Key escrow schemes with sliding window for privacy-aware anomaly detection system

Esa Hyytiä*
Telecommunications Research Center Vienna (FTW), Austria
esa.hyytia@tkk.fi

Ivan Gojmerac
Telecommunications Research Center Vienna (FTW), Austria
gojmerac@ftw.at

Simone Teofili
University of Rome "Tor Vergata", Italy
simone.teofili@uniroma2.it

Giuseppe Bianchi
University of Rome "Tor Vergata", Italy
giuseppe.bianchi@uniroma2.it

ABSTRACT

Requirements for a traffic monitoring system can be very demanding as both privacy and performance aspects have to be taken into account jointly. Moreover, the legislation sets forth strict rules that must also be met. Various cryptographic primitives provide invaluable tools for realising privacy enforcing mechanisms in such a system with respect to the above mentioned goals. In this paper, we consider an arbitrary traffic anomaly detection system consisting of two stages. The first stage pre-processes the monitored traffic with both data rate reduction and privacy protection in mind. The second stage is in charge of the final analysis and storing the relevant information. In particular, the privacy sensitive information is encrypted on per flow basis by the first stage, so that the second stage cannot access any flow without an appropriate key, which is given only when there is a strong reason to do so. In this setting, we study a sliding window type of mechanism for escrowing a secret decryption key from the first stage to the second in response to observing a sufficient number of malicious events within a specified time duration. Given the flow specific key, the second stage can then take a closer look at the corresponding part of the traffic, and decide on further actions. As a result, the privacy of the other users cannot be violated.

Categories and Subject Descriptors

C.2.3 [**Computer systems organization**]: Network operations—*Network monitoring*; K.4.1 [**Public policy issues**]: Computers and Society—*Privacy*

Keywords

privacy, Shamir's scheme, key revocation, sliding window

*Dr. Hyytiä is currently with TKK, Finland

1. INTRODUCTION

Ever more powerful data mining techniques for different purposes are continuously being developed and deployed. Data centers have got practically infinite amounts of memory in many cases, and any information that once gets stored can be easily retrieved later on. Network operators are not an exception, and they also carry out various traffic monitoring activities for network planning purposes and the optimisation and protection of their networks. However, the information carried in the networks is often sensitive and, in particular, private in nature. In other words, there is a generally acknowledged conflict of interest between privacy preservation and the usefulness of the gathered data. Collecting and processing privacy sensitive data in a proper manner is indeed a complicated task. For example, the EU has come up with directives which set guidelines for what is considered as personal data (Directives 95/46/EC and 2002/58/EC) as well as legislation regarding data retention (Directive 2006/24/EC).

Various cryptographic primitives provide invaluable tools for realising privacy enforcing mechanisms in traffic monitoring systems. In this work, we consider a two-stage traffic monitoring system where the processing of the raw data is limited to the first stage [1]. In other words, the sole purpose of the first stage is to i) (proactively) prevent privacy violations, and ii) reduce the data rate to the second stage by forwarding only the relevant information. Note that ii) in fact already supports i) directly. At the second stage more complex operations can be carried out based on the pre-processed data from which privacy sensitive data was already removed, or are at least protected by appropriate cryptographic means.

To illustate the concept, let us consider a simplified problem with volume based billing, e.g., in a mobile data network. Suppose that a billing system fulfills two purposes: i) the system has to provide the monthly traffic volumes per customer for sending the bills, and ii) in case a customer repudiates a bill, the system must be able to provide a "more detailed proof" for the usage. Let us assume that a list of contacted IP addresses together with time and volume information constitutes a sufficient proof. The privacy problem we want to alleviate in this illustrative example scenario is

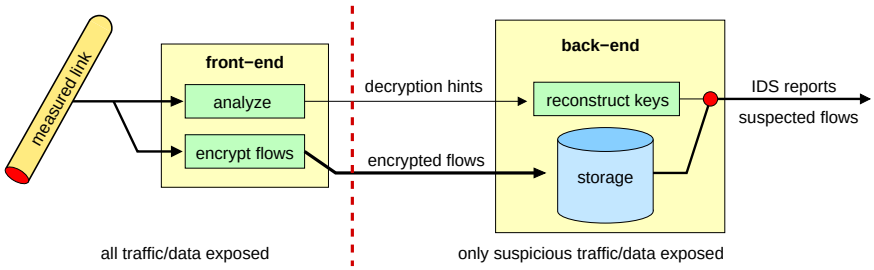

Figure 1: Privacy-aware traffic anomaly detection: front-end processes the monitored traffic in a way that back-end can carry out further analysis only on suspicious parts of the traffic. The system both enforces the privacy and improves the performance at the back-end.

