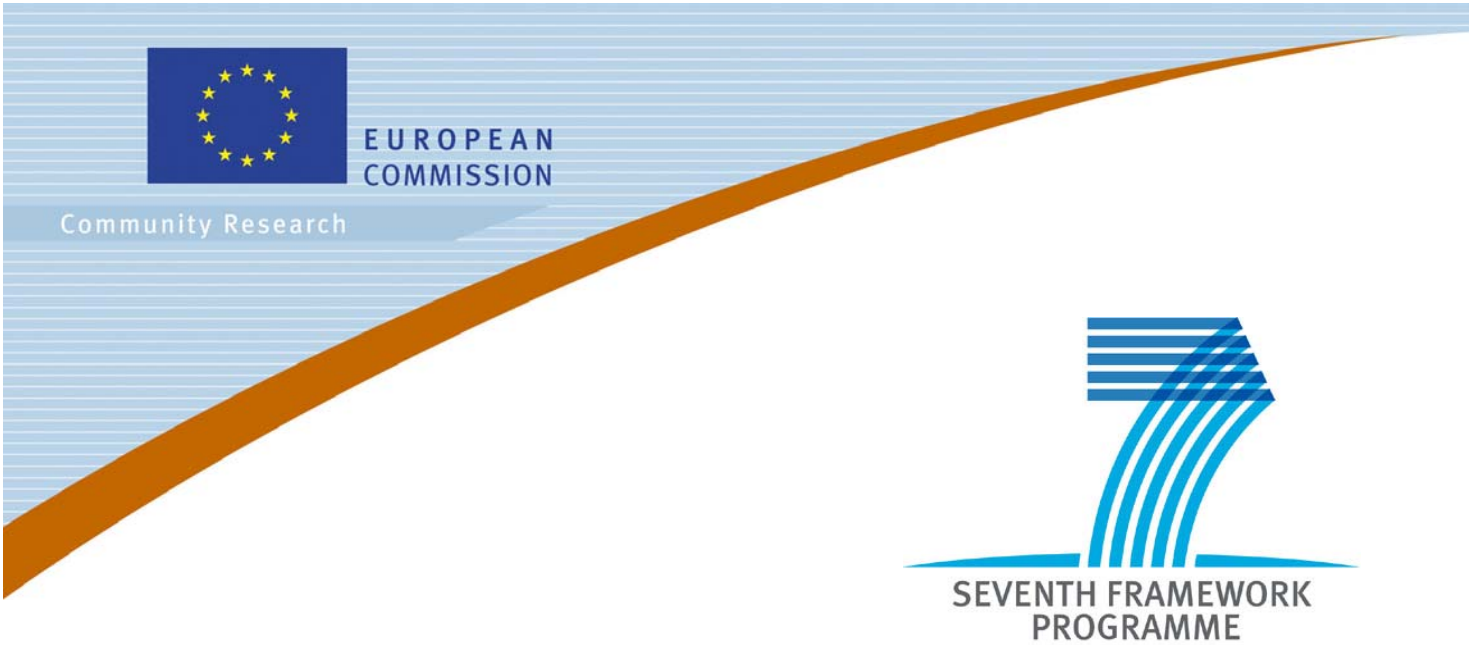

Specific Targeted REsearch Project

PRISM

State of the art on data protection algorithms for monitoring systems

Project acronym: PRISM

Project full title: Privacy-Aware secure Monitoring

Contract No.: 215350

Project Document Number: IST-2007-215350-WP3.1-D3.1.1-R1

Project Document Date: 30/06/2008

Workpackage Contributing to the Project Document: WP3.1

Deliverable Type and Security: Public

Author(s): Carsten Schmoll, FRAUNHOFER
Simone Teofili, Emanuele Delzeri, Giuseppe Bianchi, CNIT
Ivan Gojmerac, Esa Hyytia, FTW
Brian Trammell, Elisa Boschi, HIT
George Lioudakis, Fotis Gogoulos, Anna Antonakopoulou, ICCS
Dimitra Kaklamani, Iakovos Venieris, ICCS

Abstract:

Goal of this deliverable is to review and analyse data protection algorithms. The analysis takes into account both classes of approaches that are (or can be) directly applied for use in monitoring applications, as well as cryptographic approaches that may be considered for application in the specific scenario promoted by the PRISM approach (i.e., separation between front-end and back-end and possibly reversible data protection mechanisms).

Keyword list: PRISM, IST-2007-215350, Data Protection Algorithms

History

Version	Date	Description, Author(s), Reviser(s)
0.1	21/03/2008	Document creation with Skeleton as discussed at kick-off meeting, Carsten Schmoll.
0.2	05/05/2008	Revised TOC based on Telco output
0.3	04/06/2008	Collected initial contributions before the PRISM Rome meeting
0.4	24/06/2008	Extended document with partners' contributions
0.5	30/06/2008	Finalized D311

Table of Contents

Abbreviations

AAPI	Anonymisation Application Programming Interface
AES	Advanced Encryption Standard
AMD	Advanced Micro Devices, Inc.
AS	Autonomous System
ASCII	American Standard Code for Information Interchange
BIOS	Basic Input/Output System
CPU	Central Processing Unit
DBA	Database Administrator
DBMS	Database Management System
DES	Data Encryption Standard
DFA	Deterministic Finite Automaton
DOS	Disk Operating System
DRAM	Dynamic Random Access Memory
EPAL	Enterprise Privacy Authorization Language
FP7	Seventh Framework Programme
FPGA	Field-Programmable Gate Array
FTP	File Transfer Protocol
HP	Hewlett Packard
HTTP	Hypertext Transfer Protocol
ID	Identifier
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPFIX	Internet Protocol Flow Information eXport
LLC	Logical Link Control
MAC	Media Access Control
MD5	Message-Digest algorithm 5
NGSCB	Next-Generation Secure Computing Base
OAEP	Optimal Asymmetric Encryption Padding
OEM	Original Equipment Manufacturers
OPES	Order Preserving Encryption Schema
OS	Operating System
OT	Oblivious Transfer Protocol
PC	Personal Computer
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PRISM	PRivacy-aware Secure Monitoring
RBAC	Role-based access control
RSA	A crypto-algorithm by Ron Rivest, Adi Shamir, and Leonard Adleman
RTP	Real-Time-Protocol
SEM	Secure Execution Mode
SIP	Session Initiation Protocol
SMC	Secure Multi party Computation
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
SRAM	Static Random Access Memory
TC	Trusted Computing
TCP	Transmission Control Protocol

TCPA	Trusted Computing Platform Alliance
TMP	Trusted Platform Module
TSA algorithm	Time Slot Assignment algorithm
UDP	User Datagram Protocol
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
VoIP	Voice over Internet Protocol
WP	Workpackage
XACML	eXtensible Access Control Markup Language
XML	Extensible Markup Language

List of Tables

List of Figures

1 Introduction

The Internet is of vital importance for many business processes as well as for private communication. Engineers and researchers worldwide are working on measuring, analyzing, and simulating the Internet's behaviour. It is crucial for their work to have access to accurate real-world network data from life environments. Yet disclosure of any of those traces or derived results is a difficult task, since traffic traces are deemed to carry business-sensitive data. This hinders the benefit that many applications would gather from multi-domain data collection. The disclosure of data traces, even if incomplete or modified, would be of great value for researchers, helping them to understand the characteristics and dynamics of the Internet today.

