Poster Papers

Dynamic Enforcement of Dynamic Policies

Pablo Buiras and Bart van Delft

Chalmers University of Technology, Sweden

Abstract. LIO is a dynamic information-flow control system embedded in Haskell that uses a runtime monitor to enforce noninterference. The monitor is written as a library, requiring no changes to the runtime. We propose to extend LIO with a state component, allowing us to enforce not only noninterference but also information-flow policies that change while the program is running.

Enforcement mechanisms for information flows in software frequently aim to achieve the *noninterference* security property. This property states that no change in sensitive (secret) inputs to the system should affect non-sensitive (public) outputs, which captures the idea that secrets should not be leaked.

LIO [3] is a Haskell runtime monitor that enforces noninterference. Over time, LIO has been successfully extended to prevent information leaks via certain covert timing channels. The security condition has, however, not yet been generalised, even though it is generally accepted that noninterference is too strong a requirement for most applications.

There are several canonical examples of applications that necessarily violate noninterference. A password checker needs to allow for some interference from the password database to the user to signal whether the login attempt was successful or not. Information purchase applications require noninterference on confidential information to hold only until the price for that information has been paid. Yet other applications might need to introduce additional noninterference constraints over time, for example on the information flow from strategic documents to a manager who is demoted while the system is running.

To allow for the enforcement of such dynamic policies, we propose to extend LIO with a state component which records that part of the system state relevant to determine the current policy that needs to be enforced. In the following we briefly summarise how LIO works and how we propose to extend it. For now we consider only the original sequential LIO library, leaving support for extensions such as concurrency to future work.

Labelled IO. LIO leverages Haskell's monadic encoding of side-effects to provide security. In Haskell, input/output operations are provided by the `IO` *monad*, an abstract data type used to express sequencing of effectful computations. The `LIO` monad provided by the LIO library is intended to be used as a replacement for this type. It provides a collection of operations similar to `IO`, but enriched with security checks that prevent unwanted information flows. `LIO` computations

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 275–276, 2014.
DOI: 10.1007/978-3-319-11599-3,

carry the type `LIO l a`, where `l` is an arbitrary security lattice of labels specified by the code using LIO and `a` is the type of the result of the computation.

The LIO library uses a *floating-label* approach to the dynamic enforcement of information-flow policies, which is based on mandatory access control. The `LIO` monad uses its state to keep track of a *current label*, $L_{\texttt{cur}}$. This label represents, in a coarse-grained way, the least upper bound over the labels on which the current computation depends. All the (I/O) operations provided by `LIO` take care to appropriately validate and adjust this label. Consider a standard two-point lattice (`Low` $\sqsubseteq$ `High`) and a computation starting with $L_{\texttt{cur}}$ being `Low`. When this computation reads a file labelled `High`, $L_{\texttt{cur}}$ is raised to `High` and writing to `Low` files is prohibited by the `LIO` monad from that moment onwards, independent of what would actually be written to these files.

Stateful LIO. We propose for LIO computations to carry the type `LIO s l a`, where `s` is the type of the state component for `LIO` to use in its enforcement. That is, when writing to a file with label l we now check whether $L_{\texttt{cur}} \sqsubseteq_s l$. The `LIO` library exports functionality to update this state s, so the outcome of this check for the same $L_{\texttt{cur}}$ and l can vary depending on the current value of s.

As the relation between labels can now change arbitrarily over time, the labels lose their lattice structure and a least upper bound can no longer be computed. Therefore $L_{\texttt{cur}}$ is modified to contain the *set* of labels of all the information on which the current computation depends. When performing a sensitive operation like writing to a file, the $\sqsubseteq_s$ check is performed for each label in $L_{\texttt{cur}}$ individually.

Encodings. We can now present various policy change mechanisms as restricted interfaces to Stateful LIO. Clearly, we can regain the original noninterference by simply not exporting the operations to update the state component.

We can export an explicit `declassify` function, by using a boolean value as the state component and having the ordering among policies as usual except that `High` $\sqsubseteq_{true}$ `Low` holds but `High` $\sqsubseteq_{false}$ `Low` does not. An operation `p` can now be declassified by calling `declassify p` which sets the state to *true*, performs `p` and then resets the state to *false* before returning.

We can also encode policy languages that allow for much more policy change, such as Paralocks [1] (in which the state component becomes a set of open locks) or non-disclosure policies [2] (where the state tracks the set of flow-relations).

References

1. Broberg, N., Sands, D.: Paralocks – Role-Based Information Flow Control and Beyond. In: Proceedings of the 37th Annual ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, POPL 2010 (2010)
2. Matos, A.A., Boudol, G.: On declassification and the non-disclosure policy. In: 18th IEEE Workshop on Computer Security Foundations, CSFW-18 2005, pp. 226–240. IEEE (2005)
3. Stefan, D., Russo, A., Mitchell, J.C., Mazières, D.: Flexible Dynamic Information Flow Control in Haskell. In: Proceedings of the 4th ACM Symposium on Haskell, Haskell 2011, pp. 95–106. ACM, New York (2011)

Availability by Design

Roberto Vigo, Flemming Nielson, and Hanne Riis Nielson

DTU Compute, Technical University of Denmark, Denmark
{rvig,fnie,hrni}@dtu.dk

Availability is "the property of being accessible and usable upon demand by an authorised entity" [1], and its absence is termed Denial-of-Service (DoS) or unavailability. DoS typically occurs when the resources of a target server are exhausted, preventing a given service to be offered to clients and often leading to the paralysis of an entire system, with a domino effect. Our proposal aims at preventing such effect through a defensive programming style.

Despite availability has received lesser attention than confidentiality and integrity, with which it forms the so-called CIA properties [2], DoS attacks to systems of public concern occur increasingly and have become infamous on the Internet, the distributed system *par excellence*. Besides active attackers, limited resources or optimistic assumptions about the environment can be source of unavailability, suggesting that cryptography is not the ultimate solution.

We claim that the absence of first-class constructs supporting DoS considerations in existing programming frameworks is one determining factor of a great many availability threats. Promoting unavailability to be a first-class citizen of a language not only raises the awareness in developers, but also leads to devise more precise analyses. To this end we proposed the Quality Calculus [3,4], a process calculus in the π calculus family that is equipped with the notion of absence of communication and lacking information.

As availability concerns naturally apply to distributed communicating systems, we find it fruitful to study the problem in process algebraic settings, where DoS is defined as the absence of expected communication and acceptance of improper information. The first trait corresponds to classic network-level DoS (expected data are not received), while the second accounts for the migration of availability attacks up the ISO/OSI stack. Indeed, whether we receive nothing or something we cannot use, the effect is the same.