one of a curious employee of the operator who wants to find out which Web servers some customer has visited, without the customer's consent. To this end, assume that each contract is given a unique identification number (which is a prerequisite for differentiating between the customers). The billing system then works as follows. The first stage provides two data streams:

1) $\{\text{hash}_1(\text{id}), \text{volume}\}$,
2) $\{\text{hash}_2(\text{id}), \text{dest addr}, \text{time}, \text{volume}\}$,

where the former is generated on a daily basis, and the latter is per flow information. The first stream is used for sending the bills to the customers, with the aid of address information obtained from a customer database. In case there of repudiation, the corresponding customer can provide his/her identification number, which then allows the parties to examine the claimed usage in detail. If the two hash functions are cryptographically strong, then there is no easy way to compute $\text{hash}_2(\text{id})$ for a given $h_1 = \text{hash}_1(\text{id})$. In particular, a curious employee has no means to compare the two databases directly. We note that in this two-stage design, the first stage has a very important role of being responsible for pre-processing the information flows for different purposes in a privacy preserving manner. This is in stark contrast to the traditional approach, where full traffic traces, containing a wealth of privacy sensitive information, are first collected, and then later used for different purposes. As a result, a high risk of infringing the users' privacy is always present.

1.1 Two-stage traffic anomaly detection

Our reference system for traffic anomaly detection system consists of two stages [1, 2]. The first stage, referred to as *front-end* (FE), is in charge of tapping the wire and carrying out a preliminary per-packet analysis in real-time. Efficient mechanisms, such as counting operations over a large base of possible flows [3, 4], can be deployed in the FE to this end. However a discussion of these techniques is out of the scope of the present paper.

In many cases, it is not possible to identify a given flow as malicious based on the first few packets. Due to strict real-time constraints, and assumed limited storage capacity, the front-end cannot store the packets for longer durations. Instead, the relevant, and typically reduced, packet or flow level information is immediately passed further to the second stage referred to as the *back-end* (BE). In order to protect the users' privacy, the traffic under observation is first encrypted on per-flow basis before the transmission to the back-end. The overall system is illustrated in Fig. 1. We note that a flow here can correspond to different combinations of packet properties defined by the traditional five tuple, e.g., "all packets from a single IP address", or "all packets with the same five tuple" with some maximum time interval between the packets. The fact that each flow is encrypted with a unique key allows one to reveal traffic on a per flow basis. When part of the traffic is deemed to be suspicious, the system allows one to take a closer look at the particular fraction of traffic, thus preserving the privacy of the other users.

The back-end stores the encrypted traffic traces for an appropriate duration of time that is defined by the applicable legislation and technical considerations. As all data is encrypted, the BE cannot decrypt the traffic at this stage and there are no privacy concerns. However, in case the preliminary data analysis at the FE suggests a potentially malicious activity, then, and only then, the decryption key for the particular flow (or flows) is revealed to the back-end by means of a key escrow process. This means that in this case it is considered justified for the operator at the back-end to take a closer look at the contents of the potentially malicious activity and to decide on further actions. Above we have assumed that when malicious activity is suspected, the operator wants, and is allowed to, look at the corresponding traffic at the packet level. Equally well, the same system can be used to expose only, e.g., flow level data or just the corresponding IP addresses.

More specifically, in [2], it is proposed that the information about the decryption keys is embedded into encrypted packet flow data by revealing a single part of the key (i.e., a *shadow*) in response to observing a suspected anomaly event. To this end, Shamir's secret sharing scheme [5] is used to protect the key until m events have been observed. In particular, the procedures at the FE are (close to) stateless and thus they especially well suit real-time operation. In some sense, the above means that FE outsources the counting of events to the BE, and only when the given threshold of m has been exceeded does the specific decryption key for flow f become available at the BE. We note that the time horizon of the resulting secret revealing process is infinite. In [6] this scheme is further extended by combining it with a class of erasure codes.

In this paper, we focus on studying similar processes where

an "oracle" mumbles the words of wisdom (in response to each detected suspected malicious event), and only if a sufficient amount of information about particular activity is received during a finite time interval, corresponding to a sliding time window, then a particular secret becomes available for the recipient (i.e., the BE obtains a flow specific decryption key). In other words, as the shadows expire, they and the corresponding data self-destruct automatically. We note that the self-destructing property has been identified as a very important factor also in other contexts [7]. The main contribution of this work is to demonstrate how a sliding window type of key escrow mechanism can be realised in this context, and in particular, that a stateless operation, required for highly scalable solutions, is also feasible.