Moreover, privacy of network traffic data is not only an issue emerging when such data is exported to the public research community, but it is also an issue that concerns the entity that captures and controls the data. Indeed, traffic monitoring, as a process, brings about severe legal implications and potential infringement of the data protection right, which is acknowledged as a fundamental right of the individual. Even when traffic data capture is restricted to the header part of the transmitted packets, thus excluding the user payload data, a huge amount of personal information may still be gathered (e.g. who is connecting with whom or with which servers, which applications are used, etc.) and exploited to illegitimately profile individual users.

The fact that data gathered through passive monitoring may be considered as personal data, and as such subject to the data protection legislation, is made evident from a recent opinion of the Article 29 Data Protection Working Party (WP 136, Opinion 4/2007 on the concept of personal data, adopted on June 20, 2007): "*… unless [an] Internet Service Provider is in a position to distinguish with absolute certainty that the data correspond to users that cannot be identified, it will have to treat all IP information [including dynamic IP addresses] as personal data to be on the safe side*".

Therefore we can conclude that traffic monitoring is subject to the Data Protection regulation, and as such an open issue is how to technically process the monitored traffic to respect the principles set forth by such regulation. These principles place a strong emphasis on the fact that personal data should only be processed for specified explicit and legitimate purposes and may not be processed further in a way incompatible with those purposes. Moreover, the so called "proportionality" principle requires that personal data may be processed only insofar as it is adequate, relevant and not excessive in relation to the purposes for which they are collected and/or further processed. However, this is hardly the case for the majority of real world monitoring systems, especially when a huge amount of data, collected in raw (unencrypted) form, virtually allows any possible processing well beyond the monitoring purposes of the specifically running applications.

To concretely face the above described issues, a possibility is to protect network traffic data through anonymisation or encryption techniques. Anonymisation means alteration of parts (fields) of the packet so that the resulting information is considered safe in terms of private information disclosure. We remark that private information is not only carried by the packet payload: inside a raw packet data trace there are different data fields and traffic characteristics which might be considered sensible and should be therefore protected through anonymisation. These include data fields from the LLC/MAC header and IP header, such as packet length, IP addresses, and the IP ID field; data fields from the UDP or TCP header, such as TCP flags, TCP sequence numbers, and UDP/TCP port numbers, or the capture time stamp associated to

each packet. Anonymisation techniques fall in one of the following four groups: Removal, Replacement (remapping), Reduction of accuracy, or Aggregation. Even if the anonymisation process partly alters the content of a traffic trace, many useful metrics can still be derived.

1.1 Goal and Organization of this Deliverable

This deliverable, "***State of the art on data protection algorithms for monitoring systems"*** is scheduled at month 4 of the project lifetime. Goal of this deliverable is to briefly discuss the technical means to protect traffic data captured for monitoring purposes. As discussed in this document, data protection can occur in different ways.

On one side, the protection can occur in terms of control of which specific application or entity accesses collected data: such a systematic approach is expected to be deployed in the system back-end (i.e. where data traces are stored), and as such state of the art authorization and access control means will be briefly summarized in section 2 while presenting the front-end/back-end structure adopted for the PRISM project.

On the other side, protection mechanisms can be directly applied to the traffic data. Techniques devised to anonymised and/or encrypt traffic are extensively discussed in section 3. This section will first review traditional anonymisation approaches and their application to traffic monitoring. We will discuss advantages and drawbacks of the different techniques and available algorithms, and will include a qualitative comparison of their effectiveness and clear statements about the effect on the original traffic metrics. In addition, we will also review some cryptographic techniques (including search approaches over encrypted data) which, even if not applied in the past to traffic monitoring systems, are nevertheless considered solutions which might have a role in our future work in PRISM.

This work and its outcome will allow the project to make an informed decision on the recommendations and use of anonymisation techniques inside the PRISM system, and provide useful hints for the solution to the privacy concerns that often hinder the exchange of traffic data.

2 Areas of Application of Data Protection in PRISM

This section details where data protection by anonymisation can be applied, with respect to the questions (a) at what component in the PRISM system can it be applied and (b) what are the derived implications.

2.1 PRISM Frontend System

2.1.1 Network Card

Specific traffic anonymisation policies can be implemented in the front-end device; such an approach presents some advantages: when the data transferred from the front-end to the back-end are already protected, then the security and encryption constraints of such a communication can be relaxed. Furthermore, a certain amount of simple but high rate processing can be offloaded from the back-end.

Several anonymisation techniques require the obfuscation of some fields of the packet payload, their randomization or their permutation; such functionalities can be supported by a capturing device which takes advantage from the high speed processing capability of network processing hardware. Techniques requiring the creation of meta-data structures or the calculation of cumulative of simple traffic statistics can be supported by the font-end probe as well.

Implementation of other cryptography-based protection techniques can pose performance issues, since such algorithms can be processing intensive; for some encryption algorithms however; special purpose hardware support for encryption is provided by network processing devices, thus offloading a good amount of the computational burden involved with cryptographic techniques.

Indeed, several performance constraints have to be taken into account: the capturing device of the font-end probe must retain the potential for processing millions of packets per seconds, and, as a consequence, the per-packet processing time is severely constrained. Even by adopting parallel processing architectures, the performance of such a device would still be limited by the contention for the access to the large memory banks where the packets are stored (network processing hardware usually implements a hierarchy of memory blocks characterized by different sizes and access latencies); access to such banks is generally associated with an high delay, thus representing the main contribution to the packet processing time. For this reason, the complexity of the protection policies which can be implemented in the font-end capturing device is mainly limited by the number of accesses to the packet data (and to other large data structures) which are required by the algorithms. Therefore some kinds of anonymisation techniques, for example those requiring the analysis of the packet payload, cannot be implemented in the font-end capturing device. On the contrary, tasks requiring only inspection of some header fields can take advantage from the parallel architectures and can be implemented very efficiently on network processing devices; traffic classification, for example, can be executed at very high data rates. These constraints imposed a more articulated design of the front-end system, making it composed by one (or more) special purpose hardware module(s) implementing the capturing device plus one (or more) general purpose modules for more complex processing tasks. Furthermore, interaction between the capturing device and the other processing devices is limited to the following data exchanges:

- The capturing device provides the devices with the captured data (or a portion of it), eventually extended by meta-data information
- Processing devices can communicate with the capturing device and set configuration options, such as cryptographic keys or classification rules

Other algorithms requiring more complex interactions cannot be supported by the probe.

