \{1, ...m\}$.

The relevant performance metrics are the *buffer occupancy* O at the information sink and the *reaction time* R, defined as the time between the beginning of a time slice and the completed processing of the last probe data, belonging to this time slice, at the RTS. Thus, after the interval $\mathcal{R}$ the system has a decision on the current threat level in the network and may react accordingly.

C. Arrival Patterns

We consider three different arrival patterns for the counter information data at the sink. These patterns describe when the network probes are scheduled to send their measurements, and are categorized as follows:

1) *Super batch arrival pattern* This pattern is depicted in Figure 2(a). The measured data of the network probes



(a) Super batch arrivals

(b) Distributed arrivals

(c) Distributed arrivals with processing phase

Fig. 2. Arrival patterns for network probe data at the information sink.

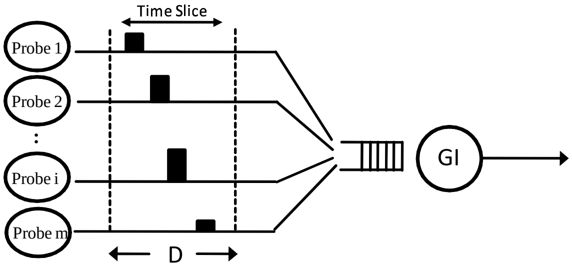


Fig. 3. Queueing model

are sent synchronously and arrive simultaneously at the server. Because of different latencies between probes and sink, the measurement data arrive in a small interval at the server and not exactly at the same time. However, in our analysis we assume perfectly synchronized arrivals. Although this pattern has stringent requirements on the clock synchronization in the network probes, it has the advantage that additional probes can be easily added to an existing installation without overly large configuration. A disadvantage is that the cumulated arrivals of all probe data at the same time leads to increased demands on the buffer capacity of the server.

2) *Distributed arrival pattern* This arrival pattern is depicted in Figure 2(b). Here we assume that the inter-arrival time between the probes is constant and equally distributed in the interval τ. The inter-arrival time of two requests is $\frac{\tau}{m}$. The advantage of this pattern is that the network utilization as well as the buffer occupation can be expected to be lower than in the case of super batch arrivals. Since the request arrival times are spread over the whole time slice, it is likely that a request has been processed completely before the next request arrives. A disadvantage of this pattern is that the information sink may have to wait for incoming probe data, thus wasting time which could be used for data processing. Thus, in high load situations it may also happen that the last probe data has not been processed with the beginning of a new time slice. Also, the installation of additional probes in the network requires the re-configuration of all probes due to the changed transmission order and time.

3) *Distributed arrival pattern with processing phase* With this arrival pattern the time slice τ is divided into two intervals τ' and $\tau - \tau'$ which we denote as *transmission phase* and *processing phase*. This approach is shown in Fig. 2(c). In the transmission phase τ', all probes are sending their data to the information sink with constant inter-arrival times τ'/m, similar to the *distributed arrival pattern* over the whole interval τ. The processing phase $\tau - \tau'$ is then exclusively used for processing the data. This pattern has the advantage that the idle time between two probe arrivals is reduced, and at the same time the buffer requirements are reduced if compared to the super batch arrival pattern. We will see later how the parameter τ' can be used to tune the system either for buffer efficiency or for shorter processing times, and thus for faster reactions.
Furthermore additional probes can be easily added to an existing installation by scheduling them after the last probe arrival. In this case, the length of the transmission phase changes.

D. Performance Model

We model the SAD system as a $D/GI/1$ delay queuing system as shown in Fig. 3. We assume an infinite queue with a first-come-first-served order, i.e. no probe data will be lost.

We compute the desired system parameters buffer occupancy O and the reaction time R with a discrete-time analysis [1] [7]. The state space is defined by means of the number of unprocessed counters in the system, U. Transitions between different states are described with state transition matrices. The time diagram of the investigated process including the state transitions is depicted in Figure 4. We assume an equally spaced inter-arrival time Δt between the requests in a time slice. Depending on the arrival pattern, the inter-arrival time takes values $\Delta t \in [0, \frac{\tau}{m}]$. We will use the following notation, which is also used in Figure 4:

$u_k(x)$ = unfinished work at the begin of time slice k

$\mathcal{Q}(\tau)$ = state transition matrix between two observation points

$\mathcal{P}(\Delta t)$ = state transition matrix between two arrivals

$\mathcal{S}(\Delta t)$ = state transition matrix depicting a reduction of unfinished work for an interval of length Δt

$t_{k,i}$ = point of the i-th arrival in time slice k

We define the start points of each interval as observation points. At this regeneration points the remaining work load U_k can be computed, cf. [8], as

$$u_k(x) = u_{k-1}(x) \cdot \mathcal{Q}. \tag{1}$$

We assume that the counter distributions and for that reason also the service times are independent and identically distributed, i.e. $c_i(t) = c(t)$ and $b_i(t) = b(t)$. Thus the steady state

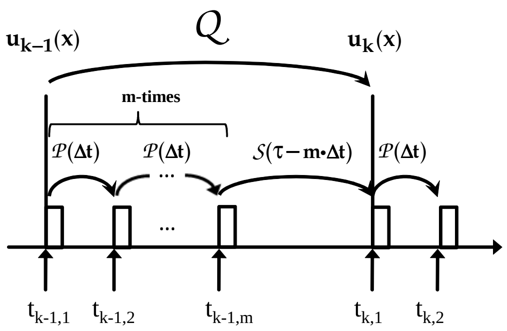


Fig. 4. Time diagram of the process including state transition matrices

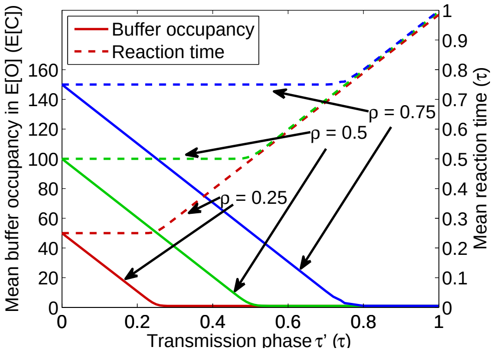


Fig. 5. Mean buffer occupancy and mean reaction time vs. τ'

distribution for the unfinished work load is

$$u(x) = u(x) \cdot \mathcal{Q}. \tag{2}$$

The state transition matrix $\mathcal{Q}$ between the observation points is computed as:

$$\mathcal{Q} = \mathcal{P}^m(\Delta t) \cdot \mathcal{S}(\tau - m \cdot \Delta t), \tag{3}$$

i.e. it is composed of m data arrivals and a variable processing phase. Note, that for the arrival pattern *distributed arrival* no additional processing phase is used. In this case, the matrix $\mathcal{S}$ becomes the Identity matrix $\mathcal{I}$. $\mathcal{P}(\Delta t)$ consists of two parts. First an arrival adds additional work with distribution $b(t_{k,i})$ to the information sink. This work distribution is represented by the state transition matrix $\mathcal{B}$. Secondly work can be processed between two successive arrivals, which is denoted by the state transition matrix $\mathcal{S}(\Delta t)$. This yields

$$\mathcal{P}(\Delta t) = \mathcal{B} \cdot \mathcal{S}(\Delta t), \tag{4}$$

with

$$\mathcal{B}_{i,j} = P(B = j - i), \tag{5}$$

$$\mathcal{S}_{i,j}(\Delta t) = \begin{cases} 1, \text{if } j = \max\{0, i - \Delta t/t_c\} \\ 0, \text{else.} \end{cases} \tag{6}$$

Thus, we compute the buffer occupancy distribution $o(x)$ after the last packet arrival in a time slice as

$$o(x) = o_m(x) = u(x) \cdot \mathcal{P}^{(m-1)}(\Delta t) \cdot \mathcal{B}. \tag{7}$$

The reaction time R comprises the service time of the last request within a time slice and the unfinished work which is queued in the buffer, i.e $R = (m-1) \cdot \Delta t + O_m + B_m$. Therefore we compute the reaction time distribution as

$$r(t) = \delta(t - ((m-1)\Delta t)) \circledast o_m(x/t_c) \circledast b(t) \tag{8}$$

IV. Numerical Results

Let us now have a look at some numerical results in order to get an impression of the system performance. We consider homogeneous scenarios where the probability distributions for the number of counters of all network probes are identically, thus, we assume i.i.d. service times. We further assume the counter distribution to follow a negative binomial distribution.