The main novelty of the Quality Calculus is a binder specifying the inputs to be performed before continuing. In the simplest case it is an input guard $t?x$ describing that some value should be received over the channel t and should be bound to the variable x. Increasing in complexity, we may have binders of the form $\&_q(t_1?x_1, \cdots, t_n?x_n)$ indicating that several inputs are *simultaneously* active and a quality predicate q that determines when a *sufficient* combination of inputs has been received to continue. Moreover, input patterns can be used to specify what sort of data an input is willing to accept, as in $t?x[p]$.

As a consequence, when continuing with the computation some variables might not have obtained proper values, as the corresponding inputs might have not been performed. To model this we distinguish between data and optional data, much like the use of option data types in programming languages like

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 277–278, 2014.
DOI: 10.1007/978-3-319-11599-3,

Standard ML. If c is the message received by an input, then the corresponding variable is bound to $\mathsf{some}(c)$, while in case the input is not received but we continue anyway the variable is bound to none. Whenever accessing an input variable, then, the calculus obliges to inspect its content through the construct $\mathsf{case}\ x\ \mathsf{of}\ \mathsf{some}(y) \colon P_1\ \mathsf{else}\ P_2$, executing P_1 if x carries $\mathsf{some}(c)$ or P_2 otherwise. In this sense, at every point of the computation we know what data we have and what we have not.

The calculus is complemented by a number of verification aids, including two static analyses and an executable specification of the semantics, that facilitate the work of the developer by pointing out where and why DoS might occur, in terms of reachability of program points and combination of values for the inputs. The analyses are implemented as Satisfiability and Satisfiability Modulo Theories problems (the latter not yet published), and thus can exploit the scalability of modern off-the-shelf solvers.

We deem that our investigation leads to a shift in the mind-set. Existing literature on availability zeroes in on mechanisms to detect and avoid DoS attacks on the target side. This is a challenging task, as the detection process itself can be frustrated by the ongoing attack. Moreover, unavailability has a great many sources that are not encompassed by active countermeasures to DoS. We advocate instead for a world in which components are aware of being part of a system and are determined to operate even if their ideal partners become unavailable and do not provide expected information, perhaps because such partners are undergoing a DoS attack. The overall perspective is then lifted from a self-centred, muscular approach to a more realistic view which admits the existence of DoS and tries to circumvent it.

The spirit of our proposal is thus to cope with the effect of DoS through a pessimistic approach to programming. Nonetheless, it is not always possible to follow alternative plans, and therefore ongoing work is focusing on quantitative considerations, to ensure that potential sources of DoS do not impact the behaviour of the system with respect to given contracts. Finally, another line of development concerns studying the portability and effect of our approach to availability on real programming languages.

References

1. ISO: ISO/IEC 7498 - Part 2: Security Architecture
2. Gollmann, D.: Computer Security, 3rd edn. Wiley (2011)
3. Riis Nielson, H., Nielson, F., Vigo, R.: A Calculus for Quality. In: Păsăreanu, C.S., Salaün, G. (eds.) FACS 2012. LNCS, vol. 7684, pp. 188–204. Springer, Heidelberg (2013)
4. Vigo, R., Nielson, F., Riis Nielson, H.: Broadcast, Denial-of-Service, and Secure Communication. In: Johnsen, E.B., Petre, L. (eds.) IFM 2013. LNCS, vol. 7940, pp. 412–427. Springer, Heidelberg (2013)

Pareto Efficient Solutions of Attack Trees

Zaruhi Aslanyan and Flemming Nielson

DTU Compute, Technical University of Denmark, Denmark
{zaas,fnie}@dtu.dk

Nowadays IT systems rarely work in isolation; they rather cooperate with each other, communicating in an interconnected world. In this growing global computing environment security has become one of the main issues. The continued integration and cooperation of distributed components creates new security problems. Formal methods are necessary to face the complexity of these new scenarios and to study their security properties and threats. Attack trees are a well-known formal yet graphical approach for describing threats on systems and representing the possible attacks.

The first graphical representation for analysing the safety of a system, called fault trees, was introduced in early 1980's. Fault trees represent a failure of a system in terms of the failure of its components [1]. Inspired by fault trees researchers adopted a similar approach to security.

In 1991, Weiss presented threat logic trees as a formal attack modeling technique [2]. Later, in 1999, Schneier introduced attack trees as a tool to evaluate the security of complex systems in a structured, hierarchical way. Attack trees allow to analyse the possible attack scenarios and reason about the security of the whole system in a formal, methodical way, based on varying attacks [3].

The root of an attack tree represents a goal of the attacker. The sub-trees of a node in the tree refine the goal of the node into sub-goals. The leaves of the tree are the basic actions to be executed by the attacker. An internal node shows how the sub-trees have to be combined in order to achieve the overall goal of the attacker. Standard attack trees combine sub-trees either conjunctively or disjunctively, thereby limiting their expressiveness. They do not consider the fact that the absence (negation) of some action might lead to an attack.

Traditional literature on attack trees focuses on single (mainframe) computers and describes threats they are subject to. Most approaches model the attacker's behaviour on such systems by considering one-parameter attack trees and analysing a particular aspect of an attack, such as feasibility or cost. The analyses are performed by assigning values to the basic actions and traversing the tree from the leaves to the root. Different analytical methods suggested different functional operators for computing the value from child nodes to the parent node, based on the type of refinement. However, the study of single computer systems is no longer adequate for dealing with the challenges of a global computing environment. Various extensions of attack trees with multiple parameters have been studied. In most multi-parameter models, values characterizing basic attacks are propagating to the root relying on local decision strategies. In case of incomparable values, however, this approach may yield sup-optimal results.

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 279–280, 2014.
DOI: 10.1007/978-3-319-11599-3,

In order to overcome the limitation in expressiveness of the standard model, we introduce negation as a refinement operator. The extension makes attack trees more flexible and allows to model and analyse a wider range of attack scenarios, including the cases of unrecoverable and conflicting actions. For instance, cutting a communication wire might be unrecoverable, and after having cut a wire we might not be able to communicate with a given device.

Moreover, for analysing complex scenarios with more than one-parameter, we present an evaluation technique that considers basic actions (leaves) characterized by more than one dimension (e.g. probability and cost). In order to deal with such a scenario our technique optimizes all the parameters at once, thus computing different aspects of an attack and handling multiple objectives. Furthermore, as different objectives may conflict with each other, we consider the set of Pareto optimal solutions to face the analysis of incomparable values.

In particular, we study the problem in the settings of a Boolean and a probabilistic semantics for attack trees. For each such semantics, we first consider the problem of feasibility of the attack, and then we extend our technique to compute optimal attacks in presence of multiple costs.

We illustrate the developments on a home-payment system. A home-payment system allows people, who may have difficulties leaving their home, to pay some services such as care-taking or rent. The payment is performed through the remote control of a television box thanks to a contact-less payment card. The card is protected by password to authenticate the owner when a transfer is initiated. The attack scenario that we consider is to steal money from the card-holder by forcing him/her to pay fake services. With our evaluation technique, we have computed the minimum cost of the tree corresponding to the attack scenario in the Boolean and probabilistic cases. In particular, the probabilistic evaluation deals with conflicting parameters, as we want to maximize the probability while minimizing the cost of attacking the system. The result of the evaluation represents the Pareto frontier, each point describing a probability of success with the corresponding cost.