2.1.2 On the Network

As the font-end block is conceived as composed not only of a packet capturing device, but also a packet pre-processing block (which, in turn, may consist in one or more general purpose units), the set of anonymisation techniques which can be implemented in the font-end can be significantly extended.
Since the tasks associated with packet capturing and classification are executed by the capturing device, pre-processing units can execute more processing intensive algorithms. In addition, if multiple pre-processing units are available, different portions of captured traffic can be multiplexed among different units, thus consistently offloading the computational burden of each unit and allowing for the implementation of selective protection policies for different kinds of traffic flows. Furthermore, since the pre-processing units can be considered as commodity PCs, every kind of existing library and tool can be used.
The communication of data from the capturing device to the pre-processing units is considered to be physically segregated from the public network. As a consequence, protection and security issues related to such a communication have not to be taken into account.
Performance requirements of the pre-processing units are much looser than those of the capturing device, since:

- They receive only a portion of the incoming traffic (a subset of the packets or a set of flows can be selected out of the whole traffic and if not necessary, packet payload can be completely cut off or reduced to a small portion)
- Packet time stamping and classification, together with simple data protection tasks, have already been performed by the capturing device
- Small packets can be concatenated by the capturing device into larger data block reducing transmission and packet handling overhead

Therefore the pre-processing units can support a large set of encryption and anonymisation techniques.
Some extremely demanding privacy preserving data mining techniques, such as k-anonymisation, cannot be implemented even in this portion of the font-end block, since they generally require the knowledge of the whole available data set and the envisioned font-end does not appear to be fit to retain and manage such a large amount of data.

2.2 PRISM Backend System

2.2.1 Privacy-Aware Access Control

The specification of access control models specifically tailored towards privacy protection has been the focus of intense research in the last few years. To that respect, several approaches have been proposed, that go beyond conventional Role-Based Access Control (RBAC) [FER01], incorporating different criteria in access control decisions, rather than just which user, having which role, is performing which action on which data object.

One of the most influential milestones in privacy-centric access control literature has been the concept of Hippocratic Databases [AGR02]. Focusing on the database level, Hippocratic

Databases offer mechanisms for the enforcement of privacy rules, conceived on the basis of privacy legislation. Hippocratic Databases introduce the notion of purpose as a central concept; purpose is stored in the database as an attribute of every table. When a query is submitted to the database, the system not only authorizes its execution according to the role of the user that submits it, but answers only queries for which the purpose is equal to that for which the information has been collected. Therefore, by means of SQL rewriting, the database is enabled to automatically return the subset of records where usage is authorized, based on privacy rules evaluation. Several enhancements have been proposed to the concept of Hippocratic Databases, in order to improve it towards the directions of the introduction of hierarchies in databases and purposes, the delegation of tasks and authorizations and the minimal disclosure of personal data [LEF04, MAS05, MAS06].

Another access control model for privacy protection based on the notion of purpose and focusing on relational databases is presented in [BYU04]. This work introduces a sophisticated approach in purpose definition, relying on a hierarchy of purposes and on a Purpose Ontology. In order to associate intended purposes with data, four different data labelling schemes are proposed, each providing different granularity. The access policies may be either positive (i.e. selectively allowing access) or negative (i.e. explicitly prohibiting access to data for certain purposes), while data filtering is supported by query modification techniques. This work has been extended in order, on the one hand, to support access to more complex objects and, on the other hand, to enable access authorization based on users' role [BYU05]. Therefore, an expansion of the RBAC model is specified.

The OASIS eXtensible Access Control Markup Language (XACML) [OASIS] is a general-purpose access control standard that describes both a policy language and an access control decision language, both written in XML. The policy language is used to describe general access control requirements and has standard extension points for defining new functions, data types, combining logic, etc. The decision language enables the specification of queries regarding whether or not a given action should be allowed. XACML uses the abstract PEP/PDP [YAV00] model defined by the IETF for authorizing access to a protected resource. With respect to privacy, the Privacy Policy Profile of XACML v2.0 [MOS05] defines standard XACML attribute identifiers for expressing the purpose for which data are collected and being accessed, as well as rules for requiring consistency between the purpose for which data are collected and the purpose for which data are being accessed. Additionally, other XACML profiles support integration with RBAC, hierarchical resources, etc.

The Enterprise Privacy Authorization Language (EPAL) [ASH03] is a formal language for writing enterprise privacy policies to govern data handling practices. It has been proposed by IBM and, similarly to XACML, it uses the abstract PEP/PDP model. EPAL refines previous milestone work of IBM [KAR02a, KAB02b, SCH02, KAR02c]. Under EPAL, the privacy policies are expressed in a way as to be enforceable by an access control system. Policies expressed in EPAL consist of a series of rules expressing the right of some actor to perform some action on some object, subject to certain conditions and obligations. An element specifying purpose makes permission conditional on the action being performed for some particular purpose.

Recently, Hewlett-Packard has presented a framework for privacy-aware access control [MON05a, MON06]. It specifically focuses on addressing the problem of privacy policies enforcement on personal data stored in a broad variety of data repositories within enterprises. Leveraging and extending the commercial access control solution HP Select Access, it offers mechanisms for the explicit modelling of personal data and privacy policies authoring along with access control policies, an authorization framework for deploying both access control

and privacy-based policies and making related access decisions, as well as a mechanism for interpreting at run-time attempts to access personal data and enforcing decisions based on privacy policies, users' privacy preferences and contextual information. The framework is complemented by the means for the explicit management and enforcement of privacy obligations, including automatic data deletion, data transformations and notifications to the users [MON05b].

The IST project DISCREET [DISCREET] has proposed a framework for the enforcement of the European privacy legislation. It is based on the concept of a privacy proxy that mediates between the users and the service providers, and in fact negotiates the delivery of personal data. At the core of the privacy proxy is the Ontology of Privacy, a set of concrete rules and policies that reflect regulatory requirements about privacy into service provision.

2.2.2 Database Security

Enhancing the security of a database is becoming one of the most urgent tasks in database research and industry. Generally, database security methods can be divided into four layers:

- Physical security,
- Operation system security,
- Database Management System (DBMS) security and
- Database encryption.

These layers protect a database in different aspects. The three first layers alone are inadequate to protect the confidentiality of data in the database to a satisfactory degree, because the data are still stored in readable form. That is, without database encryption, it is impossible to guarantee that the sensitive information in plaintext will be protected against a malicious user who has super-user power, such as a database administrator (DBA). In that way, a database server will be compromised as long as the DBA account is compromised. Besides, data are not always static; they can also be in move. Therefore, two main issues need to be considered: secure data storage and secure data transmission. Database encryption is a technology that introduces an additional security layer to the traditional database management system. It prevents exposure of sensitive information even if the database server is compromised. Furthermore, database encryption can be employed to maintain the data integrity, ensuring that even a little modification made on the data can be detected. Database encryption technologies meet the data confidentiality requirements and have become an indispensable aspect of enterprise database security.
A lot of research work has been carried out and various security mechanisms have been proposed and implemented. These mechanisms include authentication, access control, encryption, audit, intrusion detection, etc. Database encryption is supported by almost all commercial DBMS. Current work on database encryption focuses on data privacy and efficiency; however, the way to share encrypted data securely in database systems is lacked. Without support from the underlying DBMS, it is difficult to find flexible and secure sharing approach, in that a user should share neither password nor encryption keys with others.

