Cooperation Enablement for Centralistic Early Warning Systems

Ulrich Flegel
SAP Research CEC Karlsruhe
Vincenz-Prießnitz-Str. 1
76131 Karlsruhe, Germany
ulrich.flegel@sap.com

Johannes Hoffmann
TU Dortmund
Computer Science VI
44221 Dortmund, Germany
johannes.hoffmann@udo.edu

Michael Meier
TU Dortmund
Computer Science VI
44221 Dortmund, Germany
michael.meier@udo.edu

ABSTRACT

From large-scale acquisition of information on security incidents by early warning systems (EWS) arises the opportunity to draw up a situation picture that allows detection of trends and upcoming threats. While the need for integrating such information is widely accepted, there typically exist reservations concerning the distribution of information allowing outsiders insights into security incidents of individual organizations. These reservations so far prohibit the deployment of EWS in practice. In order to make EWS practical we study the conflicting interests of all involved parties regarding information processed by the EWS, and propose a resolution of the conflict based on information reduction by pseudonymization. We develop a fair balanced trade-off respecting most interests of parties as well as privacy of involved persons and propose privacy mechanisms to be applied to respective information. An implementation of the privacy mechanisms is experimentally evaluated to demonstrate the practicality of our approach.

Keywords

Information reduction, pseudonymization, early warning system

1. INTRODUCTION

Preventive security measures are more and more complemented with reactive safeguards in order to achieve IT security. Precondition to reaction on security incidents is a timely and dependable detection of respective situations. A cooperative information exchange between different organizations is not only advantageous, but also mandatory for detecting distributed and coordinated attacks. From large-scale acquisition of pertinent information by an *early warning system* (EWS) arises the opportunity to draw up the situation picture that allows the detection of trends and upcoming threats, thereby allowing to take appropriate reactive measures. The need for integrating data in order to construct such a situation picture is widely accepted (cf. e.g., [8, 5, 6, 12]). However, typically there exist reservations concerning the distribution of information allowing outsiders insights into security incidents of individual organizations. These reservations are opposing the integration of information and so far prohibit the creation of a situation picture. A practical EWS needs to take the conflicting interests of the involved parties into consideration. A resolution of the conflict can be achieved by using information reductions, e.g., pseudonymization. The particular cooperation scenario that is considered here is the centralistic malware early warning system developed in the *AMSEL* project [3]. The basic idea of this system is that parties cooperate and exchange information on malware occurred in their domain in order to provide timely information on spreading new malware. Further detection patterns (signatures) for detecting new malware are generated automatically and distributed to participating parties to allow detection of new malware. Detection results are reported to a central domain to allow construction of a situation picture which supports evaluation of malware distribution and early warning on malware epidemics.

On the one hand organizations have a clear interest in participating in the EWS due to the provided protection against new malware threats. On the other hand organizations have to expose information on malware incidents inside their domain, which may contain personal or otherwise confidential information that may be misused by other (potentially competing) organizations. Therefore typically organizations abstain from participating in such an EWS.

To solve the conflict and to enable the cooperative information exchange, information needs to be reduced in a way that still supports the cooperative goal while avoiding misuse of information. This conflict can be broken down to the fundamental conflict between confidentiality requirements (keep malware incidents secret) and availability requirements (provide information on new malware).

The main contributions of this paper are as follows: After giving an overview of the EWS architecture and deployment scenario in Sect. 2 we study the interests of involved parties regarding information processed by the EWS and identify privacy and confidentiality requirements in Sect. 3. Further we develop a fair balanced trade-off respecting most interests of parties and privacy of involved persons and introduce

SAC'10 March 22-26, 2010, Sierre, Switzerland.

classes of privacy mechanisms to be applied to respective EWS information. In Sect. 4 we describe information reductions supporting the balanced interests and propose privacy mechanisms for each identified class. We describe an implementation of the privacy mechanisms as well as results of an experimental performance evaluation of the implementation in Sect. 5 and demonstrate that our approach is usable in practice.

2. THE EARLY WARNING SYSTEM

The architecture of our EWS is shown in Fig. 1. It comprises the four basic components *collecting and learning box*, *threat repository*, *detecting and alerting box* as well as an *alert repository*, which are summarized here. A more detailed description of the EWS and its components can be found elsewhere [3]. The *collecting and learning (CL) box* bundles the functional components to collect malware, to analyze malware to extract features used for learning and to generate appropriate detection signatures.[1] Honeypots like *Nepenthes* [4] are used as malware collectors and a dynamic analysis of collected malware samples is realized by executing the samples inside *CWSandbox* [14]. The *threat repository* is used to centrally manage information on malware and detection criteria delivered by CL boxes. It supplies the information needed for detecting malware to the *detecting and alerting (DA) boxes*, which contain the functional components to detect respective security violations and to generate alerts. Alerts generated by the DA boxes as well as information on malware supplied by the threat repository form the basis for constructing a *situation picture* and they are centrally managed in the *alert repository*. The overall procedure allows to automatically and timely match signatures for detecting threats.