In future work we plan to interpret negation refinements as defender's actions and consider attack-defense trees [4]. We plan to relate an attack-defense scenario with game theory and study the interaction between attackers and defenders as a two player game.

References

1. Vesely, W., Roberts, N., Haasl, D., Goldberg, F.: Fault Tree Handbook, vol. 88. Systems and Reliability Research, Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission (1981)
2. Weiss, J.D.: A system security engineering process. In: Proceedings of the 14th National Computer Security Conference, pp. 572–581 (1991)
3. Schneier, B.: Attack Trees: Modeling Security Threats. Dr. Dobb's Journal of Software Tools 24(12), 21–29 (1999)
4. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation (2012), `http://logcom.oxfordjournals.org/content/early/2012/06/21/logcom.exs029`

Verification of Stateful Protocols

Set-Based Abstractions in the Applied π-Calculus

Alessandro Bruni, Sebastian Mödersheim, Flemming Nielson,
and Hanne Riis Nielson

Technical University of Denmark
{albr,samo,fnie,hrni}@dtu.dk

An ideally designed security protocol should not be state-dependent. In practice however real applications require a certain amount of state for different reasons: encryption keys need to be updated periodically to prevent attackers from learning valid ones, messages are signed with timestamps in order to avoid replaying them after they are no longer valid, etc. Specialised protocols that need to run with bandwidth and real-time constraints may rely solely on state mechanisms to provide their claimed security properties, such as MaCAN and CANAuth, two proposed protocols for automotive that we recently analysed [1].

We propose an extension of the applied π-calculus with support for potentially infinite sets of values. With this extension we are able to analyse protocols with unbounded number of sessions, where security and authenticity properties rely on the use of counters and timestamps, or databases of keys.

We extend the calculus of ProVerif [2,3], a widely used verification tool, which over-approximates its analysis by abstracting away state information translating processes into Horn clauses. The abstraction approach taken by ProVerif simplifies the state exploration, and allows verifying secrecy and authenticity properties over an unbounded number of sessions in many concrete protocols, but is unable of modelling the following simple protocol:

$$A \to B \ : \ \{Msg, Counter\}_{Key}$$

Alice sends to Bob a message, signed with a counter and a shared key. Bob checks whether the counter is new by comparing it with the ones already observed, and accepts only fresh messages. Injective correspondences in ProVerif cannot prove its freshness, because there is no injective relation between the session identifiers of the processes for A and B. Encoding the protocol is also non-trivial as the applied π-calculus does not have a global non-monotonic state.

StatVerif [4] presented an extension of the applied π-calculus that added a global synchronised state, allowing the analysis of stateful processes. However we were not able to encode sets of values without generating terms of ever-increasing size, which lead to non-termination of the analysis. The authors suggested the need for further abstractions in such cases.

Our analysis applies the set-membership abstraction, as proposed by AIF [5], while translating the protocol description into Horn clauses. Values in the process algebra are mapped to their membership class, a term $\mathsf{val}(x, x_{s_1}, \ldots, x_{s_n})$ that abstracts the sets to which x belongs. Our particular encoding allows a

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 281–282, 2014.
DOI: 10.1007/978-3-319-11599-3,

$$
\begin{array}{rll}
M, N ::= & x \mid a \mid f(M_1, \ldots, M_n) & \text{variables, names, constructors} \\
P, Q ::= & 0 \mid\ !P \mid\ P_1 \,|\, P_2 & \text{nil, replication, parallel composition} \\
& \mid \overline{M}\langle N \rangle.P \mid M(x:T).P \mid (\nu\, a:A)\, P & \text{output, typed input, restriction} \\
& \mid \mathsf{let}\ x = \mathsf{g}(M_1, \ldots, M_n)\ \mathsf{in}\ P\ \mathsf{else}\ Q & \text{destructor application} \\
& \mid \mathsf{if}\ M \in s_i\ \mathsf{then}\ P\ \mathsf{else}\ Q & \text{set membership test} \\
& \mid \mathsf{enter}(M, s_i).P \mid \mathsf{exit}(M, s_i).P & \text{set membership transitions} \\
& \mid \mathsf{lock}(s_i).P \mid \mathsf{unlock}(s_i).P & \text{acquire/release set lock}
\end{array}
$$

Fig. 1. The process calculus

compact representation of potentially infinite values, while still distinguishing two different values in the same class, increasing the precision of the analysis.

The calculus is presented in Figure 1. As in ProVerif, we have terms M, N which are either variables, names or constructor applications. Constructors are generally accompanied by destructors defined as rewrite rules that describe cryptographic primitives, for example $\mathsf{dec}(\mathsf{enc}(msg, key), key) \rightarrow msg$ defines the behaviour of symmetric key cryptography.

Processes P, Q are the usual stuck process, replication, parallel composition of two processes, output, typed input, restriction and destructor application. The distinguishing feature of our calculus is the ability to track values in sets: the set membership test allows us to check whether a term M is in a set s, while $\mathsf{enter}(M, s)$ and $\mathsf{exit}(M, s)$ respectively transition to a state where M is in set s and M is not in set s. Finally we use finer grained locks than StatVerif to increase the precision of our analysis.

By lifting the set-membership abstraction to the applied π-calculus we reduce the abstraction gap required in the verification of security protocols that rely on mechanism such as key databases, counter or timestamps to ensure security and authenticity properties.

References

1. Bruni, A., Sojka, M., Nielson, F., Nielson, H.R.: Formal Verification of the MaCAN Protocol. To appear in The 11th International Conference on Integrated Formal Methods. LNCS (to appear, 2014)
2. Blanchet, B.: An efficient cryptographic protocol verifier based on Prolog rules. In: IEEE Computer Security Foundations Workshop. IEEE Computer Society (2001)
3. Blanchet, B.: From Secrecy to Authenticity in Security Protocols. In: Hermenegildo, M.V., Puebla, G. (eds.) SAS 2002. LNCS, vol. 2477, pp. 342–359. Springer, Heidelberg (2002)
4. Arapinis, M., Ritter, E., Ryan, M.D.: Statverif: Verification of stateful processes. In: 2011 IEEE 24th Computer Security Foundations Symposium (CSF). IEEE (2011)
5. Mödersheim, S.A.: Abstraction by set-membership: Verifying security protocols and web services with databases. In: Proceedings of the 17th ACM Conference on Computer and Communications Security. ACM (2010)

Improvement Proposal for the CryptoCloak Application

Dijana Vukovic[1,2], Danilo Gligoroski[1], and Zoran Djuric[2]

[1] Department of Telematics, NTNU, O.S. Bragstads plass 2B, Trondheim, Norway
[2] Department of Computer Science and Informatics, Faculty of Electrical Engineering, Patre 5, Banja Luka, Bosnia and Herzegovina

Abstract. Since July 2013, huge effort was invested into spy-resistant application development. Different initiatives were organized world-wide by EFF[6] and similar organizations to fight for the Internet as it is used to be (e.g. "The day we fight back", "Reset the Net", etc.). CryptoCloak is an application for privacy protected chat communication. Encrypted communication is masked with dynamic cheap chat conversation. In current version of this application, Diffie-Hellman key exchange is done in clandestine manner - instead of sending uniform sequence of numbers, sentences are sent. It produces huge overhead. In this paper, one proposal for its improvement is given.