The rest of the paper is organised as follows. Firstly, in Section 2 we introduce the notation and the key concepts relevant in the context of this work. Section 3 describes several different schemes for realising the sliding window type of behaviour in a key escrow process, after which Section 4 concludes the paper with a brief discussion of the proposed approaches and an outlook on future work.

2. PRELIMINARIES

Given a set of $k+1$ points, $(x_0, y_0), \ldots, (x_k, y_k)$, the Lagrange polynomial $L(x)$ is a polynomial of (at most) degree k such that $L(x_i) = y_i$ for all i, and is given by a linear combination,

$$L(x) := \sum_{j=0}^{k} y_j \cdot \ell_j(x), \tag{1}$$

of Lagrange basis polynomials,

$$\ell_j(x) := \prod_{i=0, i\neq j}^{k} \frac{x - x_i}{x_j - x_i}. \tag{2}$$

This holds for both rational and real numbers, as well as, for modulo p arithmetic with p prime.

In a (m, n) secret sharing scheme, n shadows (or shares) are distributed to n different parties in such a way that given any set of m shadows, $m \leq n$, one can reconstruct the secret S. Shamir's secret sharing scheme, given in Table 1, is based on a random polynomial of degree $m-1$ in finite modulo p field, where p is a prime and $m \leq n < p$. Given m points,

$$(x_1, P(x_1)), \ldots, (x_m, P(x_m)),$$

with $x_i \neq x_j$ when $i \neq j$, the polynomial $P(x)$ is well-defined and, in particular, $P(0)$ provides the secret. Note that once some points have been revealed there is no way to take this information back, i.e., revoke it, unless one somehow modifies the polynomial.

The key escrow scheme given in [2] uses Shamir's scheme. Threshold m defines the number of suspicious events resulting a decryption key disclosure at the back-end:

Scheme 1: (Basic)

1. *For each flow f, choose a secret decryption key S_f and define a secret polynomial*

$$P_f(x) = S_f + a_1^{(f)} x + \ldots + a_{m-1}^{(f)} x^{m-1} \mod p,$$

with random coefficients $a_1^{(f)}, \ldots, a_{m-1}^{(f)}$ where p is a constant prime.

2. *Upon observing a suspicious event for flow f, disclose a shadow $(x, P_f(x))$, with random x.*

Given m or more shadows, one can compute $P_f(x)$ using the Lagrange interpolation polynomial , (1) and $P_f(0)$ yields the secret S_f.

2.1 Stateless operation

When the flow specific coefficients $a_i^{(f)}$ of $P_f(x)$ can be obtained using a pseudorandom function, the front-end operation is essentially stateless:

Definition 1: *Key escrow scheme is **stateless** if a shadow, disclosed in response to an observed event for flow f at time t, can be computed efficiently as a function of triple (f, t, s^*), where s^* is a global salt.*

Scheme 1 is stateless in the sense of Def. 1. To this end, as explained in [2], we define the flow specific coefficients a_i of the Shamir's polynomial $P(x)$ as

$$a_i = h_i(s^*, f), \quad i = 1, \ldots, m-1,$$

where the $h_i(x)$ are appropriate pseudorandom functions to $[0, p-1]$, parameter s^* is a secret salt known only to the FE (a common constant), and the decryption key $a_0 = S_f$ is the constant term, $S_f = h_0(f)$ where $h_0(f)$ is some cryptographic hash function.

Stateless operation has two important implications. Firstly, stateless operation tends to translate to a higher performance and constant execution time, both of which are ideal for real-time systems. Secondly, stateless operation means that the system is scalable to any number of flows by means of parallel operation. For example, several front-end units can operate independently of each other, and at the same time, by comprising a distributed traffic monitoring system, provide information about all observed events in orchestrated manner. Consequently, stateless operation is an important objective for all the schemes studied in this paper.

2.2 Revocation of keys

A typical problem with secret sharing schemes is when one or few shadows accidentally leak out. In this case one wants to revoke the exposed keys and replace them with new ones, which translates to a new secret polynomial $P^*(x)$ [8]. Formally,

$$(m, n) \xrightarrow{k} (m^*, n^*),$$

where k denotes the number of shadows kept the same. For obvious reasons,

$$k + 1 < \min\{m, m^*\}.$$

Table 1: Shamir's secret sharing scheme [5].

Shamir's scheme:

- Choose $m \leq n < p$, where m is threshold, n is number of shadows, and p is some large prime.
- Set $a_0 = s$ (secret) and pick in random $a_i \in [0, p-1]$, $i = 1, \ldots, m-1$, giving
$$P(x) = a_0 + a_1 x + \ldots + a_{m-1} x^{m-1} \mod p.$$
- Distribute the shadows: $\{i, P(i)\}$, $i = 1, \ldots, n$.