This allows for a simple calculation of the maximum number of probes the information sink can process. Under stable conditions, i.e. $\rho < 1$, this number can be expressed as

$$m_{max} < \left\lfloor \frac{\tau}{E[B]} \right\rfloor, \tag{9}$$

the length of a time slice divided through the mean processing time of one probe.

Accordingly, the load can be expressed by the approximate equation

$$\rho \approx \frac{m}{m_{max}}. \tag{10}$$

As an example, let us assume a scenario with $m_{max} = 200$ supported network probes. That leads to an average processing time for each counter of $b = \frac{\tau}{E[C]\cdot 200}$. Under this assumption, the number of network probes can be computed for a given system load ρ.

We show the influence of different system loads ρ and coefficients of variations of the counter distribution $c_v = \frac{E[C]}{STD[C]}$ on the investigated system parameters buffer occupancy O and reaction time R.

A. Macroscopic Behavior

First, we investigate the influence of the length of the transmission phase τ' on the mean buffer occupancy and on the mean reaction time as shown in Fig. 5. We assume different load values as $\rho \in \{0.25, 0.5, 0.75\}$ with a corresponding number of probes $m \in \{50, 100, 150\}$. The coefficient of variation of the counter distribution is set to $c_v = 0.4$. The solid lines indicate the average buffer occupancy given as multiples of the mean packet size $E[C]$. The dashed lines show the corresponding reaction time, i.e. the time until the last probe data has been processed. The reaction time, as well as the transmission phase τ' is presented as fraction of the time slice length τ.

Let us define $\tau'^* = \rho \cdot \tau$ as average time the sink needs to process all data *without* consideration of any idle phases. Thus, the curves are divided into two parts, one with $\tau' < \tau'^*$ and one with $\tau' > \tau'^*$.

For the case $\tau' < \tau'^{*}$, the arriving measurement data can not be entirely processed in the transmission phase, queuing occurs and the buffer occupancy increases with decreasing τ'. For $\tau' = 0$ the arrivals correspond to the super batch arrival scheme, i.e. the measurement data of all probes arrive simultaneously at the information sink and the buffer occupancy reaches its maximum. Accordingly, the mean reaction times correspond to the average processing time τ'^{*}.

For $\tau' > \tau'^{*}$ the mean buffer occupancy is $E[C]$ independent of the transmission phase length. The probe inter-arrival time is large enough to allow a complete processing of the measurement data of one probe before the next data set arrives. At most one data set has to be stored in the queue. Since the reaction time is dominated by the processing time of the last probe data, it increases linearly with the transmission phase.

We observe that the curves are congruent for values of τ' far from τ'^{*}.

Now let us investigate the influence of the length of the transmission phase τ' on the mean buffer occupancy and on the mean reaction time for a constant load value $\rho = 0.5$ and different values $c_v = 0.4, 1, 2, 3$. The results are depicted in Fig. 6. The solid lines indicate the average buffer occupancy given as multiples of the mean packet size $E[C]$. The dashed lines show the corresponding reaction time, i.e. the time until the last probe data has been processed. The reaction time, as well as the transmission phase τ' is presented as fraction of the time slice length τ.

However, for values of τ' close to $\tau'*$ the mean buffer occupancy and the mean reaction time increase with increasing c_v. We can conclude that for a high c_v the value of τ' has to be chosen smaller than $\tau'*$ in order to achieve the same reaction time as for a low c_v. It should be noted, that this leads to a higher buffer occupancy, as indicated by the solid lines.

B. Buffer and Reaction Time Distributions

Let us now investigate the trade-off between the buffer occupancy and the reaction time for selected values of the transmission phase for a fixed system load of $\rho = 0.5$ and $c_v = 0.4$. Figure 7 shows the cumulative distribution function of the buffer occupation. Note that the x-axis is scaled logarithmically. For $\tau' \neq 0.5$ the curves rise quickly, i.e. the variation is small. For $\tau' > 0.5$ this is due to the small variation of the counter data and for $\tau' > 0.5$ this is due to the aggregation of the probe data in the queue.