1 Introduction

Internet surveillance exists for a long time, but people became more aware of it after Snowden affair started[7]. Revelations about NSA partnership with leading companies in the Internet communication area appeared (e.g. Microsoft, Google, Skype, etc.). The fact that the NSA had an access to the private communication of individuals is a huge violation of privacy. Surveillance, as a close observation of a person or a group, can be justified in the case that observed person is under suspicion, or in the case that surveillance can help in preventing crime or terrorism. Considering the simple definition of privacy as "the right to be left alone", conclusion is - the surveillance can be a big threat to the privacy. "Surveillance/privacy" issue led to developing tools for anonymous communication over the Internet. The most popular chat application with this purpose are: Cryptocat[1] and Telegram[2]. Our approach implemented in CryptoCloak application was for the first time presented on BalkanCrypt Workshop[3]. The main idea was use of solid and secure cryptoalgorithms to provide secure chat communication, but do it in the clandestine way - instead of sending encrypted information, mask them with cheap chat. Cheap chat - sentences such as: "Hello!", "How are you?", used in everyday chat communication, are the cloak for hiding encrypted information. CryptoCloak application is written in Java programming language. Major disadvantage in the current version of CryptoCloak is: to accomplish successful key exchange using cheap chat it takes around 30 minutes[4]. Second chapter gives an overview of proposed improvement for the CryptoCloak application. At the end, some notes and comments on the current state of CryptoCloak project is given.

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 283–284, 2014.
DOI: 10.1007/978-3-319-11599-3,

2 An Improvement Suggestion for the CryptoCloak

To speed up the current key exchange process in CryptoCloak, parameters a and b, needed for Diffie-Hellman key exchange can be sent as an e-mail message. It can be implemented the way is shown in Figure 1. Using the same algorithm from previous version[4], parameters will be converted into array of sentences, and, instead of sending these sentences via chat communication, they will be sent using legitimate e-mail account from well-known and secure e-mail server. When the particular parameter is received, it will be transformed into its real value, and the key will be calculated. The same process will be executed on both sides, Bob's and Alice's, and after successful Diffe-Hellman key exchange, they can start AEC-CBC encrypted communication. User can send/receive e-mail message over/from different accounts. If we split communication this way, it will be efficient and harder to follow. This technique will be similar to the the one the Tor[5] uses - based on twisty, hard to follow routes. Although, this provides exposure diversification - if communication is intercepted, it will still be hard to determine from where the message is sent or who is the sender.

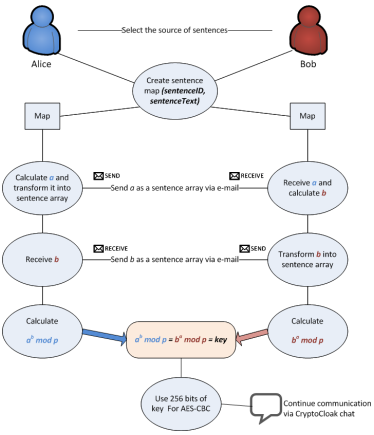

Fig. 1. CryptoCloak improvement

Notes and Comments. Suggested improvement is in an implementation phase. In further versions, threat model has to be created, the perfect forward secrecy has to be proven, and a way to cope with cryptanalysis techniques has to be given.

References

1. Cryptocat, `https://crypto.cat/`
2. Telegram, `https://telegram.org/`
3. Vukovic, D.: The CryptoCloak Project. BalkanCrypt Kickoff Meeting and Workshop, Sofia, Bulgaria (2013)
4. Vukovic, D., Gligoroski, D., Djuric, Z.: On privacy protection in the Internet surveillance era. In: SECRYPT 2014, Vienna, Austria (accepted for publication, 2014)
5. Tor project - About, `https://www.torproject.org/about/overview.html.en`
6. Electronic Frontier Foundation - About, `https://www.eff.org/about`
7. Edward Snowden and the NSA files – timeline, `http://www.theguardian.com/world/2013/jun/23/edward-snowden-nsa-files-timeline`

Process Tracking for Forensic Readiness

Yi-Ching Liao and Hanno Langweg

Norwegian Information Security Laboratory, Gjøvik University College,
Teknologivn. 22, 2815 Gjøvik, Norway
{yi-ching.liao,hanno.langweg}@hig.no

Abstract. We summarize our research on process tracking for forensic readiness, including the state-changing activities of processes, cost-benefit analysis of process tracking, and the architecture for process tracking. We consider the privacy and admissibility issues as future work.

Keywords: forensic readiness, process tracking, kernel tracing.

1 Introduction

Forensic analysis suffers from insufficient logging of events, and current system loggers do not record enough information for incident analysis and replay. Similar to flight data recorders preserving performance parameters for aircraft accident investigation, comprehensive process tracking can provide precise, timely, complete, and dependable information for incident investigation and replay. Moreover, the collected traces can recover the traceability links between the incident and the person or action accountable for the incident.

2 The State-Changing Activities of Processes

Without identifying the state-changing activities of processes, it is impossible to know when to track and what to log. To answer the question: "What are the state-changing activities of processes?", we evaluated the existing process tracking systems from the perspectives of forensic analysis and forensic readiness, including the logging method, the tracing granularity, the replay boundary, and the implementation method [1]. We found that most process tracking systems for security aim at the process-level granularity, which is insufficient for determining the root cause of incidents. On the other hand, the instruction-level tracing can provide fine-grained information for incident analysis and replay, but the cost of process activity tracking can be quite expensive. To strike a balance between the forensic effectiveness and efficiency, it is essential to evaluate the soundness, completeness, and cost of process activity tracking.

3 Cost-Benefit Analysis of Process Tracking

To meet the two objectives of forensic readiness [2]: maximizing the usefulness and minimizing the cost, it is important to perform cost-benefit analysis of process tracking. Since kernel tracing systems can provide more dependable and

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 285–286, 2014.
DOI: 10.1007/978-3-319-11599-3,

comprehensive process activities for incident investigation and forensic analysis, to answer the question: "How effective, efficient, and expensive is comprehensive process activity tracking?", we conducted a cost-benefit analysis of three kernel tracing systems: strace, SystemTap, and LTTng [3]. We discovered that LTTng can provide system-wide tracing with lower performance and storage overhead. On the other hand, strace and SystemTap can provide better flexibility for tracing evolving intruder tactics and hacking techniques through dynamic instrumentation. For cost-benefit trade-off, it is necessary to design the architecture for flexible and adjustable process tracking.