Given k shadows (or points) and the secret, $P(0) = S$, the new polynomial has $m^* - (k+1)$ degrees of freedom, which allows one to pick a new secret polynomial $P^*(x)$ and the remaining $n^* - k$ shadows.

A practical problem is the fact that the coefficients of the $P^*(x)$ cannot be chosen independently in random, but instead the existing k points set constraints. To this end, one can first choose $m^* - k - 1$ new points in random, so that in total m^* points are fixed. Then, by using the Lagrange's interpolation polynomial (1), one can determine the coefficients of the new polynomial and also the remaining $n^* - m^*$ new shadows. Note that the described change to a new polynomial does not jeopardise the security of the scheme.

Moreover, replacing the current polynomial with a new one that has a higher degree allows one to "push" the disclosure threshold higher while keeping k, $k < m$, already disclosed points the same. In other words, this operation can be seen equivalent to taking a negative step with regards to disclosure process of Scheme 1. Other possible means to approximate a negative step are:

- To reset the polynomial, which is lightweight approximative method to achieve similar behavior,
- At step k (after $k < m$ shadows has been revealed), first to reduce the threshold to k, which gives wrong secret a_0^*. Then define a new secret polynomial P' with threshold $m - k + h$ in such a way that the original secret a_0 is $a_0^* + a_0' \mod p$.

Neither is really elegant for our purposes. In the next section, we provide several solutions which automatically revoke the expired shadows as the time goes by.

3. SLIDING WINDOW MECHANISM

The so-called sliding window is a very popular transmission control mechanism in communication engineering [9], which is also utilized by TCP. The essential principle of this mechanism is that the number of non-acknowledged packets during any time interval $(t_0, t_0 + \Delta t)$ should be at most m, where m is some constant, Δt denotes the length of the time window, and t_0 is an arbitrary time offset. If the number of non-acknowledged packets is greater than m, then the transmitter has violated the transmission policy by sending too many packets.

Conversely, the sliding window criterion can be also applied as a threshold. That is, we say that m or more events during a time interval of length W should trigger an alarm. In discrete time this means that during W consecutive time slots m or more events have occurred. In terms of queueing theory, this is equivalent to considering the time to the first blocked customer in $G/D/n/n$ system with general interarrival times, a constant service time of $D = W$, and $n = m - 1$ servers and no waiting places.[1] The actual question is if a "customer" would be blocked during the duration of a flow or not.

Coming back to the key escrow scheme, let us pose the question of how a key escrow process with a sliding window type of expiration can be realised. More specifically, we are interested in devising a scheme in which one of the conditions given above leads to the disclosure of a secret. At the same time, shadows disclosed W or more time units earlier provide no information, i.e., past shadows have been revoked. I.e., when a given threshold is not exceeded, the decryption key remains unknown and the corresponding data self-destructs automatically.

Note that even though privacy aware traffic anomaly detection systems represent the main background of this work, the concept of key escrow schemes with expiring "bits of information" itself is general. Hence, the latter part of this paper in fact applies also to the general problem. Unless explicitly stated otherwise, in the following we always consider a single flow f, where f denotes a flow identifier (e.g., a five tuple, source IP address, etc.).

3.1 Continuous time mechanism

Let us start with the continuous time case, which leads to the most computationally intensive solution. The threshold for secret disclosure at time $t + W$ is

$$A(t, t+W) \geq m,$$

where $A(t_0, t_1)$ denotes the number of events during time interval (t_0, t_1). This can be achieved by modifying Shamir's polynomial $P(x)$ in a dynamic fashion as follows:

[1] Another similar mechanism is the token bucket, which corresponds to a $G/D/1/n$ queue: single server and $n = m - 2$ waiting places.

Scheme 2: (Exact revocation of shadows)
Initially pick a random polynomial $P(x)$, and record all the disclosed points:

$$(t, x_i, P(x_i)) \to \mathcal{P}.$$

Then, upon observing an event at time t:

1. *If $|\mathcal{P}| \geq m$ then STOP. (S already disclosed)*
2. *Clear shadows older than W time units from $\mathcal{P}$.*
3. *If no point got removed, then keep old $P(x)$. Otherwise, $|\mathcal{P}| \leq m-2$, choose in random $m-1-|\mathcal{P}|$ new points, and (1) yields new $P(x)$.*
4. *Disclose a shadow $(t, x, P(x))$ with x random.*

Clearly, the revocation of shadows disclosed W or more time slots earlier works and they provide no additional information for computing $P(0)$ in the current state, except for the case the the secret has already been disclosed earlier. Note that the communication costs are the same as with the basic scheme, i.e., each event triggers a single point $(t, x, P(x))$ to be disclosed. However, this scheme involves more computational efforts at the source, and in particular, the operation is stateful (cf. Def. 1) and does not scale well with the increasing number of flows. For a discrete time version, one can use W Bloom filters [10, 11] to keep track of the disclosed points, which mitigates the problem of statefulness to some degree.