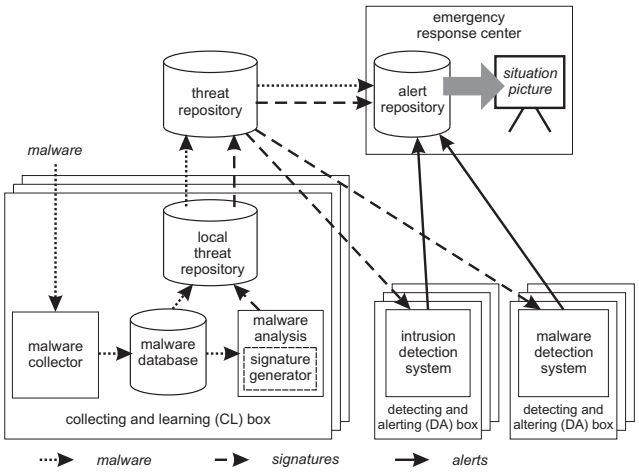

Figure 1: Architecture of the early warning system

Both, the protection level achieved for the participating systems as well as the quality of the situation picture to be constructed from alert repository data, depend on the number and placement of deployed CL boxes as well as DA boxes. The more CL boxes are deployed at suitable locations in the network, the higher will be the likelihood that new malware is collected during an early stage of its distribution and signatures for detection systems are supplied by the threat repository early enough to observe, detect and restrict further distribution of the malware. The more DA boxes are suitably placed in the network the more comprehensive the information base for constructing the situation picture will be. For these reasons, for a productive deployment of the early warning system, the installation of a larger number of CL boxes as well as DA boxes in different network domains is anticipated as sketched in Fig. 2.

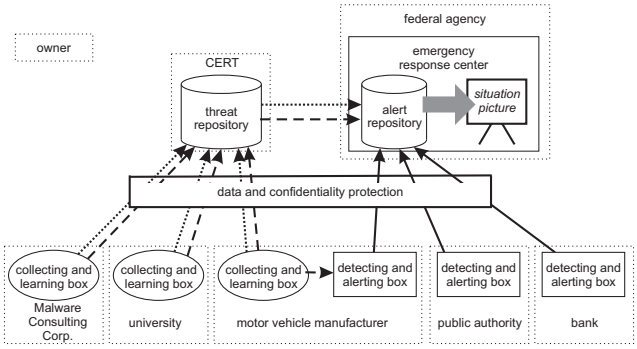

Figure 2: Domains of an example deployment scenario

A primary prerequisite is that the domain owners agree to cooperatively exchange information on security incidents. Without additional protective measures, there exists, e.g., for the owner of the threat repository the possibility to gain knowledge about the occurrence of security incidents in the domains of CL boxes. Analogously, the owner of the emergency response center gains insight into security incidents detected in the domains of DA boxes. Since this may conflict with the interests of the domain owners, an agreement about exchanging the required information is hard to achieve. To resolve this problem and to rebut respective reservations, we investigate requirements on data and privacy protection for required information flows (Sect. 3) and consider complying information reductions, e.g., pseudonymization (cf. 'data and confidentially protection' in Fig. 2). In particular, such information reductions effectively confine owners of central components to acquire sensitive information, without affecting the functionality of the system or significantly reducing the quality of the situation picture. In the following we refine the deployment scenario (cf. Fig. 2) by discussing the suitable parties for operating the system components and elaborating on their interests.

For the current setting, we expect that the *alert repository* is operated by a government agency, such as the *German Federal Office for Information Security (BSI)*, which is officially commissioned to provide a situation picture. The agency might also outsource the operation to a *computer emergency response team (CERT)* or a private sector enterprise. Alternative models would be a private company providing the situation picture as a payable service, or a closed consortium of the EWS member organizations.

The *threat repository* may also be operated by a government agency, a CERT, a private sector company or a consortium of EWS member organizations. If the operating organizations of the alert and the threat repository differ, suitable combinations of operating organizations are subject

[1] In order to counter prevalent malware polymorphism *behavioral* detection signatures are used by the EWS.

to further investigation. Organizations that already operate an *intrusion detection system* (IDS) are natural candidates for member organizations of an EWS. The primary driving force behind installing a *DA box* is the organization's interest in the information provided by the situation picture and its value for safeguarding its assets. All sectors that rely on IT services are promising candidates for operating DA boxes, i.e., government agencies, academia and private sector organizations. For the *CL boxes*, the same consideration as before applies. In particular, a service provider may be specialized on providing a broad and representative collection of malware samples. While it is perfectly possible that an organization specializes either on collecting malware and generating signatures, or on detecting incidents and providing alerts, a combined operation of DA and of CL boxes is valuable. Operating both kinds of boxes allows the organization to generate new signatures to cater to its own need for detecting locally occurring malware and thwarting its recurring occurrence.

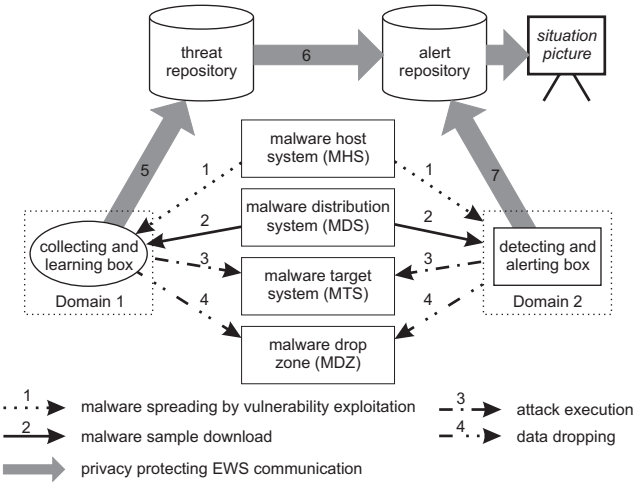

Figure 3: Flow of personal and confidential data

3. PRIVACY AND CONFIDENTIALITY REQUIREMENTS

We first consider entities and their potential relation to personal data. Malware samples may contain (hidden) clues about their authors and endpoint[2] addresses of *malware target systems* (MTS), which are targeted *victim systems* (VS) of attacks executed by malware (e.g. a denial of service attack) and *malware drop zones* (MDZ), where data copied from VS is uploaded. While MDZ may be operated by malware authors, the copied data usually is rather delivered to VS that are controlled by malware authors.