However, developing a sound security strategy including database encryption still involves many open issues. Key management and security are of paramount importance in any encryption-based system and were therefore among the first issues to be investigated in the framework of database encryption [DAV81], [HAC04]. Later, techniques have been developed aimed at efficiently querying encrypted databases [SON00], some of them related to parallel efforts by the text retrieval community [KLE89] for executing hidden queries, that is, queries where only the cipher text of the query arguments is made available to the DBMS. On the other hand, architectural research investigated optimal sharing of the encryption burden between secure storage, communication channels and the application where the data

originates [JEN00], looking for a convenient trade-off between data security and application performance. Recently, much interest was devoted to secure handling of database encryption in distributed, Web-based execution scenarios, where data management is outsourced to external services [BOU02]. The main purpose of this line of research is to find techniques for delegating data storage and the execution of queries to external servers while preserving efficiency. The index of range technique proposed in [HAC02] in the framework of a database-service-provider architecture relies on partitioning the domains of attributes in client tables into sets of intervals. The value of each remote table attribute is stored as the index countersigning the interval to which the corresponding plain value belongs. Indexes may be ordered or not, and the intervals may be chosen so that they have all the same length, or are associated with the same number of tuples. This representation supports efficient evaluation on the remote server of both equality and range predicates; however, it makes it awkward to manage the correspondence between intervals and the actual values present in the database. In [DAM04], an approach for obfuscating data that guarantees protection of data while allowing the execution of both equality and range queries on the obfuscated data is presented. Privacy homomorphism has also been proposed for allowing the execution of aggregation queries over encrypted data [Hacig04]. The proposed approach is based on the technique introduced in [RIV78] according to which an encrypted function is homomorphic. In this case, the server stores an encrypted table with an index for each aggregation attribute (i.e., an attribute on which the aggregate operator can be applied) obtained from the original attribute with privacy homomorphism. An operation on an aggregation attribute can then be evaluated by computing the aggregation at the server site and by decrypting the result at the client side. Other work on privacy homomorphism illustrates techniques for performing arithmetic operations on encrypted data and does not consider comparison operations [BOY03, DOM96, and DOM98]. In [AGR04], an order preserving encryption schema (OPES) is presented to support equality and range queries as well as max, min, and count queries over encrypted data. The basic idea is that given a target distribution, the plaintext values are transformed by using an order-preserving transformation and in such a way that the transformed values follow the target distribution. A distinct, though related solution is proposed in [Bouganim02], where smart cards are used for key management.

An access control model called Share Secret Securely Role-Based Access Control (3S-RBAC) extended from the Role-Based Access Control (RBAC) model is also proposed. New concepts and features are introduced in 3S-RBAC, including: the novel concept of strong and weak permission; the hierarchy of database objects and keys; the permission and key inheritance; the binding of keys and permissions. These features make this approach secure, general, flexible and practical.

Furthermore the European Project PRIME [PRIME] focuses on solutions for privacy-enhancing identity management that supports end-users' sovereignty over their private sphere and enterprises' privacy-compliant data processing. The PRIME console’s data track function maintains a database of the personal data disclosed by the user. It provides a comprehensive overview of what personal data the user has released to whom, under which partial identity (pseudonym), when, and for what purpose (i.e. under what policy). This functionality requires the implementation of PRIME Middleware at the user's side and the server's side. The most powerful function of the PRIME concept is the technical enforcement of agreed policies on the service's side when equipped with PRIME enabled middleware. The machine-readable part of the sticky policies can be processed automatically by the PRIME server Middleware. The system will detect the fulfillment of certain conditions that warrant action on the user's data.

2.2.3 Trusted Computing

The term “Trusted Computing” (TC) refers to a technology introduced in the very months by the Trusted Computing Group (TCG), in which Personal Computers, consumer electronic devices, PDAs and other mobile devices are equipped with a special hardware chip called Trusted Platform Module (TPM). In accordance with other security hardware extensions, the TPM is empowered with cryptographic mechanisms to (1) certify remotely the integrity of the (application/system) software running on the device, (2) to protect I/O and storage of data inside the device and, (3) to strictly isolate the data residing inside memory from other potentially malicious applications.

In essence, TC systems would cryptographically seal off the parts of the computer that deal with data and applications and give decryption keys only to programs and information that the technology judges to be trustworthy. The basic idea behind the concept of Trusted Computing is the creation of a chain of trust between all elements in the computing system, starting from the most basic ones. Therefore, the chain starts with the tamperproof hardware device, known as Trusted Platform Module (TPM), which analyses the BIOS of the computer and, in case it is recognized as trusted, passes control to it. This process is repeated for the master boot record, the OS loader, the OS, the hardware devices and finally the applications. In a Trusted Computing scenario a trusted application runs exclusively on top of protected and pre-approved supporting software and hardware.

This practice is well designed to effectively fight against malicious code, viruses, privacy violations, etc. Online content providers could also use the systems’ ability to prevent computers from accessing applications and data to keep people from listening to, copying, or otherwise using intellectual property in ways that providers don’t want. The reason is that current practices for fighting against malicious code and other threats purely at the software level by their very nature are uncompromising. Indeed, it has been learned from past experience that a trusted and tamper-proof security basis cannot be achieved using software-based solutions alone.

A comprehensive defense against the security threats faced by PC users should involve several approaches, not just one. An insecure system can't magically become "secure" with the addition of a single piece of technology. Changes to the design of PC hardware are one useful tool among many for improving security. While hardware changes aren't a prerequisite for increased security, they're undeniably helpful – for example, by providing a way to store private keys (and therefore the documents protected by those keys) safely.

Proponents say the initiatives will increase users’ trust in their ability to protect their systems from malicious code and guard their data from theft. Opponents, on the other hand, contend the projects provide security by giving TC technology control that users should have over their own machines. They say this gives the vendors too much power over computing platforms, which they could abuse to help their own bottom line. Detractors also say that trusted computing’s intellectual property protection capabilities unfairly favor online content providers, often partners with TC vendors, over consumers. Nonetheless, vendors are moving forward with trusted-computing technology, so the key issue is whether businesses will adopt it.

The broad term "trusted computing" includes a mix of initiatives by individual processor manufacturers and OEMs (Original Equipment Manufacturers), along with particularly well-known larger projects.

Companies have established three major trusted-computing initiatives: Microsoft’s Next-Generation Secure Computing Base (NGSCB), formerly known as Palladium; Intel’s LaGrande; and the Trusted Computing Platform Alliance (TCPA), an industry work group of more than 190 companies. Wave Systems has released a cryptography chip for trusted computing and Microsoft is already developing its system. The major TC initiatives differ primarily in where the encryption/decryption functionality occurs.