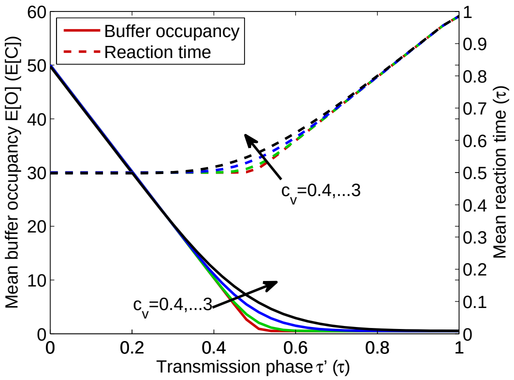


Fig. 6. Mean buffer occupancy and mean reaction time vs. τ' for different c_v

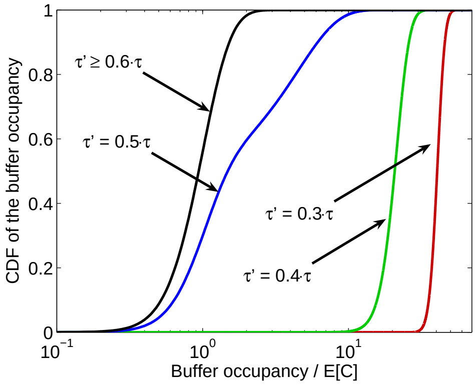


Fig. 7. Buffer occupation for the arrival pattern distributed arrival with passive phase for different τ', $c_v = 0.4$

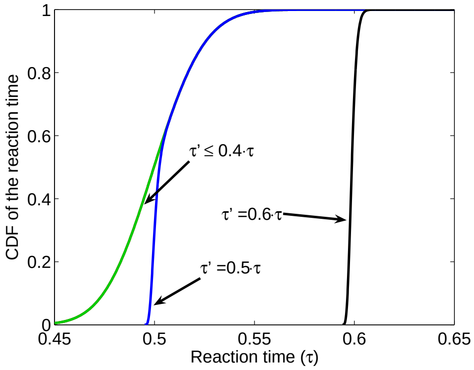


Fig. 8. Reaction time for the arrival pattern distributed arrival with passive phase for different τ', $c_v = 0.4$

If the transmission phase is identical to the average processing time, the shape of distribution follows from the superposition of probe data still queued from previous arrivals, and from the arrival of the last probe data. This leads to a comparatively large variation.

The distribution of the reaction time is shown in Fig. 8. The variation clearly decreases and the average reaction time increases with the transmission phase. The curves with short transmission phases of $\tau' = 0.3 \cdot \tau$ and $\tau' = 0.4 \cdot \tau$ show the largest variation which corresponds to the observation that in this case the processing time is dominating the reaction time. In case of $\tau' = 0.6 \cdot \tau$, the reaction time is nearly deterministic and equal to the transmission phase. Note that this behavior is a desirable property for real-time systems. In case of a transmission phase equal to the processing time, the variation is smaller than in case of lower values of τ'. The distribution is right-skewed due to the processing of queued data probes from previous arrivals.

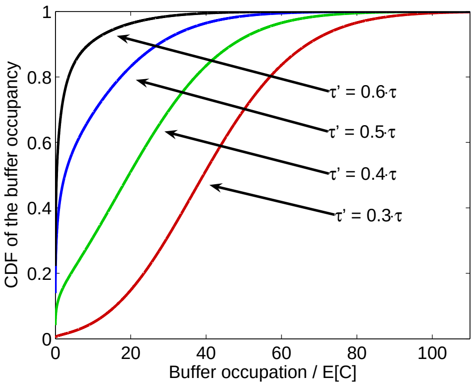


Fig. 9. Buffer occupation for the arrival pattern distributed arrival with passive phase for different τ' , $c_v = 2$

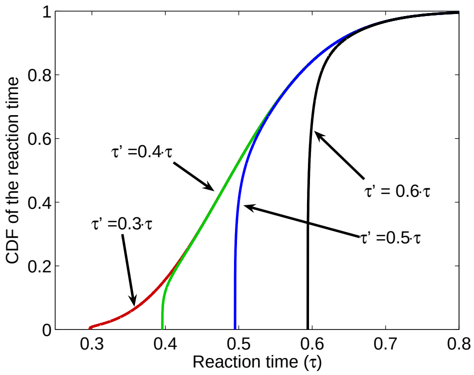


Fig. 10. Reaction time for the arrival pattern distributed arrival with passive phase for different τ', $c_v = 2$

The distributions of the buffer occupancy and the reaction time for selected values of τ' are depicted in Figure 9 and Figure 10. The system load and the variation are fixed to $\rho = 0.5$ and $c_v = 2$.