Malware spreads from so-called *malware host systems* (MHS), and a CL box can observe the sending MHS endpoint of a malware trying to spread to the CL box. From the observed exploit payload the CL box can extract the *malware distribution system* (MDS) download endpoint from where the exploit payload is intended to download the malware for execution on the VS. In the vast majority of cases the MHS and MDS also are VS.

VS are often poorly managed home computers operated by natural persons. Since MTS, MDZ, MHS, and MDS may be VS we treat these endpoint data as personal data. Note that we do not intend to protect the interests of malware authors and will therefore ignore them in the following.

Next, we inspect the data flow from malware VS via a CL box to the central threat repository (see flows 1 to 5 in Fig. 3). Features are extracted from flows 1 to 4 and forwarded in privacy protecting form in flows 5, 6, and 7. The CL box initially receives the exploit of a given malware and can identify the originating MHS (flow 1). In a second step, the CL box extracts the download endpoint of the (possibly different) MDS where the complete malware sample is situated and finally downloads it (flow 2). As follows, the CL box receives the following data concerning the attacking MHS: sending endpoint (IP, port) and payload (exploit); and from the malware sample serving MDS: download endpoint (IP, port, protocol-specific path) and payload (malware sample). In addition the following data can be determined: receiving endpoint (CL box IP, port), name of the vulnerable service, timestamp, malware sample ID (e.g. RIPEMD hash value). For further processing, the CL box persists the following data to the local threat repository, which then will be sent to the global threat repository (flow 5): sending endpoint, receiving endpoint, download endpoint, name of vulnerable service, malware sample, timestamp, malware sample ID.

Malware analysis systems may extract additional potentially confidential data from malware samples (e.g. endpoint data of MTS, MDS or MDZ) and provide it to the threat repository. CL boxes also provide automatically generated signatures to the threat repository, where a signature detects an observable behavioral pattern of the collected malware sample. The signatures are distributed to the DA boxes for malware detection. DA boxes may use a network interface (*observing endpoint*) for observing network behavior. When a DA box successfully matches a signature to observed behavior it generates an *alert* (see flow 7 in Fig. 3) that primarily[3] contains a timestamp and the name of the signature.

We classify the functionality of the EWS into two broad classes: analysis and reaction. Analyzing the features delivered by the CL and DA boxes means relating events (malware samples, alerts) by means of their features. This requires that the features that are used to relate events to each other be *linkable* (*linkability* requirement, see Sect. 4).

The EWS warns its participants by providing names or port numbers of services that are currently critical, MDS endpoints from where malware is downloaded, notifies VS endpoints that have been infected by malware, etc. To be able to do so and to make the distributed information actionable for the receivers, the information can be provided in transformed form, e.g., encrypted, but the receivers must be able to *disclose* the original data (*disclosure* requirement, see Sect. 4). As an example, a DA box may use the MDS download endpoint feature to block outbound malware downloads at the firewall of the domain.

We finally examine the interests of the involved parties regarding their ability to link and disclose personal or other-

[2]Endpoints are communication endpoints that characterize a specific object, e.g. a certain process by its IP address and port number, or a file by its URL. The concept of endpoints is frequently used in this paper and endoints are always explicitly qualified or qualified by the textual context.

[3]Additional alert information may be endpoint data, e.g. of MHS or MDZ.

Table 1: Privacy interests of the involved entities – c: confidentiality of own/domain-local feature; l: linkability of remote feature; d: disclosure of remote feature; blank: party has no interest; I: party's interest is not supported; S: party's interest is fully supported; P: party's interest is supported merely against selected other parties;

feature	used in flow no. in Fig. 3	privacy mechanism class	controlled by attacker	victim			CL box	DA box		threat repository		alert repository	
				c	l	d	c	l	d	l	d	l	d
timestamp	1,5,6	1	Yes				P	S	I			S	S
alert signature name	5,6,7	1	Yes				P	S	I		S	S	S
sending endpoint of MHS	1,5,6,7	1	Yes	I	I	I	P	S	I		S	S	S
receiving endpoint of MTS	3,5,6,7	1	Yes	I	I	I	P	S	I		S	S	S
download endpoint of MDS	1,2,5,6,7	1'	Yes	P	I	I	I	S	S		S	S	S
upload endpoint of MDZ	4,5,6,7	1'	Yes	P	I	I	I	S	S		S	S	S
vulnerability module name	1,5,6	1'	Yes				I	S	S		S	S	S
receiving endpoint of CL box	1,5,6	2					S	I	I		S	S	
observing endpoint of DA box	7	2					S	I	I		S	S	
malware exploit payload	1,5,6	3	Yes				I	S	S		S	S	
malware sample payload	2,5,6	3	Yes	P	I	I	I	S	S		S	S	

wise confidential data from the data flows considered above.