3.2 Discrete time mechanism

Let us next consider discrete time, where each time interval corresponds to one time slot. With $W = 1$, the criterion applies to each time slot individually. This can be accomplished simply by picking a new Shamir polynomial for each time slot, which allows an arbitrary intra time slot threshold of m events. For $W > 1$, the straightforward way is to use multiple concurrent Shamir polynomials:

Scheme 3: (Concurrent Shamir polynomials)

1. *For each time slot j, choose a new polynomial, $P_j(x)$, which is used for W time slots.*
2. *Consequently, there are W concurrently active polynomials during each time slot (see Fig. 2).*
3. *Upon observing an event at time t, disclose one share of each polynomial,*

   $$\{j,\, P_j(x),\, \ldots,\, P_{j+W-1}(x)\},$$

   where $j = \lfloor t - W + 1 \rfloor$ and $x \sim \mathrm{U}[0, p-1]$.

It is easy to see that this construction achieves the goal and works with any $W \geq 1$. Moreover, Scheme 3 is stateless in the sense of Def. 1, as one can define the coefficients using

$$a_i^{(j)} = h_i(s^*, f, j), \quad i = 1, \ldots, m-1,$$

where j corresponds to the jth polynomial, and h_i is a suitable pseudorandom function. However, the communication cost is a linear function of W, and thus the scheme becomes too tedious when W is large.

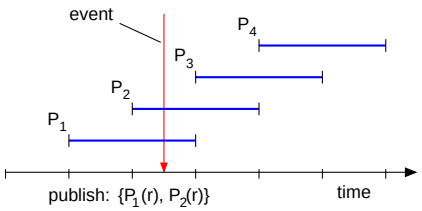

Figure 2: Using multiple polynomials allows sliding window type of threshold, $W = 2$.

3.3 Accuracy performance analysis

Let us next demonstrate the performance improvement with regards to the *accuracy* in the disclosure that the use of concurrent polynomials yields over a single polynomial. For simplicity, we consider the cases of $W = 1$ (single polynomial) and $W = 2$ (two concurrent polynomials), as illustrated in Fig. 2.

As the background lies in privacy preserving traffic monitoring, we do not allow false positives, i.e., no input pattern with less than m events during a given time window may result in the disclosure of a secret. Both schemes obviously achieve this as they both "monitor" only some of the possible time intervals iof a given length.

In order to carry out the following analysis, we define *a test input sequence* which should be detected. To this end, we consider a situation where $M \geq m$ events occur during a time interval equal to W time slots and refer to this time window as *event window*. The offset is assumed to be random, i.e., the event window starts at a random point in some time slot. We note that all M events occur during a time interval of $W + 1$ time slots, i.e., during the time interval two consecutive polynomials cover. At most $m - 1$ events can be "assigned" to each of these without disclosing the secret. Thus, when $M > 2m - 2$ the secret will always be disclosed independently of the chosen W, i.e., the detection probability increases from 0 to 1 as the number of events M increases from $m - 1$ to $2m - 1$. With $W \to \infty$, it just becomes highly unlikely that m or more events do occur in such a way that the secret remains undisclosed.

With $W = 1$ the event window overlaps with two consecutive time slots. The system discloses the secret if m out of M events occur during the first time slot, $N_1 \geq m$, or during the second time slot, $N_2 \geq m$, where $N_1 + N_2 = M$. Assuming further that the M events are independently and uniformly distributed within the event window, we have

$$N_1 \sim \mathrm{Bin}(M, p) \quad \text{and} \quad N_1 + N_2 = M,$$

where p corresponds to the random offset between the first time slot and the event window, $0 \leq p \leq 1$. Consecutively, let $\mathrm{P}\{D\} = \mathrm{P}\{\text{disclosure}\}$ and for $m \leq M \leq 2m-2$ we have

$$\mathrm{P}\{D \,|\, p\} = \sum_{i=0}^{M-m} \mathrm{P}\{N_1 = i\} + \sum_{i=m}^{M} \mathrm{P}\{N_1 = i\},$$

where the first sum corresponds to cases when $N_2 \geq m$ and the second sum to cases when $N_1 \geq m$. As p is uniformly