4 The Architecture for Process Tracking

Security incidents or digital crimes must occur with system resource usage. Since system calls cause state transitions of resource usage, they are at the proper level of granularity for process tracking. To answer the question: "Which architecture facilitates process activity tracking?", we presented a resource-based event reconstruction prototype that corresponds to different phases of digital forensics framework, and conducted a feasibility study by assessing the applicability of existing open-source applications to the proposed prototype [4]. By regarding system resources as an evidence source and system calls as digital events, the proposed prototype can enhance the capability of an organization for gathering, preserving, protecting, and analysing digital evidence.

5 Future Work

Since kernel traces may contain personal information and aggregating traces from various hosts can raise serious privacy concerns, to protect the confidentiality of personally identifiable information, we need to answer the question: "What are privacy implications for users of systems that support comprehensive traceability?" by conducting privacy impact assessments. Moreover, to ensure that the collected traces are admissible as evidence, we will answer the question: "How does comprehensive traceability affect evidence gathering and the legal process?" by conducting security and vulnerability assessments.

References

1. Liao, Y.C., Langweg, H.: A Survey of Process Activity Tracking System. In: 6th Norsk Informasjons Sikkerhets Konferanse, pp. 49–60 (2013)
2. Tan, J.: Forensic Readiness, pp. 1–23. @ Stake, Cambridge (2001)
3. Liao, Y.C., Langweg, H.: Cost-Benefit Analysis of Kernel Tracing Systems for Forensic Readiness. In: Proceedings of the 2nd International Workshop on Security and Forensics in Communication Systems, pp. 25–36. ACM, New York (2014)
4. Liao, Y.C., Langweg, H.: Resource-based Event Reconstruction of Digital Crime Scenes. In: IEEE Joint Intelligence and Security Informatics Conference (to be published, 2014)

Computationally Analyzing the ISO 9798–2.4 Authentication Protocol

Britta Hale and Colin Boyd

Norwegian University of Science and Technology – NTNU

As it is widely agreed that authentication protocols are difficult to design correctly, standardized authentication protocols are very useful for practitioners. Among the protocols available from a variety of different standards bodies, some are widely deployed, such as the well known TLS and SSH protocols. Among its 9798 series of standards, the ISO have standardized a suite of authentication protocols, yet like most standardized authentication protocols, these are not defined in a fully formal way. Effectively, among other possible undesirable consequences, this can lead to uncertainty about how to correctly implement the protocols securely.

With the goal of providing computational proofs for one of the 9798–2 protocols which have so far been lacking, we focus on ISO 9798–2.4 (9798-2, section 6.2.2 Mechanism 4 of the standard [2]) which is shown below. Notationally, $Text_i$ is an optional text field, $\mathcal{E}_K$ an "encipherment function" between A and B [2, p. 4], I_B an optional unique distinguisher, and R_i a random nonce.

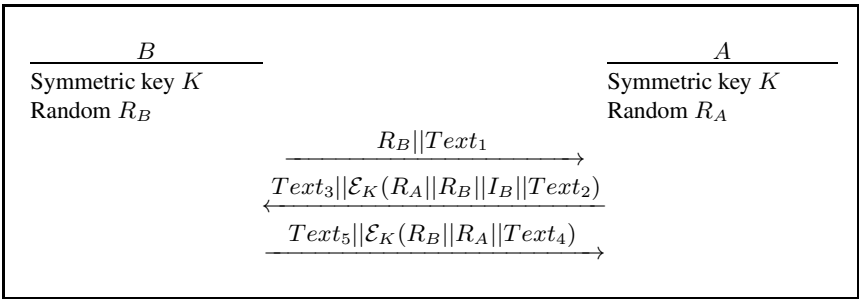

Fig. 1. ISO 9798–2.4 Protocol

CHOICE OF CRYPTOGRAPHIC PRIMITIVES. ISO 9798–2.4 protocol makes use of an encipherment algorithm with a shared symmetric encipherment key and requires that it is able to detect "forged or manipulated data"[2, p. 4]. Authenticated encryption (AE) is recommended for implementation. However, any formal definition or technical description of such properties is missing from the standard and it is observable that entity authentication can be achieved without use of encryption at all. Thus, aiming to obtain security under maximal efficiency, we show in our computational security proof that a message authentication code (MAC) algorithm can be safely implemented.

COMPUTATIONAL SECURITY. Focus in the computational security proof is on the protocol core – the optional fields in the protocol are not considered. In the Bellare–Rogaway '93 model [1], principals possess matching conversations if and only if they accept. Correspondingly, adversarial advantage, $\mathbf{Adv}^{\mathrm{MA}}_{\Pi}(E)$, is defined as the probability that the adversary can succeed in persuading an oracle to accept without a matching

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 287–288, 2014.
DOI: 10.1007/978-3-319-11599-3,

conversation. If p the number of principals, S the number of sessions, 1^k the security parameter, and q queries allowed to $\mathcal{A}$, and E runs in time t and asks q queries, then

$$\mathbf{Adv}^{\mathrm{MA}}_{\Pi}(E) \leq 2p^2S \cdot \mathbf{Adv}^{\mathrm{MAC}}_{\Pi}(F) + \tfrac{q^2}{2^{k+1}}.$$

Moreover, F runs in time $t_F \approx t$ and asks $q_F = q$ queries.

ROGAWAY–STEGERS FRAMEWORK. While the analysis above demonstrates security of the ISO 9798–2.4 protocol core it omits the optional text fields, an important aspect of the original protocol. Rogaway and Stegers [3] introduced a model that addresses this issue by splitting the protocol into two parts: the partially specified protocol core (PSP) and the protocol details (PD), whicht selects content for the text fields. Yet, since there is no restriction on the data that is sent in these fields, it is necessary to consider that data choice could weaken the protocol. Allowing the adversary itself to choose the optional text fields models this weakness. Essentially, the Rogaway–Stegers framework under this assumption requires that mutual authentication is satisfied in addition to requiring that matching session IDs (in our implementation, matching conversations) imply matching AD. Thus, we capture the optional fields of ISO 9798–2.4 while applying the Rogaway–Stegers framework in its first application to a standardized protocol.

Discriminately, data fields fall into two categories, with the authenticated associated data (AD) being of salient concern. Of the ISO 9798–2.4 protocol's text fields T_l for $l \in \{1, \ldots, 5\}$, only T_2 is authenticated and guaranteed received by the protocol, and hence is the only AD. Succinctly, the proof of security, when AD is considered, builds on that of the protocol core and the final reduction of security is the same.