Malware VS are usually involved when activity from MHS is recorded (exploit phase, flow 1), when a malware sample is downloaded from MDS (flow 2), when malware attacks MTS (flow 3), as well as when malware copies data from VS to MDZ (cf. flows 1 to 4 in Fig. 3). Addresses of MDS, MTS and MDZ may be hardcoded in the malware sample. Since we consider VS to be identified with the operating persons, the related data is personal data. These persons are assumed to be interested in protecting their privacy and as such are not interested in the disclosure and linkability of their personal data. They however are interested in the disclosure and linkability of the personal data of other VS attacking them, to be able to defend their assets or claim compensation.

CL boxes, as well as DA boxes are assumed to be operated by organizations, not by natural persons. We also assume that the boxes are provided as stand-alone systems that have no users, except for initial administrative purposes. As a result we do not need to consider personal data regarding these boxes. It remains to consider the remaining data that is exchanged with the repositories.[4] This data can be considered sensitive when observed by customers or competitors, such that the organization is interested in keeping them confidential. At the same time a given organization may be interested in the malware incident data of other organizations to gain a competitive edge.

The threat repository and the alert repository merely collect data from the CL boxes and DA boxes, respectively. The data in the threat repository may be refined and enriched manually, but it will still not refer to the operating organization. Hence, we do not need to consider personal data of the operating organizations here. The role of the global threat repository is providing information to DA boxes for detecting and possibly directly blocking malware. For the purpose of transitory blacklisting sites involved in the outbreak of a given malware, the threat repository needs to disclose[5] the download endpoints of MDS, as well as the upload endpoints of MDZ. Proactively patching or shutting down services would be enabled by disclosing the receiving port numbers of the malware collector. Repository data refinement and analysis for enrichment is enabled by disclosure of the payloads (exploits and malware sample). Duplicate elimination is necessary for efficiency reasons.

The alert repository needs to be able to link all data items to create a situation picture. Some data needs to be disclosed for detailed reporting and advisories for time-critical and transitory blacklisting.

The right-hand part of Table 1 summarizes the potentially conflicting interests of the parties involved in an EWS. Conflicting interests are indicated in rows where at least one party is interested in the confidentiality of its local feature, and at least one remote party is interested in the disclosure or linkability of the given feature. For example, VS want to keep their endpoints confidential and CL boxes also want to keep their existence confidential. However, e.g. DA boxes of other domains are generally interested in disclosing these features. It is therefore necessary to define a suitable balance between the conflicting interests. In some cases it suffices to support a given interest in linkability or disclosure only for the repository owners, in other cases it is necessary to support the given interest also for DA boxes. The proposed balance supports the confidentiality requirements of VS only partially: box owners and repository owners can in most cases link and disclose the VS endpoint. The confidentiality requirements of CL and DA box owners can only be supported for the receiving CL endpoint, the observing DA endpoint and the timestamp feature, for the other features it is necessary to let other DA boxes link and disclose them to allow for a timely response.

4. PRIVACY MECHANISMS

The interests of the parties involved in the setting of an EWS have been made explicit in Sect. 3, as well as proposed trade-offs between conflicting interests. In the following we propose privacy mechanisms to enforce the trade-offs pro-

[4]We assume that data collected and generated locally is provided for local detection purposes in the clear, we ignore this part of the data in Table 1.

[5]In principle blacklisting would only require linkability of endpoint data. Yet, disclosure is necessary, considering state-of-the-art safeguards such as firewalls, which today are unable to transform endpoint data before matching it against their filter rules (see privacy mechanisms class 1').

posed in Table 1. For the mechanisms we had to consider trade-offs between privacy/confidentiality of data to be hidden, availability in the sense of linkability and disclosure of the hidden data, and performance of the mechanisms.

The general rationale guiding the selection of the mechanisms is as follows: DA boxes and alert repositories frequently exploit the linkability of features and need to be able to do it in real-time. We use cryptographic one-way functions $h(f)$ as proposed by Flegel and Biskup for cooperating intrusion detection agents [7]. The actually used implementations of the cryptographic primitives are described and benchmarked in Sect. 5. We call the result $h(f)$ the *pseudonym* of feature f. Two pseudonyms $h(f)$ and $h(f')$ generated by a given CL box for two given features can be directly compared, i.e. $f = f' \Rightarrow h(f) = h(f')$. In order to compare a pseudonym $h(f)$ generated by a CL box with a cleartext feature f' the DA box needs to compute $h(f')$.

Considering state-of-the-art COTS safeguards, such as firewalls, we cannot assume that they support the computation of the pseudonym $h(f')$ for a given feature f' prior to comparing it against a blacklist, which would contain pseudonyms originally computed by CL boxes. Hence, some features need to be disclosed by DA boxes for enforcing blacklists, e.g. blocking network data flows to MDS and MDZ. For features where only few instances are expected, but where a high security benefit is expected for the domains, we leave the features open. We argue that the privacy of a few critical endpoints is sacrificed for the benefit of the community. This is in principle an undesirable solution, the linkability of the features in hidden form is still supported in order to foster the privacy enhancement of COTS safeguards.