In NGSCB and LaGrande, the TPM is incorporated into the main CPU, thereby avoiding the problem of unencrypted data going over the data bus to the dedicated processor. However, this would require new CPUs that have the encryption/decryption functionality built in. In contrast, TCPA and Wave Systems' Embedded Application Security System (Embassy) move the workload from the CPU to a special-purpose chip.
Of the three major TC projects, NGSCB is closest to deployment. The NGSCB project specifies software changes that take advantage of the security benefits made available by a planned new PC hardware design.
The other well-known project is a hardware specification project run by a consortium originally called the Trusted Computing Platform Alliance, or TCPA. TCPA issued several specification documents and then changed its name to the trusted computing group, or TCG.
Between them, these two projects have created a bewildering array of new terminology, including the obligatory thicket of new acronyms. In several cases, one of these projects has devised many different names for a single concept even as the other project has its own entirely different terminology. In the interest of simplicity, the requirements of NGSCB are converging with the features of the design specified by TCG. (Microsoft is a TCG member and has expressed an interest in using the TCG design in the role of the hardware components required by NGSCB.) Some OEMs have begun to integrate early TCG chips onto their computers' motherboards; in the future, more computer manufacturers may include future versions of trusted computing circuits in their PCs. The NGSCB software would be one application of several which could take advantage of the features of these chips. While these projects are still distinct, it is reasonable to speak of a single "trusted computing architecture" toward which both projects are headed. (Only a portion of this architecture is described by the most recently published TCG specification, and, as TCG notes, additional software will be required to make use of many of these features.)
The less well known trusted computing projects under development by processor vendors (and TCG members) Intel and AMD may fill in some of the gaps between what TCG has so far specified and what NGSCB would require. Intel's LaGrande Technology (LT) and AMD's Secure Execution Mode (SEM), for example, provide hardware support needed for all the major feature groups in NGSCB. Their features would build on TCG features to provide the hardware support demanded by NGSCB. One important similarity between the NGSCB design and the existing TCG specification is that both contain a "remote attestation" feature, which we will criticize extensively below. Even though there are differences between Microsoft's and TCG's technical descriptions of remote attestation, both can, given proper operating system support, be used in functionally equivalent ways.
Furthermore the Open Trusted Computing [OPENTC] programme funded under 6th FP (Sixth Framework Programme) is based on security mechanisms provided by low-level operating system layers with isolation properties and interfaces to Trusted Computing hardware. These layers allow leveraging enhanced trust and security properties of the platform for standard operating systems, middleware, and applications. The suggested architecture is applicable to a wide range of platform types, e.g. servers, GRID technology, mobile phones and industrial automation. It provides basic building blocks for complex, distributed scenarios with inherent, multilateral trust and security capabilities. To enable maximum community benefit, project results will be integrated in and distributed as Open Source software, supporting Linux in particular.
Trusted computing technology can't prevent computer security holes altogether. In general, it seeks to contain and limit the damage that can result from a particular flaw. For instance, it should not be possible for a coding flaw in one application (like a web browser) to be abused to copy or alter data from a different application (like a word processor). This sort of isolation and containment approach is an important area of computer security research and is used in many different approaches to computer security, including promising techniques outside of trusted computing. The trusted computing features will add new capabilities to the PC. To be

used, they must be supported by software; in the absence of trusted computing software drivers, the trusted computing PC is just an ordinary PC, which remains capable of running all existing PC software. To put this way, the trusted computing architecture is designed to be backwards-compatible in supporting the ability to run existing operating systems and application software. Microsoft also anticipates that future versions of Microsoft Windows (which could include NGSCB software) would be backwards compatible, able to run essentially all of today's DOS and Windows applications. In addition, the new PCs could run new trusted-computing-aware applications that take advantage of the new hardware features.

3 State of the Art in Data Protection

This chapter reviews and presents information about state of art with respect to anonymisation and data protection geared towards Internet packet traces. It sheds a light on what the sensible information is that shall be protected, what the techniques and algorithms are that are in use to apply this protection and what they effort and drawbacks are when it comes to withstanding malicious reverse engineering attacks. We will see that a multitude of options are available – but these need to be applied in the right dose to find a worthwhile compromise between security and the usability of the anonymised data sets.

Sensible Parts of Packet Traces
Inside a raw packet data trace there are different data fields and traffic characteristics that might be considered sensible and are therefore subject to anonymisation.
We can differentiate between direct and indirect traffic data:
(see Appendix B for the definition of direct and indirect data)
Direct traffic data are:

- data fields from the LLC/MAC header and IP header, such as
  - packet length
  - IP addresses
  - IP ID field
- data fields from the UDP or TCP header, such as
  - TCP flags
  - TCP sequence numbers (TCP timestamps?)
  - port numbers
- the capture time stamp associated to each packet
- the application level datagram payload

Indirect traffic data are those metrics or statistics derived from the direct data. Among them are traffic characteristics such as:

- number of distinctive IP addresses (per time interval)
- traffic bandwidth, jitter, or loss (per time interval)
- number of traffic flows active in parallel
- top 10 endpoints (IP addresses) or top 10 applications in terms of data volume
- number of TCP connection requests, successful connects, or even e.g. ssh logins
- network structure and addresses of distinctive hosts (server, routers, gateways, etc.)

Other indirect statistics are conceivable.
Anonymisation processes work on removing or replacing this information in a non-reversible way, so that vital characteristics are still kept intact.

3.1 Packet Data Anonymisation

3.1.1 One-Way

The techniques to perform anonymisation on direct or indirect trace data can be classified into four groups:

- Removal
- Replacement (remapping), also partial
- Reduction of accuracy
- Aggregation

3.1.1.1 Removal

This is the most simple anonymisation technique. An attribute which shall not be disclosed is simply removed from the packet trace data. Data removal is frequently applied to IP addresses

and/or port numbers. To keep the structure of the datagram headers in the trace data intact the fields are then replaced by zeroes. Logically we still consider this the removal of the information. The data most often removed is the application level packet payload. Often packets are captured with a so-called snap-length, only storing the first n bytes. This usually truncates the payload somewhere in the beginning of the application level data. We therefore can call it a partial removal of the payload. It is also common to remove the payload completely and keep packet data until the end of the IP/UDP/TCP header (whichever is the last one before the application data).

3.1.1.2 Replacement

A replacement technique substitutes one or more attributes while working on the traffic data trace. The replacement strategy includes changing the values of the fields (e.g. IP addresses) while keeping valuable information, such as the number of distinct addresses and their order intact. Replacement is most often performed on IP Addresses and port numbers. The most common algorithm constructs a 1-to-1 mapping of original to replacement addresses (usually 1.1.1.1, 1.1.1.2, 1.1.1.3 and so on) in a way that packets which had the same address before the anonymisation still have the same address afterwards. Other algorithms shuffle parts of the data (e.g. last 6 bytes of MAC address) to hide the original sender and recipient of a packet. Shuffling can keep the uniqueness of attributes too when applied as a fixed mapping. After processing the data trace the resulting mapping information - which now contains the sensible information - is discarded. For replacement of IPv4 addresses algorithms exist which also keep the distribution into addresses of class A, B, and C networks intact (see paper link below).