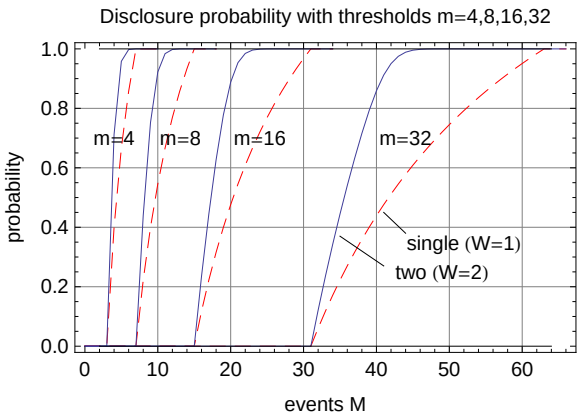

Figure 3: Two concurrent polynomials ($W = 2$ in Scheme 3) improve the disclosure accuracy considerably on average.

distributed to $(0, 1)$ the above gives

$$\mathrm{P}\{D\} = \int_0^1 \mathrm{P}\{D \,|\, p\}\, dp = \frac{2(M+1-m)}{M+1},$$

when $m \le M \le 2m-1$. Obviously, $\mathrm{P}\{D\} = 0$ for $M < m$ and $\mathrm{P}\{D\} = 1$ for $M \ge 2m-1$.

For $W = 2$, the situation is similar. In this case, the M events fall into $W + 1 = 3$ time slots out of which the middle one is common to both polynomials. Hence, instead of binomial distribution, here we have a trinomial distribution for N_1, N_2 and N_3, and the secret is disclosed when either $N_1 + N_2 \ge m$ or $N_2 + N_3 \ge m$.

Fig. 3 illustrates the resulting disclosure probabilities as a function of M with thresholds $m = 4, 8, 16, 32$ and $W = 1, 2$ concurrent polynomials. For each value of m, the left solid curve corresponds to $W = 2$ concurrent polynomials, and the right dotted curve to $W = 1$ concurrent polynomial. We observe that $W = 2$ indeed provides a significantly better and more prompt response on average especially with larger threshold values of m. Note that in an ideal case the response would be an impulse function at $M = m$ corresponding to Scheme 2.

3.4 Dynamic polynomials

It is also possible to do without W concurrent polynomials and instead modify the polynomial in a dynamic fashion, i.e., define a sequence of polynomials denoted by $P_j(x) = P_j(x, f)$ where f denotes the flow identifier. The idea is that two consecutive polynomials $P_j(x)$ and $P_{j+1}(x)$ can share, in a certain way, some of the shadows. The time between two polynomials P_j and P_{j+1} is referred to as *time epoch*, which can consist of one or more time slots. Then, let W denote the window size, i.e., the number of shadows, out of which d are revoked and w are preserved at the end of each time epoch. Thus, $W = w + d$, and

$$(P_j(0), \overbrace{P_j(1), .., P_j(d)}^{\text{revoked}}, \overbrace{P_j(d+1), .., P_j(W)}^{\text{preserved}}) \to P_{j+1}.$$

A polynomial of degree $(m-1)$ can be defined by specifying points $P(0), \ldots, P(m-1)$ (note that here $P_j(0) = S_f$).

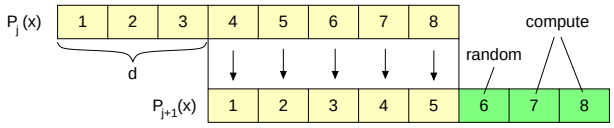

Figure 4: Sliding window scheme realised by Shamir's scheme with discrete time: $W = 8$ and $d = 3$, threshold $m = 7$ allows $w = 5$ common points between two consecutive polynomials.

Consequently, as for each epoch we wish to have a new random polynomial, we can keep at most $m-2$ points, i.e., $w \le m-2$, and $W = w + d \le m + d - 2$. Additionally, one can also require that $2d \le W$, i.e., that all shadows remain within the window at least for two time epochs. These two together yield

$$2d \le W \le m + d - 2 \quad \Leftrightarrow \quad d \le w \le m - 2 \qquad (3)$$

which sets bounds for a practical window size W as a function of m and d

When $W < m$ one cannot compute the secret even if a full window of W shadows is available. However, one can always disclose shadows $P_j(x)$ with $x > W$, which then expire immediately after the current epoch (see Scheme 5). Optional condition $W \ge m$ together with (3) means that $d \ge 2$, and (3) becomes

$$\max\{m, 2d\} \le W \le m + d - 2.$$

For the maximum slack, $\max(W - m)$, one obtains the following diagram:

	d=2	d=3	d=4	d=5	d=6
m=4	0				
m=5	0	0			
m=6	0	1	0		
m=7	0	1	1	0	
m=8	0	1	2	1	0
m=9	0	1	2	2	1
m=10	0	1	2	3	2
⋮			⋮		

The above diagram reveals the unavoidable fact that longer time windows than m are not possible unless the number of revoked shadows d is rather large. Next we introduce two schemes that define such a sequence of polynomials, $P_j(x)$. Later on, we will discuss how these polynomials can actually be used to disclose shadows in response to the observed events.