All other features are encrypted by the CL boxes. For performance reasons we use a symmetric cryptosystem for the encryption function $e_k(x)$. Privacy and confidentiality of the features are protected by actually using a symmetric threshold cryptosystem, where the key k is shared between operating staff in the threat and alert repository, and a *privacy official*. The key management between the threat repository and the CL boxes is left out as a standard issue here. The decryption operation is not a real-time critical operation and the purpose binding of the decryption is enforced by organizational means, i.e. the operational staff needs to cooperate with the privacy official, who scrutinizes the purpose of decryption on a case-by-case basis. DA boxes may ask the threat repository for ephemeral decryption keys on a case-by-case basis, such that mass disclosure will be detected by the repository staff.

In the actual implemented system we represent a given feature in the threat records generated by the CL boxes by a *linkability tag* (L-tag) containing the pseudonym, and by a *disclosure tag* (D-tag) containing the encrypted feature.

When taking a look at Table 1 we can identify classes of features that have very similar requirements wrt. privacy (VS) and confidentiality (CL boxes), as well as linkability and disclosure. In the following we define these classes and propose the construction of L-tags and D-tags for each class. For defining a class we look for similar requirements and also consider the semantics of the features. For proposing mechanisms we consider the interests that need to be supported within a class, as signified by 'S' markers in Table 1. The threat repository will always contain both, L-tags and D-tags, even if it is not interested in particular tags itself. In the following we treat the threat repository and the alert repository as a *common central authority* (CCA) wrt. tags.

Class 1 comprises features of VS that need to be actionable, e.g. the endpoint data of MHS, MTS, MDS, and MDZ. These features are considered personal data, or in the case of VS within domains, as confidential. From Table 1 we conclude that the features in the class need to be linkable for the DA boxes, as well as for the CCA. Wrt. disclosure the class is further subdivided: being able to recognize MDS and MDZ endpoints with state-of-the-art technology is critical for defending the domains. As described above, the number of MDS and MDZ endpoints is assumed to be low and the privacy/confidentiality interest of these VS is sacrificied by leaving these features open. As soon as COTS safeguards support the L-tags, class 1' should be treated as class 1.

MHS and MTS are less critical for domain-local defense reasons and would also occur in larger numbers, possibly intractable to handle for blacklisting reasons. These features are encrypted using a symmetric threshold cryptosystem, as described above.

L-tag: $\langle h(f) \rangle$; A more advanced approach to pseudonymizing timestamps has been proposed by Kerschbaum [9]. His approach allows for ordering close timestamps and could be used here for timestamps, if required by the generated signatures.

D-tag: $\langle e_k(r|f) \rangle$, r is a random value, $|$ denotes concatenation.

Class 2 comprises features representing resources of the domains, i.e. the DA and CL box endpoints. All of these features may be relevant for correlation by the alert repository, but highly confidential wrt. other parties, in particular other domains. Noteworthy features in this class are CL and DA endpoints. If these endpoints would be linkable or disclosed, malware authors could blacklist them, such that their malware avoids the EWS, but can still attack machines in the domains. Hence, the CL and DA endpoints must be kept confidential in the repositories. The threat repository therefore must not log the CL and DA endpoint address delivering threat or alert information, or these boxes deliver threat or alert information via an anonymous channel such as a mix cascade (cf., e.g., TOR [1]).

The features in this class must only be linkable by the CCA. Also merely the CCA should be able to disclose these features subject to organizational purpose binding.

L-tag: $\langle e_l(r|h(f)) \rangle$, l is an ephemeral session key negotiated between the CL box and the CCA. Note that the alert repository needs to decrypt the L-tag prior to correlating it.

D-tag: $\langle e_k(r|f) \rangle$; see class 1.

Class 3 comprises features containing malware or exploit code payload. These features are relevant for correlation by the alert repository and signature refinement by the threat repository, but confidential wrt. other parties, in particular other domains. Hence, the features in this class are linkable for the domains and the CCA. Merely the CCA should be able to disclose these features subject to organizational purpose binding. Domains may request disclosure of features in exceptional cases from the CCA subject to organizational purpose binding.

L-tag: $\langle h(f)\rangle$; see class 1.

D-tag: $\langle e_o(f), e_k(o)\rangle$, o is a one-time key that the CCA may disclose to the domains.

4.1 Example

For a better illustration we consider an example scenario where a given malware tries to spread at time t_1 from a MHS vs_h to a CL box in domain a with receiving endpoint d_a using a vulnerability characterized by vulnerability module vm. For simpler application we assume that vm is the primary listening port number of the vulnerability module. The malware would then download its code m from a MDS vs_d. The CL box in domain a generates a signature that matches m and includes its signature name sn_1 in the threat information:

feature	class	L-tag	D-tag
t_1	1	$\langle h(t_1)\rangle$	$\langle e_k(r_1\|t_1)\rangle$
sn_1	1	$\langle h(sn_1)\rangle$	$\langle e_k(r_2\|sn_1)\rangle$
vs_h	1	$\langle h(vs_h)\rangle$	$\langle e_k(r_3\|vs_h)\rangle$
vs_d	1'	$\langle h(vs_d)\rangle$	$\langle vs_d\rangle$
vm	1'	$\langle h(vm)\rangle$	$\langle vm\rangle$
d_a	2	$\langle e_{l_1}(r_4\|h(d_a))\rangle$	$\langle e_k(r_5\|d_a)\rangle$
m	3	$\langle h(m)\rangle$	$\langle e_{o_1}(m), e_k(o_1)\rangle$