3.1.1.3 Address Anonymisation

Since, in general, the IP address can uniquely identify a host, it definitely cannot be left in clear in a traffic trace.
A trivial anonymisation method could then be to substitute each IP address with a random pseudonym, that can be obtained by using a cryptographically secure random number generator; the Blum Shub generator is considered secure, but it is computationally demanding, since it deals with module algebra and requires an iteration for each bit of the address. Hardware number generators, which are in some cases provided by the network processing devices, constitute an alternative and fast source of secure random numbers. However, in order to keep the anonymised traffic trace coherent, it is necessary to keep track of the association between the real and anonymised values of the protected fields; depending on the number of different values in the trace, that can lead to a consistent memory occupation.
Another solution is to compute the anonymised address as a cryptographic hash of the original one; such an operation can be significantly accelerated by using the hardware crypto-units which are provided by several network processing platforms.
However, a simple encryption of the original IP addresses can hide a large amount of important information which is provided by the hierarchical nature of IP addresses; a common prefix shared by several packets can reveal that a large part of the traffic crossing a given link is generated by hosts belonging to the same network.
For this reason, algorithms for prefix preserving anonymisation have been devised; such algorithms anonymised IP addresses in a way that, if two original addresses share the first K bits, the anonymised addresses will also share the first K bits.
A pseudo prefix preserving anonymisation scheme, based on secret key and blowfish encryption, has been proposed by Pehkuri [PEU01]; however, such a scheme preserves only the first 8 bits of the original address, while the others are mapped in a non prefix-preserving way.
Another solution for prefix-preserving anonymisation is adopted by the well known tool TCPdpriv [MIN97]; this software is one of the most well known anonymisation tools; it

operates on pcap traces, defining several different levels of anonymisation for each field of the packet header and discarding the packet payload. It also implements a prefix-preserving anonymisation scheme based on random number generation. When it finds a new address X in a trace, simply performs a search over the list of the addresses which have already appeared in the trace; as a results, the address X' which shares the longest prefix with X is returned.
Let Y' be the anonymised version of X' and Y be the anonymised version of X: if X and X' share the first k bits, the first k bit of Y will equal those of Y', while the rest of the original address will be randomized in a non prefix preserving way. Such an approach has two main drawbacks:

- If the trace is large and a lot of different addresses appear, a lookup in the table can take a long time
- The mapping of the addresses depends on the order in which they appear in the trace: consistent mapping among different traces is not guaranteed.

Another approach is based on the so called canonical form theorem [XU02], which formally proves that Prefix preserving anonymisation requires the calculation, for the k-th bit of the original address, a cryptographic function of the previous k bits of the original address; for 32-bit addresses such a calculation turns out to be unfeasible in real time.
Several proposals have been made to make such an approach scalable ([RAM07], [ZHA06]). Most of them take advantage of the fact that prefix preserving anonymisation can be performed as a search on a precomputed complete binary tree, whose depth equals the number of bits to be anonymised. Such a solution presents the drawback of a consistent memory occupation; in addition, in many network processing architectures, such an algorithms still requires too many memory lookups (one for each level of the tree) to be performed at line speed.
In order to reduce both the memory occupation and the number of lookups, the TSA algorithm has been devised [RAM07]: the first 8 bit of the addresses are anonymised in a non prefix-preserving way (which generally convey very little topological information), but rather using an hash function, while the remaining bits are mapped by using the approach which has been described above. Such an idea, integrated with the tree replication idea, leads to a total of 24 memory lookups and a memory occupation which can reach down to 2 megabytes (depending on the particular cryptographic function); such a reduced occupation would allow to store the tree in one of the fastest memory blocks (generally SRAM blocks) which are available on the network processing platform, thus allowing fast lookup.
The authors state that, by using a multibit-trie lookup scheme, the anonymisation can be performed with a total of 4 memory lookups, but at the expense of a higher memory occupation (which is not quantified in the paper).
Even with such an improvement, implementing prefix preserving anonymisation at line speed over current network processing platform appears to be a challenging task; a further optimization of the address mapping scheme, taking advantage of the highly symmetric structure of the tree, would greatly enhance the performance achievable by the PRISM measurement probe.

3.1.1.4 Anonymisation of other Packet Header Fields

Several kinds of attacks, which take advantage of header fields other from the IP addresses in order to infer the identities of certain users, have been proposed in the literature ([PAN06], [PAN03], [SEE07]).
For this reason some anonymisation policies can require to process different fields of the headers, such as the port number or the TCP options (the timestamps included by TCP, for example, can be exploited to effectively fingerprint a host).
In some cases, such fields can simply be black-marked (replaced with a fixed value such as a sequence of zeros).

Another possible way of anonymising a packet field is to perform a random permutation over it; such an operation preserves some information about the original value (the number of 0 and 1 if the permutation is performed at the bit level). Permutation of small sequences can be performed in the local memory of a processor and it presents no issue for line speed processing; however, if a random number must be generated in order to be used as one time pad for the permutation, the delay introduced by the random number generator must be taken into account.
However, since such fields are generally small, they can be mapped to random pseudonyms by using a small table which can be kept on a fast memory block (for the port number such a table would have an extension of the order of 100kb) and would require only one memory access for lookup.

3.1.1.5 Timestamp Modification

As it has been documented in the literature, accurate timestamps (especially when expressed as UTC times) of packets can be used in order to discover specific patterns and to infer information about the users. As an example, [SON01] states that, by analyzing the interarrival times of packets sent by remote-terminal applications such as Telnet, it is possible to gather information which is useful to crack passwords.
Reduction of timestamp accuracy is a mild form of anonymisation whereby some of the lower significant digits of an attribute are removed, i.e. set to zero. This can be useful to hide jitter or burstiness effects from analyzers of a trace.
Several modifications of time stamps have been proposed in the literature [SLA06]:

- Time unit annihilation: deletion of the portion of timestamp referring to a given time unit, (such as minutes, days, seconds…)
- Random offset: addition of a random offset to each timestamp of the trace, in order to hide when the trace has actually been captured

Another approach can be to simply replace the per-packet timestamp with cumulative information about the packets which have arrived in a given period.
All of these approaches, since they require neither any complex calculation nor any memory lookup, represent no issue for implementation in a probe.