ISO 9798–2.4 WITH AE. While the ISO 9798–2 standard currently does not specify the primitive for $\mathcal{E}_K$, it does suggest AE per ISO/IEC 19772. Consequently, it is desirable to know if the security of a protocol implemented under AE is traceable to that under a MAC primitive. Hence we achieve the following result.

Theorem 1. *Let Π be the 9798–2.4 protocol implemented with a strongly unforgeable AE algorithm $(\mathcal{K}, \mathcal{E}, \mathcal{D})$. Let Π' be the 9798–2.4 protocol implemented with the MAC as $\mathrm{MAC}_K(M) = (M, \mathcal{E}(K, M))$. An efficient adversary against Π can be efficiently converted into an adversary against Π' with the following advantage, for n adversarial queries:*

$$\mathbf{Adv}^{\mathrm{AE}}_{\Pi}(\mathcal{A}) \leq (2p^2S + n) \cdot \mathbf{Adv}^{\text{SUF-AE}}_{(\mathcal{K},\mathcal{E},\mathcal{D})}(\mathcal{A}) + q^2/2^{k+1}.$$

Ultimately, these results underscore the security of ISO 9798–2.4, a real-world mutual authentication standard – demonstrating a notable improvement to the standard's current requirements while also validating security in the computational model.

References

1. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
2. ISO: Information technology – security techniques – entity authentication – part 2: Mechanisms using symmetric encipherment algorithms. ISO ISO/IEC 9798-2:2008, International Organization for Standardization, Geneva, Switzerland (2008)
3. Rogaway, P., Stegers, T.: Authentication without Elision: Partially Specified Protocols, Associated Data, and Cryptographic Models Described by Code. In: Proceedings of the 2009 22nd IEEE Computer Security Foundations Symposium, pp. 26–39. IEEE Computer Society (2009)

Differential Privacy and Private Bayesian Inference*

Christos Dimitrakakis[1], Blaine Nelson[2,**], Aikaterini Mitrokotsa[1], and Benjamin I.P. Rubinstein[3]

[1] Chalmers University of Technology, Sweden
[2] University of Potsdam, Germany
[3] The University of Melbourne, Australia

We consider a Bayesian statistician ($\mathcal{B}$) communicating with an untrusted third party ($\mathcal{A}$). $\mathcal{B}$ wants to convey useful answers to the queries of $\mathcal{A}$, but without revealing private information. For example, we may want to give statistics about how many people suffer from a disease, but without revealing whether a particular person has it. This requires us to strike a good balance between utility and privacy. In this extended abstract, we summarise our results on the inherent privacy and robustness properties of Bayesian inference [1]. We formalise and answer the question of whether $\mathcal{B}$ can select a prior distribution so that a computationally unbounded $\mathcal{A}$ cannot obtain private information from queries. Our setting is as follows:

(i) $\mathcal{B}$ selects a model family ($\mathcal{F}_\Theta$) and a prior (ξ).
(ii) $\mathcal{A}$ is allowed to see $\mathcal{F}_\Theta$ and ξ and is computationally unbounded.
(iii) $\mathcal{B}$ observes data x and calculates the posterior $\xi(\theta|x)$ but does not reveal it. Instead, $\mathcal{B}$ responds to queries at times $t = 1, \ldots$ as follows.
(iv) $\mathcal{A}$ sends a query q_t to $\mathcal{B}$.
(v) $\mathcal{B}$ responds $q_t(\theta_t)$ where θ_t is drawn from the posterior: $\theta_t \sim \xi(\theta|x)$.

We show that by choosing $\mathcal{F}_\Theta$ or ξ appropriately, the resulting posterior-sampling mechanism satisfies generalised differential privacy and indistinguishability properties. The intuition is that robustness and privacy are linked via smoothness. Learning algorithms that are smooth mappings—their output (*eg.* a spam filter) varies little with perturbations to input (*e.g.* similar training corpora)—are robust: outliers have reduced influence, and adversaries cannot easily discover private information. Consequently, robustness and privacy may be simultaneously achieved and perhaps are deeply linked.

Our results [1] show that mild assumptions are sufficient to obtain a differentially-private mechanism in the Bayesian setting. As a first step, we generalise the definition of differential privacy [2] to arbitrary dataset spaces $\mathcal{S}$. To do so, we introduce the notion of differential privacy under a pseudo-metric ρ on the space of all datasets.

* This work was partially supported by the Marie Curie Project ESDeMUU grant No: 237816 and the FP7 STREP project BEAT, grant No: 284989.
** Blaine Nelson is now at Google, Mountain View.

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 289–290, 2014.
DOI: 10.1007/978-3-319-11599-3,

Definition 1 ((ϵ, δ)-differential privacy under ρ.). *A conditional distribution $P(\cdot \mid x)$ on $(\Theta, \mathfrak{S}_\Theta)$ is* (ϵ, δ)-differentially private under *a pseudo-metric $\rho : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$ if, for all $B \in \mathfrak{S}_\Theta$ and for any $x \in \mathcal{S}$, then $P(B \mid x) \leqslant e^{\epsilon\rho(x,y)} P(B \mid y) + \delta\rho(x, y)\ \forall y$.*

Our first assumption is that the $\mathcal{F}_\Theta$ is smooth with respect to some metric d:

Assumption 1 (Lipschitz continuity) *Let $d(a, b) \triangleq |\ln a/b|$. There exists $L > 0$ such that, for any $\theta \in \Theta$: $d(p_\theta(x), p_\theta(y)) \leqslant L\rho(x, y), \quad \forall x, y \in \mathcal{S}$.*

As it can be hard for this assumption to hold uniformly over Θ, we relax it by only requiring that $\mathcal{B}$'s *prior* probability ξ is concentrated in the smoothest members of the family:

Assumption 2 (Stochastic Lipschitz continuity) *Let Θ_L be the set of L-Lipschitz parameters. Then $\exists c > 0$ such that, $\forall L \geqslant 0$: $\xi(\Theta_L) \geqslant 1 - \exp(-cL)$.*

One consequence of either of those assumption is that the posterior is robust, in the sense that small dataset changes result in small changes in the posterior:

Theorem 1. *If ξ is a prior on Θ and $\xi(\cdot \mid x)$ and $\xi(\cdot \mid y)$ are the respective posterior distributions for datasets $x, y \in \mathcal{S}$, then the posterior KL-divergence satisfies: $D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leqslant O(\rho(x, y))$, with linear terms depending on L, c.*

Consequently, one way to answer queries would be to use samples from the poster distribution. In fact, we show that such posterior-sampling mechanisms are differentially private:

Theorem 2. *Under Assumption 1, the posterior is $(2L, 0)$-differentially private under ρ. Under Assumption 2, the posterior ξ is $\left(0, \sqrt{\frac{\kappa}{2c}}\right)$-differentially private under $\sqrt{\rho}$.*

As the adversary performs more queries, he obtains more information about the true dataset. Finally, we bound the effort required by an adversary to be ϵ-close to the true dataset:

Theorem 3. *The adversary can distinguish between data x, y with probability $1 - \delta$ if $\rho(x, y) \geqslant O(\frac{\ln 1/\delta}{n})$, with a linear dependency on L or c.*

We have shown that both the privacy and robustness properties of Bayesian inference are inherently linked through the choice of prior distribution. Such prior distributions exist for example in well known conjugate families. There is also a natural *posterior sampling* mechanism through which differential privacy and dataset indistinguishability can be achieved.