The CL box sends the threat information to the threat repository, which stores the information and also distributes it to the alert repository, as well as the DA boxes in the domains. For the example we focus on a given DA box with observing endpoint d_b in domain b. When the DA box has received the threat information it may use the feature tags as follows (non-exclusive list):

$\langle h(vs_h)\rangle$**:** detect (and block) traffic to/from the infected MHS (connection endpoint x) by computing $h(x)$ and testing $h(x) = h(vs_h)$[6]

$\langle h(vs_d)\rangle$**:** detect (and block) download of m to machines in the own domain by computing $h(x)$ and testing $h(x) = h(vs_d)$ for outbound connections (external connection endpoint x)[7]

$\langle vs_d\rangle$**:** detect (and block) download of m to machines in the own domain by testing $x = vs_d$ for outbound connections (external connection endpoint x)[7]

$\langle h(vm)\rangle$**:** detect (and block) exploitation of services on port x vulnerable to the vulnerablity characterized by vm by computing $h(x)$ and testing $h(x) = h(vm)$

$\langle vm\rangle$**:** detect (and block) exploitation of services on port x vulnerable to the vulnerablity characterized by vm by testing $x = vm$, remove or secure vulnerable services on port x

$\langle h(m)\rangle$**:** detect (and block) inbound malware m, e.g. also on other infection vectors of m such as email attachments and instant message file transfers, by computing $h(pl)$ and testing $h(pl) = h(m)$ on payload pl at the application layer

[6]We could as well detect (and block) traffic to MTS, if the CL box provides the respective feature.

[7] We could as well detect (and block) traffic to MDZ, if the CL box provides the respective feature.

$\langle e_{o_1}(m), e_k(o_1)\rangle$**:** inspect a given malware m by requesting o_1 from the threat repository and decrypting $e_{o_1}(m)$[8]

Note that due to the lack of e_{l_1} and e_k the domain b cannot link and disclose the domain a in which a threat occurs. It also cannot disclose the timestamp of a threat occurrence, the signature name of threats detected in domain a, as well as the VS from which the malware spreads ($\langle e_{l_1}(r_4|h(d_a))\rangle$, $\langle e_k(r_5|d_a)\rangle$ $\langle e_k(r_1|t_1)\rangle$, $\langle e_k(r_2|sn_1)\rangle$, $\langle e_k(r_3|vs_h)\rangle$).

At time $t_2 > t_1$ the given malware actually tries to spread to domain b and is detected by a DA box with observing endpoint d_b using signature sn_2, subsequently sending an alert to the alert repository, such as d_a has done before, providing the following threat information:

feature	class	L-tag	D-tag
t_2	1	$\langle h(t_2)\rangle$	$\langle e_k(r_6\|t_2)\rangle$
sn_2	1	$\langle h(sn_2)\rangle$	$\langle e_k(r_7\|sn_2)\rangle$
d_b	2	$\langle e_{l_2}(r_9\|h(d_b))\rangle$	$\langle e_k(r_{10}\|d_b)\rangle$

The alert repository may use the features provided by d_a and d_b as follows, e.g. to create a situation picture (non-exclusive list):

$\langle h(t_1)\rangle, \langle h(t_2)\rangle$**:** in order to determine the order of the events of an outbreak of the given malware we would use Kerschbaum's approach [9] to pseudonymize timestamps

$\langle e_k(r_1|t_1)\rangle$**:** determine the time of certain events, such as the first occurrence of the given malware, by decrypting $e_k(r_1|t_1)$

$\langle h(sn_1)\rangle, \langle h(sn_2)\rangle$**:** correlate alerts by signature name to track the spreading of the given malware phenomenon across domains by testing $h(sn_1) = h(sn_2)$

$\langle e_k(r_2|sn_1)\rangle$**:** provide statistics, spread reports and recommendations for the given malware under its name by decrypting $e_k(r_2|sn_1)$

$\langle h(vs_h)\rangle$**:** determine how many distinct MHS the given malware has infected, or how many distinct malware samples the given MHS (endpoint mhs) has spread by testing $h(mhs) = h(vs_h)$[9]

$\langle e_k(r_3|vs_h)\rangle$**:** warn an infected MHS by decrypting $e_k(r_3|vs_h)$ and determining the operating person[9]

$\langle h(vs_d)\rangle$**:** determine how many distinct MDS the given malware uses by testing $h(mds) = h(vs_d)$ for $h(mds)$ pseudonyms in available alerts, or warn the domains about the MDS by distributing $h(vs_d)$[10]

$\langle vs_d\rangle$**:** determine how many distinct MDS a malware uses by testing $mds = vs_d$ for mds features in available alerts, warn the domains about the MDS by distributing vs_d or warn the operating person of the MDS[10]

$\langle h(vm)\rangle$**:** correlate alerts by vulnerability module names to determine the vulnerable population of VS across domains by testing $h(vm_i) = h(vm)$

[8]We could as well inspect exploit code, if the CL box provides the respective feature.

[9] The same would be possible for MTS, if the CL box provides the respective feature.

[10] The same would be possible for MDZ, if the CL box provides the respective feature.