Let $r_i = r_i(f)$ denote a (pseudo) random sequence for flow f. The first scheme is rather straightforward:

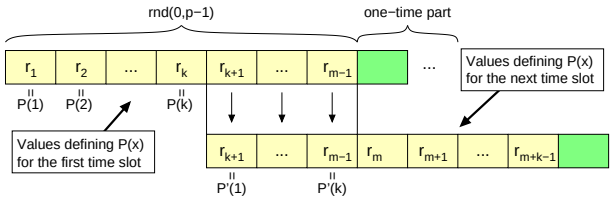

Figure 5: Sliding window type of scheme, which enables stateless operation as the part shared between consecutive polynomials comprises random points.

Scheme 4: (Dynamic polynomial)
Initially, define $P_1(x)$ *by*

$$P_1(0)=S_f, \text{ and } P_1(i)=r_i,\ i=1,\ldots,m-1,$$

and the following $P_{j+1}(x)$ *recursively using*

$$\begin{aligned} &P_{j+1}(0) = S_f, \\ &P_{j+1}(x) = P_j(x+d), \quad x=1,\ldots,w, \\ &P_{j+1}(x) = r_{j(m-w-1)+x},\ x=w+1,\ldots,m-1. \end{aligned} \tag{4}$$

By using (1)*, the above* m *points define* $P_j(x)$, $\forall\, j$*, and one can compute the remaining shadows with* $x = m,\ldots,W$ *(also for* $x > W$ *if needed).*

The scheme is illustrated in Fig. 4 for $d = 3$, $m = 7$ and $W = 8$, i.e., the criterion is 7 out of 8 consecutive time slots. Note that the basis polynomials (2) are constant here, i.e., the step computing the next $P_{j+1}(x)$ can be implemented efficiently, e.g., in hardware. Unfortunately, Scheme 4 is not stateless in the sense of Def. 1, as (4) results in a recursion which ends only at $P_1(x)$.

Another similar scheme is depicted in Fig. 5, where window size $W = m - 1$. Consequently, even the knowledge of all W shadows is insufficient in order to decrypt the secret, and additional one-time shadows with $x > W$ are required. The gain is that the next polynomial $P_{j+1}(x)$ can be defined in such a way that it depends only on the (pseudo) random numbers r_i:

Scheme 5: (Stateless dynamic polynomial)
Using (1)*, define* $P_j(x)$, $j = 0, 1, \ldots$*, by the points*

$$\begin{aligned} &P_j(0) = S_f \quad \textit{(flow specific secret)}, \\ &P_j(i) = r_{d\cdot j+i},\ i=1,\ldots,m-1. \end{aligned}$$

The pseudorandom sequence $r_i = r_i(f)$ defines directly the $P_j(x)$, and thus this scheme is stateless in the sense of Def. 1. Note that $d = 1$ is generally the best choice, unless the computation time of each $P_j(x)$ per time epoch has to be taken into account.

Note that with both schemes a shadow $(x, P_j(x))$ expires at time which depends on both x and j. In particular, we have three types of shadows:

i) Shadows with $x = 1,\ldots,d$ that may have been already disclosed during the previous time slots, and will expire at the end of the current time slot,

ii) Shadows with $x = d+1,\ldots,W$ that are valid for

$$1 + \left\lfloor \frac{x-1}{d} \right\rfloor \text{ time slots,}$$

iii) *One-time shadows* with $x > W$ that are valid only for the current time epoch (cf. Scheme 5).

Hence, there is an incentive to disclose the latest possible preserved shadow, $x = W$. However, disclosing the same shadow twice is not useful either. In general, with randomly chosen x, one should not disclose type i) shadows with $x = 1,\ldots,d$ at all, because they are less useful than the type iii) one-time shadows in the sense that one-time shadows have a smaller probability of having been disclosed earlier on, and both expire at the end of the current epoch.

Finally, we note that changing the polynomial dynamically is not straightforward as it introduces an additional dimension to the disclosure process corresponding to the freedom to choose between the preserved shares and one-time shadows. This complicates both the analysis and the actual use of this type of approach. A more detailed performance analysis study is thus a topic for future research.