Let us investigate how to use the evaluated results for the design of the presented system. We want to identify the interaction of the reaction time and the buffer occupancy. Let us assume a buffer size of $60 \cdot E[C]$ for the system and a desired $90\%-$quantile for the reaction time of $0.6 \cdot \tau$. First we consider a small variation of the counter distribution $c(x)$. In Figure 7 we can see that we always have a buffer occupancy smaller than $60 \cdot E[C]$ for the presented values of τ'. Note that for $\tau' = 0.2 \cdot \tau$, the buffer occupancy exceeds the given buffer constraint. From Figure 8 we can conclude that only for a value of $\tau' = 0.6 \cdot \tau$, the desired $90\%-$quantile for the reaction time is exceeded. From this it follows that $\tau' \in \{0.3, 0.4, 0.5\} \cdot \tau$ fullfilles the given requirements. From this values, the reaction times for $\tau' \in \{0.3, 0.4\} \cdot \tau$ are equal and smaller than the others. That means, that in order to reduce the buffer occupancy a value of $\tau' = 0.4 \cdot \tau$ should be used to tune the system.

Now we want to investigate the system with the same constraints for a higher variation of $c(x)$. From Figure 9 we can conclude, that only for $\tau' \in \{0.5, 0.6\} \cdot \tau$ the buffer size requirement is fulfilled. But, in Figure 10 we see, that for these, and also for the other values, the $90\%-$quantile of the reaction time is exceeded. Due to the high variation, the requirements can not be guaranteed. In order to fulfill the requirements the server would have to be dimensioned faster or the variation of the counter distribution would have to be reduced. Without such an extension, $\tau' = 0.5 \cdot \tau$ would be most suitable for the system. For a slightly larger buffer, $\tau' = 0.4 \cdot \tau$ would be possible and speed up the mean reaction time considerable.

V. Conclusion & Outlook

In this paper we investigated and developed a mathematical model for a SAD system. This approach can be used not only to design the presented application, but also other systems composed of multiple data providers and one information sink. With the introduced analytical method we compute the reaction time, an important performance metric for such a system. We further describe the trade-off between the reaction time and the buffer occupancy.

We investigate the system for different system loads, different variations of the counter distributions and different arrival patterns. We further show how the to use the presented results for the design of the system and explain how to tune the system to fulfill the given requirements.

Further work will have to deal with different types of sensor probes, classified by different counter distributions. Another open issue is the investigation of transmission failures and their influence on buffer occupancy and reaction times. Last but not least the model of the system could be extended by including an additional random service time denoting the influence of the operating system on the reaction times.

Acknowledgments

The authors would like to thank Barbara Staehle for the fruitful discussions and her outstanding valuable input during the course of this work.

References

[1] Kleinrock L. *Queueing Systems, Volume I: Theory*. Wiley Interscience, New York, 1975.

[2] M. Mahoney and P. Chan. Phad: Packet header anomaly detection for identifying hostile network traffic. In *Florida Tech. technical report CS-2001-4*, 2001.

[3] M. V. Mahoney and P. K. Chan. Learning nonstationary models of normal network traffic for detecting novel attacks. In *Proceedings of the eighth ACM SIGKDD*, pages 376–385, New York, NY, USA, 2002. ACM.

[4] M. Proest N. Pohlmann. Messverteilung Die globale Sicht auf das Internet. *iX*, 2:104–108, 2006.

[5] D. Q. Naiman. Statistical anomaly detection via httpd data analysis. *Computational Statistics & Data Analysis*, 45(1):51–67, February 2004.

[6] P. A. Porras and P. G. Neumann. EMERALD: event monitoring enabling responses to anomalous live disturbances. In *National Information Systems Security Conference*, October 1997.

[7] P. Tran-Gia. Discrete-time analysis technique and application to usage parameter control modelling in ATM systems. In *Proceedings of the 8th Australian Teletraffic Research Seminar*, Melbourne, Australia, December 1993.

[8] P. Tran-Gia. *Einführung in die Leistungsbewertung und Verkehrstheorie*. Oldenbourg, München, Deutschland, July 2006.