References

[1] Dimitrakakis, C., Nelson, B., Mitrokotsa, A., Rubinstein, B.: Robust and private bayesian inference

[2] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)

Event Invitations in Privacy-Preserving DOSNs*

Formalization and Protocol Design

Guillermo Rodríguez-Cano, Benjamin Greschbach, and Sonja Buchegger

KTH Royal Institute of Technology
School of Computer Science and Communication
Stockholm, Sweden
{gurc,bgre,buc}@csc.kth.se

The most common form of Online Social Networks (OSNs) are run in a logically centralized manner (although often physically distributed), where the provider operating the service acts as a communication channel between the individuals. Decentralization has been proposed to reduce the effect of these privacy threats by removing the central provider and its ability to collect and mine the data uploaded by the users as well as behavioral data.

One of the standard features of OSNs is the handling of event invitations and participation, i. e., a call for an assembly of individuals in the social graph for a particular purpose, e. g., a birthday celebration, demonstration, or meeting. There is usually metadata related to each event, such as date, location and a description. An implementation of this feature must provide security properties, e. g., that a user can verify that an invitation she received was actually sent by the organizer. Furthermore, it must support certain privacy settings. For example, an organizer could choose that only invited users learn how many other users were invited and that only after a user has committed to attend the event, she learns the identities of these other invited users.

Realizing this in a decentralized scenario is non-trivial because there is no Trusted Third Party (TTP) which all involved users can rely on. This is a problem especially for privacy properties where information shall only be disclosed to users with a certain status. In the example above, a neutral, trusted broker could keep the secret information (the identities of invited users) and disclose it only to users who committed to attend the event. This would guarantee fairness to both the organizer and the invited users. It becomes more challenging to implement this without a central TTP and still allowing different types of information about the event to be shared with different groups of users in a secure way.

In our proposal for a privacy-preserving decentralized implementation of the event invitation feature, as depicted in Figure 1, we divide the users of the Decentralized Online Social Network (DOSN) into the organizer of the event, the invitees, those who confirmed attending the event (attendees) and the remaining users. We assume basic functionality of popular OSNs to be available in a decentralized manner, such as user search or user messaging. We also assume

* A full paper on this work will be presented at the Privacy and Identity Management for the Future Internet in the Age of Globalisation - 9th IFIP WG 9.2, 9.5, 9.6/11.7, 11.4, 11.6 and Special Interest Group 9.2.2 International Summer School, Patras, Greece, September 7-12, 2014.

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 291–292, 2014.
DOI: 10.1007/978-3-319-11599-3,

that users are identified by a public key and the ability to verify the identity of other users via some sort of Public Key Infrastructure, which can be realized in a decentralized manner. Moreover, we rely on a distributed storage featuring access right management, e. g., that a certain storage object is only writeable by a specific user, and "append-only" storage objects, where new data can be appended, but existing data cannot be modified or removed without notice.

We describe and formally define two basic and five more complex security and privacy properties for the event invitations feature in DOSNs, such as invitee/attendee identity privacy (who learns the identities of the invitees/attendees), invitee/attendee count privacy (who learns the count of invitees/attendees), and attendee-only information reliability (availability of information exclusive to the attendees).

We also describe privacy enhancing tools, such as storage location indirection (to control not only who can decrypt an object but also who can see a ciphertext), controlled ciphertext inference (to allow a controlled information leak, e. g., about the size of an encrypted object to parties not able to decrypt the content) and a custom "commit-disclose protocol" to disclose a secret only to users who committed to attend an event. Using these tools together with standard cryptographic primitives, we discuss and propose a TTP-free architecture and decentralized protocols to implement the event invitation feature in a DOSN and analyze the usability and privacy implications. The suggested protocols cover all of our defined properties, considering 20 different parameter combinations for the tunable privacy properties.

The results can be applied in the context of Privacy-Preserving DOSNs, but might also be useful in other domains such as Working Environment and their corresponding collaborative-specific tools, i. e., groupware.

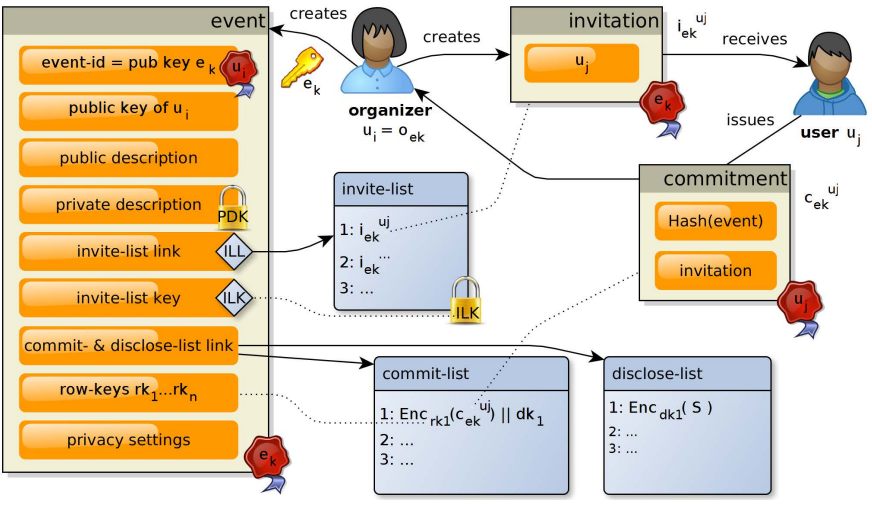

Fig. 1. Overview of system components and actions between stakeholders

Attacks on Privacy-Preserving Biometric Authentication

Aysajan Abidin, Elena Pagnin, and Aikaterini Mitrokotsa

Chalmers University of Technology, Gothenburg, Sweden
{aysajan.abidin,elenap,aikmitr}@chalmers.se

Abstract. Biometric authentication based on facial image, fingerprint, palm print, iris, retina, or veins are becoming increasingly popular. However, compromised biometric templates, indeed, may lead to serious threats to identity and their inherent irrevocability makes this risk even more serious. Because of such serious privacy implications the need for *privacy-preserving biometric authentication protocols* is of utmost importance. Recently, Yasuda *et al.* [1,2] proposed two efficient privacy-preserving biometric authentication using packed homomorphic encryption based on ideal lattices and on ring learning with error. We review these protocols and analyse their security against *malicious* internal adversaries.