3.5 Summary of the schemes

All schemes are summarised in Table 2. As already mentioned, the state information per flow, e.g., coefficients of the polynomial $P(x)$, can often be derived using appropriate pseudorandom functions having time t, the flow identifier f, and a common (secret) salt s^* as input parameters. In this case, the front-end can compute each shadow efficiently on the spot based solely on the triple (f, t, s^*). The last column indicates whether the stateless operation is possible. For clarity, the flow identifier f varies case by case, but as an example, it can stand for the source IP address. The computational burden column indicates the complexity of the scheme with respect to the front-end.

4. CONCLUSIONS

We have considered several schemes which, by cryptographical means, allow one to escrow a secret key when certain events have emerged. Our motivation for such a scheme is the model of a two-stage anomaly detection system where the role of the first stage is to protect the users' privacy. In order to keep the first stage operation scalable and stateless, each detected suspicious event triggers a piece of information, a shadow, to be disclosed. Given a sufficient number of shadows, the second stage can compute the secret key, which allows it to carry out further analysis. The actual information disclosed by the secret key is implementation specific detail.

The main focus of our work has been set on key escrow mechanisms with expiring shadows which ensure self-destruction of the corresponding data. The considered threshold schemes allow one to define the number of events that must occur during W time units in order for the secret key, and the corresponding data, to be disclosed. We have given several solutions to this end, which differ in terms of flexibility, computational complexity and the number of internal states. The common property is that the bits of information disclosed

Table 2: Summary of the key escrowing schemes.

	time horizon	time	communication per event	computational burden	states per flow	pseudorandom operation
Scheme 1	infinite	continuous	1 shadow	light		yes
Scheme 2	sliding window	continuous	1 shadow	heavy	active shadows	no
Scheme 3	sliding window	discrete	W shadows	medium		yes
Scheme 4	sliding window	discrete	1 shadow	heavy	m	no
Scheme 5	sliding window	discrete	1 shadow	heavy		yes

earlier than W time units ago expire and self-destruct. This is achieved by dynamically changing Shamir's secret polynomial(s).

This type of key escrowing mechanisms is useful for the described privacy aware anomaly detection system, and it also bears great potential for further types of traffic monitoring activities. When such a scheme is used as a part of traffic anomaly detection system for escrowing a secret flow-specific decryption key, the scalability requirements imply that the scheme must be able to provide a new shadow on spot given the flow identifier, time, and a random salt value. In particular, no flow-specific states should be kept. In this paper, we have proposed a number of such stateless schemes (see Table 2).

Statefulness is a very important factor for our two-stage traffic monitoring system. In a more general framework, when several front-end units co-operating and have limited computational resources, a stateless key escrow scheme allows one to reveal decryption keys by providing the shadows in distributed fashion, thus achieving a high scalability and preserving the privacy of the users at the same time. Future work includes the investigation of other means to enforce the users' privacy in such systems.

Acknowledgements

This work was done within the scope of the EU FP7 project PRISM (grant no. 215350) and partially supported by the strategic project N0 at FTW.

5. REFERENCES

[1] G. Bianchi, E. Boschi, D. Kaklamani, E. Koutsoloukas, G. Lioudakis, F. Oppedisano, M. Petraschek, F. Ricciato, and C. Schmoll, "Towards privacy-preserving network monitoring: Issues and challenges," in *Proc. of the PIMRC 2007*, Sep. 2007.

[2] G. Bianchi, S. Teofili, and M. Pomposini, "New directions in privacy-preserving anomaly detection for network traffic," in *1st ACM Workshop on Network Data Anonymization (NDA 2008)*, Oct. 2008.

[3] C. Estan, G. Varghese, "New directions in traffic measurements and accounting", in *ACM SIGCOMM 2002*, Pittsburgh, USA, Aug. 2002.

[4] S. Ramabhadran, G. Varghese, "Efficient Implementation of a Statistics Counter Architecture", in *ACM SIGMETRICS 2003*, San Diego, USA, June 2003.

[5] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[6] E. Hyytiä, "Hybrid Secret Key Escrow Mechanisms as Counters", in *Proc of the 1st International Workshop on Security and Communication Networks (IWSCN)*, May 2009.

[7] R. Geambasu, T. Kohno, A. Levy, and H. M. Levy, "Vanish: Increasing data privacy with self-destructing data," in *Proc. of the 18th USENIX Security Symposium*, 2009.

[8] B. Schneier, *Applied Cryptography; Protocols, Algorithms and Source Code in C*, 2nd ed. John Wiley & Sons, 1996.

[9] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Prentice-Hall, 1992.

[10] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[11] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2005.