3.1.1.6 Payload Modification

The payload of the captured packets, although extremely useful for purposes such as intrusion detection, likely contains a lot of personal information, which often appears in association with short strings such as “passwd”, “root”... Such strings often precede other strings which convey sensitive information (pathnames, URLs passwords). Once such strings have been located, several methods for their anonymisation can be applied, such as MD5 keyed hashing [MIL03].
The task of recognizing such short string within the packet payload appears to be even more demanding in the case of stream-oriented applications, such as those implementing HTTP, and FTP. In that case the transaction involves several TCP connections and a lot of different packets.
Existing anonymisation tools (such as [KUO06]) provide a COOK primitive to reconstruct the original application level transaction from the packets included in the trace and an UNCOOK primitive to partition the modified transaction into multiple packets, together with protocol specific parsers to easily identify sensitive fields. However, this kind of application is not designed for on line trace anonymisation, but rather for off-line modification of already captured trace. Implementing this kind of design over a traffic capturing probe is clearly unfeasible, both in terms of computational complexity and memory requirements.
Even searching every single packet for strings revealing the presence of privacy sensitive data constitute a very time consuming task at gigabit rates, since each character of the packet payload has to be read and compared against a particular matching data structure (in general a

Deterministic Finite Automaton). Therefore, for each character, an access to the packet payload and (at least) one to the data structure implementing the DFA are required.
Furthermore, after such a check, strings spanning two consecutive packets cannot still be revealed.
[ZAM06] proposes a method for scrambling the payload of a packet, while keeping the possibility to reveal the presence of small strings for intrusion detection. Such an algorithm, however, involves two permutations of the whole payload of the packet, leading to a high number of memory lookups (furthermore the payload of the packet must be kept in a large and slow memory block, such as a DRAM bank) and, therefore, is not suitable for on line anonymisation.
To our knowledge, no techniques have especially been proposed for online payload anonymisation; even the LOBSTER project tool, which implements a hardware FPGA based platform for anonymisation ([UBI06]), postpones payload inspection to software based post processing.
However, if releasing the captured traces from the probe with non-protected payloads is considered not to be acceptable for the purposes of the project, reversible encryption of the payload can be considered.
Such an operation, however, is quite demanding, even if it can be accelerated by the presence of a crypto unit on the network processing hardware and would likely cause a sensitive reduction of the probe performance. As a benchmark, it can be estimated that the encryption of 16 bytes involves a latency which is of the order of that of a memory lookup.

3.1.1.7 Aggregation into Derived Data

Aggregation of packet data traces denotes the process of summarizing information for groups of packets with common attributes (or a common set of attributes). Aggregation is often performed after packets have been classified into traffic flows of a selected granularity (e.g. host-to-host, network-to-network). The aggregation process analyzes the packets and only keeps the common attributes and those metrics derived from the packets of each group (e.g. host A to host B, port X to port Y: volume=100kBytes, peakDataRate=1.4Mbps). The resulting information (in our example: volume and data rate) are called “derived data” within the PRISM project. Aggregation semantically changes the traffic data trace because it takes a packet trace as input and generates a flow data trace with additional derived flow characteristics as output. Of course the remaining raw packet data (addresses, ports, etc.) can be further anonymised using the previously mentioned techniques.

3.1.2 Cryptographic Two-Way

In the last years a diversity of approaches based both on symmetric and asymmetric cryptography has been proposed in order to allow data mining procedure without violation for the user privacy. In particular, provably secure approaches have been presented in order to allow the best protection available to sensitive data. The can provide features such as:

- Secure multiparty computation
- Private information retrieval
- Private regular expression matching
- Searches on encrypted data

The secure multiparty computation allows different entities to evaluate some function without a need to share the input that each entity passes to the function itself with the others. That is, one can compute $f(x_1, x_2, ..., x_n)$ without sharing the xi. The private information retrieval procedures could be employed to obtain information stored in a database administered by a “curious” system administrator. The private regular expression matching is used to match regular expression over data owned by someone else.

All these protocols are designed for scenarios where the performance requirements are less strict than with the real time traffic monitoring. Nonetheless, they provide a good starting point for solutions that are applicable in the context of the PRISM project.
These techniques are described in more detail in the following sections. After a brief introduction to the basic concepts of the cryptography, we describe the homomorphic properties of some cryptographic algorithms, and the oblivious transfer protocols that constitute the building blocks of many of the aforementioned approaches. Furthermore, the advantages and limitations of each protocol within the PRISM framework are discussed.

3.1.2.1 Traditional Methods

The traditional encryption/decryption techniques rely on some ciphering function which transforms a given plaintext to cipher text. Ideally it is impossible to revert the operation unless one has a secret key in possession. The encryption methods can be divided into two well-known classes, symmetrical and asymmetrical, where in the former the same key is used both for encryption and decryption, and in the latter two different keys are involved. In particular, the asymmetric cryptography is often referred to as the public key cryptography as it allows sharing one of the keys publicly (depending on the purpose). Typical examples of symmetric ciphers include DES and AES (see, e.g., [Sch96]), and for asymmetric RSA [RSA78] and ElGamal [ElG85].
In the context of PRISM project, the stored trace files are envisioned to be stored in encrypted form to a so-called back-end component. The cryptographic secrets needed to decrypt the data are stored into another component referred to as privacy preservering controller. In particular, the trace data will be only decrypted when needed, e.g., for data retention or trouble shooting purposes. In these cases, it is important to provide only that part of the information which is i) needed for the task in question, and ii) allowed by the local legislation (e.g., the payload data without court order is private and should not be processed in any way.).

3.1.2.2 Homomorphic Encryption

Homomorphic encryption (or privacy homomorphism) refers to all encryption algorithms which allows one to perform some algebraic operation (or operations) on cipher text instead of plaintext [Den82, FG07]. Formally,

$$E(m_1 + m_2) = E(m_1) * E(m_2),$$

where + and * denote some possibly different algebraic operations. Perhaps the most familiar example is the RSA algorithm, where the encryption step corresponds to computing a modular exponentiation,

$$E(m) = m^e \bmod n.$$

This encryption scheme is clearly homomorphic with respect to multiplication, i.e.

$$E(m_1 \cdot m_2) = E(m_1) \cdot E(m_2) \bmod n.$$

A common example application for the homomorphic property are the secure privacy preserving election systems where the final aim is to tail the votes given by the eligible voters while, at the same time, making it impossible to infer the individual votes (see, e.g., [CGS97]).
Actually all the public key cryptosystem owned homomorphic property with respect to one operation. In particular the ElGamal algorithm [ElGamal84] allows one to come up with a homomorphic scheme with respect to addition operation. Well-known homomorphic schemes

include Paillier's [Pai99] (generalized in [DJ01]) and Benaloh [Ben94] cryptosystem, both homomorphic with respect to the sum, and Goldwasser-Micali [GM82] homomorphic with respect to the exclusive-or. None of the existing algorithms are "doubly homomorphic", i.e. there is not an algorithm homomorphic with respect to two different algebraic operations. Boneh & all presented in [Bon05] an algorithm based on the Paillier cryptosystem able to be homomorphic for addition and for (only) one multiplication.
An interesting proposal is by Domingo-Ferrer [DF02], where the author claims that the proposed additive and multiplicative privacy homomorphic system would be provably secure. Unfortunately, this did not turn out to be the case, but instead Wagner, in [Wag03], has shown some flaws in the logic. Similarly, in [YLP08], Yu et al. also consider the topic of how secure the homomorphic schemes can be by relying on a black-box model. In particular, it seems that homomorphic encryption has some really strong inherent constraints, which might make it infeasible in many scenarios [Den82, FG07].
Nonetheless, the homomorphic property or the privacy homomorphism are very appealing concepts for the PRISM project. In an ideal case such schemes may allow one to perform some data analysis tasks on the encrypted data without decrypting it in the process, thus improving the level of privacy considerably at the same time. For example, the problem of secure election, where the main task is, e.g., to count the YES ballots can be depicted in the framework of data analysis. A naive example could be to compute the amount of packets sent by a specific IP address without the need to reveal the real IP if the user is not flooding the network.