Yasuda *et al.* [1,2] have proposed two packed homomorphic encryption schemes based, respectively, on ideal lattices and on ring-LWE (ring-learning-with-errors). Let $\mathsf{vE}_1(\cdot)$ be the type 1 packed encryption, and $\mathsf{vE}_2(\cdot)$ the type 2 packed encryption. Let A and B be bitstrings of length N. Then, $\mathsf{ct}_H = C\mathsf{vE}_1(A) + C'\mathsf{vE}_2(B) - 2\mathsf{vE}_1(A)\mathsf{vE}_2(B)$ corresponds to an encryption of the Hamming distance between A and B, for suitable chosen constants C and C'. In particular, $\mathsf{vE}_1(A)\mathsf{vE}_2(B)$ provides an encryption of the inner product between A and B. Both protocols involve three entities (a client server $\mathcal{C}$, a computation server $\mathcal{CS}$ and an authentication server $\mathcal{AS}$) and are composed of three phases:

– ***Setup Phase:*** $\mathcal{AS}$ generates the public key pk and the secret key sk for the SHE schemes, and distributes only pk to both $\mathcal{C}$ and $\mathcal{CS}$.

– ***Enrolment Phase:*** $\mathcal{C}$ generates a feature vector A from the client's biometric readings, computes $\mathsf{vE}_1(A)$, and sends it with client's ID to $\mathcal{CS}$, who then stores $\mathsf{vE}_1(A)$ and ID in its database $\mathcal{DB}$.

– ***Authentication Phase:*** $\mathcal{C}$ generates a feature vector B from the client's fresh biometric readings, computes $\mathsf{vE}_2(B)$, and sends it with the client's ID to $\mathcal{CS}$. Then, $\mathcal{CS}$ retrieves the template $\mathsf{vE}_1(A)$ corresponding to ID from $\mathcal{DB}$, computes ct_H and sends ct_H to $\mathcal{AS}$. Subsequently, $\mathcal{AS}$ decrypts ct_H with the secret key sk to obtain the Hamming distance $\mathsf{HD}(A, B)$. Finally, $\mathcal{AS}$ returns the authentication result YES (resp. NO) to $\mathcal{C}$ if $\mathsf{HD}(A, B) \leqslant \tau$ (resp., otherwise), where τ is a pre-defined threshold.

We briefly describe the attack algorithms that could be employed when $\mathcal{C}$ (Algorithm 1) and $\mathcal{CS}$ (Algorithm 2) are malicious. Note that Algorithm 1 can also be employed by a compromised $\mathcal{CS}$. In the attack algorithm descriptions, $\mathcal{C} \xrightarrow{A} \mathcal{CS}$ denotes $\mathcal{C}$ sends A to $\mathcal{CS}$.

K. Bernsmed and S. Fischer-Hübner (Eds.): NordSec 2014, LNCS 8788, pp. 293–294, 2014.
DOI: 10.1007/978-3-319-11599-3,

Algorithm 1 Center search attack

Input: $B = B_1, \cdots, B_N$ (fresh)
Output: $A = A_1, \cdots, A_N$ (reference)
for $i = 1$ to N: **do**
  $D \leftarrow \overline{B_1}, \ldots, \overline{B_i}, B_{i+1}, \ldots, B_N$
  $\mathcal{C} \xrightarrow{\mathsf{vE}_2(D)} \mathcal{CS}$
  $\mathcal{CS} \xrightarrow{\mathsf{ct}_H} \mathcal{AS}$
  if rejected **then**
    break
  end if
end for
for $i = 1$ to N: **do**
  $\mathcal{C} \xrightarrow{\mathsf{vE}_2(D_1,\ldots,\overline{D_i},D_{i+1},\ldots,D_N)} \mathcal{CS}$
  $\mathcal{CS} \xrightarrow{\mathsf{ct}_H} \mathcal{AS}$
  if accepted **then**
    $A_i \leftarrow \overline{D_i}$
  else
    $A_i \leftarrow D_i$
  end if
end for

Algorithm 2 Cheating attack

Input: $\mathsf{vE}_1(A)$
Output: $A = A_1, \cdots, A_N$
Initialise: $A = 0_1 0_2 \cdots 0_N$
for $i = 0$ to $N - \tau$: **do**
  $D \leftarrow 1_1 \cdots 1_{\tau+i}\, 0_{\tau+i+1} \cdots 0_N$
  $\mathcal{CS} \xrightarrow{\mathsf{vE}_1(A)\mathsf{vE}_2(D)} \mathcal{AS}$
  if rejected **then**
    break
  end if
end for
$i' \leftarrow \tau + i$; $A_{i'} \leftarrow 1$
for $i = 1$ to $i' - 1$: **do**
  $D \leftarrow 1_1 \cdots 1_{i-1}\, 0_i\, 1_{i+1} \cdots 1_{i'} 0 \cdots 0_N$
  $\mathcal{CS} \xrightarrow{\mathsf{vE}_1(A)\mathsf{vE}_2(D)} \mathcal{AS}$
  if accepted **then**
    $A_i \leftarrow 1$
  end if
end for
for $i = i' + 1$ to N: **do**
  $D \leftarrow 1_1 \cdots 1 0_{i'} \cdots 0 1_i 0 \cdots 0_N$
  $\mathcal{CS} \xrightarrow{\mathsf{vE}_1(A)\mathsf{vE}_2(D)} \mathcal{AS}$
  if rejected **then**
    $A_i \leftarrow 1$
  end if
end for

We reviewed two recently proposed privacy-preserving biometric authentication protocols and presented two attack algorithms. The center search attack (Algorithm 1) enables to recover a reference biometric template using a fresh acceptable template. The second attack (Algorithm 2) allows the recovery of reference templates of arbitrary users. Both attacks require a number of authentication attempts that is linear in N (*i.e.* the length of the biometric template) to fully recover a reference template.

Acknowledgements. This work was partially supported by the FP7-STREP project "BEAT: Biometric Evaluation and Testing", grant number: 284989.

References

1. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Packed homomorphic encryption based on ideal lattices and its application to biometrics. In: Cuzzocrea, A., Kittl, C., Simos, D.E., Weippl, E., Xu, L. (eds.) CD-ARES 2013 Workshops. LNCS, vol. 8128, pp. 55–74. Springer, Heidelberg (2013)
2. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Practical packing method in somewhat homomorphic encryption. In: Garcia-Alfaro, J., Lioudakis, G., Cuppens-Boulahia, N., Foley, S., Fitzgerald, W.M. (eds.) DPM 2013 and SETOP 2013. LNCS, vol. 8247, pp. 34–50. Springer, Heidelberg (2014)

Author Index