Table 2: Benchmark results; L-tag: average time to create 1000 linkable pseudonyms, D-tag: average time to create 1000 disclosable pseudonyms, Link: average time to link one pseudonym against 1000 pseudonyms, Link/C: average time to link one cleartext feature against 1000 pseudonyms

privacy	AES-128/Ripemd-128				Twofish-192/Tiger-192				feature
mech. class	L-tag	D-tag	Link	Link/C	L-tag	D-tag	Link	Link/C	size
class 1	3.4ms	69.7ms	0.4ms	see Link[a]	3.8ms	118.6ms	0.5ms	see Link[a]	50 byte
class 2	50.8ms	see class 1	30.8ms	15.9ms	106.4ms	see class 1	133.6ms	75.8ms	50 byte
class 3	1768.6ms	7272.6ms	0.6ms	see Link[a]	1950.1ms	7705.9ms	0.4ms	see Link[a]	233 kbyte

[a]This operation only includes the generation of one pseudonym and then link this pseudonym against all the others, which equals the pseudonym-linkability-check.

$\langle vm \rangle$**:** provide statistics, spread reports and recommendations about and to the population vulnerable to vm by testing $vm_i = vm$

$\langle e_{l_1}(r_4|h(d_a)) \rangle, \langle e_{l_2}(r_9|h(d_b)) \rangle$**:** determine statistics on phenomena involving the machines of a given domain by decrypting $e_{l_1}(r_4|h(d_a))$ and $e_{l_2}(r_9|h(d_b))$ and testing $h(d_a) = h(d_b)$

$\langle e_k(r_5|d_a) \rangle, \langle e_k(r_{10}|d_b) \rangle$**:** provide each domain with statistics on phenomena involving the machines of the given domain by decrypting $e_k(r_5|d_a)$ and $e_k(r_{10}|d_b)$, testing $d_a = d_b$ and determining and informing operating personnel of d_a and d_b

$\langle h(m) \rangle$**:** distinguish phenomena in statistics by malware code or duplicate elimination by testing $h(m_i) = h(m)$

$\langle e_{o_1}(m), e_k(o_1) \rangle$**:** inspect a malware m by decrypting $e_{o_1}(m)$ and refine threat information such as signatures[11]

5. MECHANISM IMPLEMENTATION AND PERFORMANCE EVALUATION

In the following we investigate the practicality of the privacy mechanisms proposed in Sect. 4 using an implementation based on JAVA and the *Bouncy Castle* library for cryptographic primitives. Table 2 summarizes the measurements taken on an AMD Athlon X2 4450B with 4GB of memory running 64bit Gentoo Linux and the IcedTea JVM for Java 6. For the benchmark, all operations were performed by a single thread. All feature data was kept in memory; this abstracts from latency introduced by harddrives, network connections etc. The feature data sizes are based on real data observations. Feature classes 1 and 2 almost always represent an endpoint, most likely an URL or an IP address. Based on malware collector data observations 50 bytes are almost always more than needed to accomodate the observed URLs. The amount of data for class 3 is based on the average size of 1000 randomly chosen collected malware samples, which is 233 kbyte. We measured the time to pseudonymize features with two different sets of cryptographic algorithms, AES-128/Ripemd-128 and Twofish-192/Tiger-192. For each privacy mechanism class and both algorithm-sets we determined the runtime duration for generating L- and D-tags, as well as for testing two given pseudonyms for equality and to check whether a pseudonym contains a specific cleartext feature. Since pseudonym disclosure is subject to organizational purpose binding it is not considered realtime critical and is not measured. All pseudonymized features were stored as a byte array in memory for further processing along with used cryptographic keys. For privacy mechanism classes employing a random value, the random value was incorporated as an inline initialization vector to the cipher algorithm. Both ciphers were used in CBC mode along with PKCS7 padding. When linking pseudonyms byte arrays containing the pseudonym values were checked for equality.

All runtimes are averaged over 10 runs, each run using new synthetic feature data.[12] Nevertheless, there are still some fluctuations in speed. The time for generating synthetic data is not included in the given runtimes. Still, the generation of random values for the class mechanism is included in the measured times.

Although we only used two different sets of algorithms, we assume that most modern cryptographic algorithms are suitable for the task of pseudonym generation. The biggest impact on the runtime was determined by the feature size with the highest penalty to the cipher algorithms.

By calculating the time to generate one pseudonym of each privacy mechanism class and multiplying it with the given count for each class from Table 1, we can conclude that our implementation is able to pseudonymize the data of all involved entities at a rate of 49 attacks/s. This amount is for the slower Twofish-192/Tiger-192 algorithm-set and includes the generation of L- and D-tags. We do believe this is suitable for most systems, since this rate exceeds our malware collector capture-rate by far; a malware sample is downloaded every 20 seconds on average.[13] We also performed all tests with all data encapsulated in a XML-structure. With all additional overhead we were still able to generate 17 pseudonyms/s. Further speedup can be gained by using multi-threading or high-performance machines.

6. RELATED WORK

There is related work on balancing confidentiality and availability requirements in particular in the areas of network log anonymization and information reduction for cooperative intrusion detection. Flegel and Biskup discuss requirements on respective information reductions for cooperating intrusion detection agents [7]. Slagell and Yurcik [13] discuss and motivate the distribution and publication of pseudonymized log data on an abstract level without proposing any mechanisms.

An anonymization mechanism that is particularly tailored to IP addresses and their interpretation is proposed by Xu et

[11] The same would be possible for exploit code, if the CL box provides the respective feature.

[12]The first few runs were not measured because the JVM performs poorly at the beginning.

[13]Only successful malware downloads are counted. Including failed attacks we see an attack every two seconds on average. The malware collector covers a complete class B net.

al. [16]. This approach supports correct analysis of prefixes of anonymized IP addresses. Xu and Ning propose an approach aiming to support cooperation of intrusion detection agents [15]. They merely consider information reduction by coarsening features using concept hierarchies, which obviously introduces uncertainty, leading to worse correlation results, which may or may not be acceptable.

Lincoln et al. propose a solution for alarm repositories, which collect alarms from source agents and publish them for further analysis [11]. They analyze, which kinds of surveillance modes are supported by their solution with the result that the supported analysis capabilities are limited.

The generic framework FLAIM for anonymization of network logs is proposed by Slagell and Lakkaraju [2]. These authors provide a tool for network log anonymization that supports different specific anonymization algorithms that can be applied to distinct features of data. However the availability requirements supported by FLAIMS's anonymization algorithms are limited to specific linkability operations of anonymized data.

A privacy-preserving framework for distributed audit that allows member organizations to detect distributed attacks without requiring the release of excessive private information is described by Lee et al. [10]. They consider a setup similar to ours, organizations represent individual security domains, within which no data anonymization is necessary. They discuss data types of log data that are supported by the framework as well as respective algorithms for anonymization of data types. However, only availability requirements of the central auditor are supported. Requirements of foreign domains are not considered. Most of the related work focus on anoymization of network log data, particularly IP adresses. Approaches proposed in related work only consider and support the availability requirement for linkability. In contrast our approach additionally supports the availability requirement for controlled disclosure.

7. CONCLUSIONS

EWS require the cooperative exchange of information on security incidents by different organizations. In order to resolve existing conflicts which prevent organizations from participating in an EWS we elaborated on the specific interests of all involved parties regarding data features processed by the EWS. We developed a fair balanced compromise respecting most interests of parties and privacy of involved persons. Based on that we proposed privacy mechanism classes for features with very similar requirements wrt. privacy and confidentiality as well as availability in terms of linkability and disclosure. Mechanisms for each class were described and implemented. An experimental evaluation of the mechanisms demonstrated that application of our approach is feasible in practice. By removing initially existing conflicts of interests between involved parties our approach enables cooperation of organizations in the EWS.

8. ACKNOWLEDGMENTS

This work has been accomplished in cooperation with the German Federal Office for Information Security (BSI) and the German Federal Ministry of Economics and Technology (BMWi).

9. REFERENCES

[1] Tor: Anonymity online (`http://www.torproject.org/`).

[2] K. L. A. Slagell, K. Lakkaraju. A multi-level anonymization framework for computer and network logs. In *Proc. of USENIX LISA 2006*, pages 63–77. USENIX, 2006.

[3] M. Apel, J. Biskup, U. Flegel, and M. Meier. Towards early warning systems – challenges, technologies and architecture. In *Proc. of CRITIS 2009 (to appear)*, LNCS. Springer, 2009.

[4] P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. Freiling. The Nepenthes platform: An efficient approach to collect malware. In *Proc. of RAID'06*, volume 4219 of *LNCS*, pages 165–184. Springer, 2006.

[5] K. Bsufka, O. Kroll-Peters, and S. Albayrak. Intelligent network-based early warning systems. In *Proc. of CRITIS'06*, volume 4347 of *LNCS*, pages 103–111. Springer, 2006.

[6] DShield. DShield website, (`http//www.dshield.org`), 2008.

[7] U. Flegel and J. Biskup. Requirements of information reductions for cooperating intrusion detection agents. In *Proc. of ETRICS 2006*, volume 3995 of *LNCS*, pages 466–480. Springer, 2006.

[8] B. Grobauer, J. Mehlau, and J. Sander. Carmentis: A co-operative approach towards situation awareness and early warning for the internet. In *Proc. of IMF'06*, pages 55–66, 2006.

[9] F. Kerschbaum. Distance-preserving pseudonymization for timestamps and spatial data. In *Proc. of ACM WPES 2007*, pages 68–71. ACM, 2007.

[10] A. J. Lee, P. Tabriz, and N. Borisov. A privacy-preserving interdomain audit framework. In *Proc. of ACM WPES 2006*, pages 99–108. ACM, 2006.

[11] P. Lincoln, P. A. Porras, and V. Shmatikov. Privacy-preserving sharing and correlation of security alerts. In *Proc. of USENIX Security Symposium 2004*, pages 239–254. USENIX, 2004.

[12] T. E. C. Network. The European CSIRT Network Website (`http://www.ecsirt.net/`), 2008.

[13] A. J. Slagell and W. Yurcik. Sharing computer network logs for security and privacy: A motivation for new methodologies of anonymization. *CoRR*, cs.CR/0409005, 2004.

[14] C. Willems, T. Holz, and F. Freiling. Toward automated dynamic malware analysis using CWSandbox. *IEEE Security & Privacy*, 5(2):32–39, 2007.

[15] D. Xu and P. Ning. Privacy-preserving alert correlation: A concept hierarchy based approach. In *Proc. of ACSAC 2005*, pages 537–546. IEEE Computer Society, 2005.

[16] J. Xu, J. Fan, M. Ammar, and S. Moon. Prefix-preserving ip address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In *Proc. of IEEE ICNP 2002*, pages 280–289. IEEE Computer Society, 2002.