3.1.2.3 Oblivious Transfer Protocol

An oblivious transfer protocol (often abbreviated OT) allows a user to receive some information from a sender in a way that the server is unable to know what is received. The first form of oblivious transfer was introduced in 1981 by Rabin [Rab81] based on the RSA cryptosystem. A more useful form of oblivious transfer, called 1-2 oblivious transfer or "1 out of 2 oblivious transfer", was developed later by Shimon Even, Oded Goldreich, and Abraham Lempel [EGL85], where the aim was to design protocols for secure multiparty computation. It has also been generalised to "1 out of n oblivious transfer" where a user obtains exactly one database element while the server is unable to infer which element was queried. Another interesting approach has been proposed by Pinkas and Naor in in [Noa01,Noa00].

The following example is taken by [GOL04]:

Inputs:

- the sender has input $\{\sigma_1, \sigma_2, \ldots\ldots, \sigma_k\} \in \{0,1\}^k$,
- the receiver has input $i \in \{1,2,\ldots..,k\}$
- both parties have the auxiliary security parameter 1^n

Step 1: the sender uniformly selects an index-trapdoor pair (α,t), by running the generation algorithm G, on input 1^n . That is, it uniformly selects a random-tape, r, for G and sets $(\alpha,t)= G(1^n,r)$ It sends the index α to the receiver.

Step 2: The receiver uniformly and independently selects $x_1, x_2, \ldots\ldots, x_k \in D_\alpha$, computes $y_i=f_a(x_i)$ and $y_j=x_j$ for every $i \neq j$ and sends $y_1, y_2, \ldots\ldots, y_k$

Step 3: The sender computes $z_i=f^{-1}(y_i)$ for every $j= \{1, 2, \ldots\ldots, k\}$ and sends $\sigma_1 \oplus b(z_i)$, $\sigma_2 \oplus b(z_2)$, ……, $\sigma_n \oplus b(z_n)$

Step 4: The sender computes $\sigma_i \oplus b(z_i) \oplus b(x_i) = \sigma_i \oplus b(f^{-1}(f_a(x_i))) \oplus b(x_i) = \sigma_i$

The incomplete description of the protocol proposed by Goldreich shows the main limitation of the oblivious transfer protocols: in order to receive one piece of information a user has to send and receive k different messages. Consequently, the performance degradation and the

bandwidth consumption with this kind of protocols is likely to be unacceptable for the anticipated PRISM scenarios.

3.1.2.4 Searchable Symmetric Encryption

In networked infrastructures one often stores files to dedicated file servers. In some cases, these servers are outsourced to a third party, and, consequently, the files must be encrypted in order to avoid disclosure of the private and/or sensitive data. However, it may not be feasible to download all files to a local machine when some relatively simple task needs to be carried out. One of the most common tasks is to find a correct file for further processing, which often involves searching files for a certain keyword or keywords. Thus, if the data servers are high performance systems when compared to the network capacity, it would be extremely useful if the documents could be searched locally at the server without decrypting the documents.
This basic problem has been recently studied in [SON00, CGKO06]. In particular, the problem considered is as follows:

1. Alice wants to store some documents to server of untrustworthy party Bob in encrypted form.
2. At some moment of time, she needs to check/retrieve documents containing a certain keyword.
3. For performance reasons, the searching task through the documents will be carried out by Bob.
4. However, due to privacy reasons Alice does not want that Bob learns anything else but the search result in the process.

The solution (or a serie of solutions) proposed by Song et. al, in [SON00], utilizes extensively the three-way symmetry of exclusive-or (XOR) operation, denoted by

$$a \oplus b = c \quad <=> \quad b \oplus c = a \quad <=> \quad a \oplus c = b.$$

Recall that also the one-time-pad cryptosystem achieving the perfect secrecy simply combines the plaintext with an ideal pseudorandom sequence using the exclusive-or [Sch96].

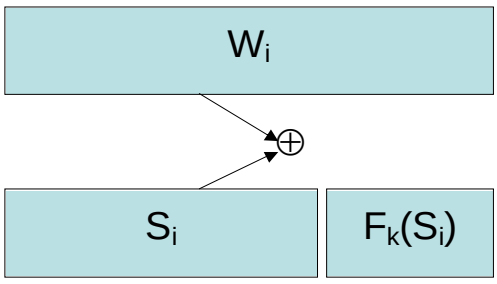

Figure 1: The initial encryption scheme for a searchable cipher proposed by Song et. al in [SON00]

Encryption: The “final (encryption) scheme", as proposed in [SON00], is illustrated in Fig. 2. The steps are as follows:

1. Let W_i denote the ith block of plaintext, each block having a length of n bits.
2. The initial pre-encryption, $X_i=E(W_i)$ obscures the plaintext resulting a n bit long word. Note that each block goes through the same transformation, i.e., two identical input blocks remain identical after the transformation, $E(W_i)=E(W_j)$ when $W_i=W_j$. This step is important as it allows one to outsource the searches without revealing the actual search key, i.e., Alice tells E(W) to Bob instead of W.

3. Then, the pre-encrypted block $X_i=E(W_i)$ is divided into m bits long lower part and n-m bits long higher part, denoted by L_i and R_i, respectively,

   $$X_i = <L_i,R_i> = R_i+2^m L_i$$

   and Alice calculates a key k_i based on the L_i, $k_i = f_{k'}(L_i)$, where k' is a secret key held by Alice. The division to higher and lower parts makes is possible for Alice to decrypt the ciphertext (see below).
4. Then Alice obtains n-m pseudo random bits, denoted by S_i, and forms a n bit long word, $Y_i = < S_i, F(k_i,S_i) >$, where $F(k_i,S_i)$ is a pseudo random function with parameter k_i.
5. The ciphertext is obtained by taking an exclusive-or between $X_i=E(W_i)$ and $Y_i = < S_i, F(k_i,S_i) >$,

   $$C_i = X_i \oplus Y_i = E(W_i) \oplus < S_i, F(k_i,S_i) >.$$

Decryption: Alice knows both the pseudo random sequence S_i and the ciphertext C_i.

The three-way symmetry of exclusive-or allows her to compute the Li, and consequently, the keys ki, which in turn reveal the F(ki,Si) and the Ri, i.e., the pre-encrypted text Xi. As (only) Alice knows the cryptosecrets for the pre-encryption E(.), she can further compute the plaintext Wi from the Xi.

Blind/hidden search: Assume that Alice wants to find out if word W exists in a given ciphertext C_i. At the same time, she does not want that Bob learns what the word W is,

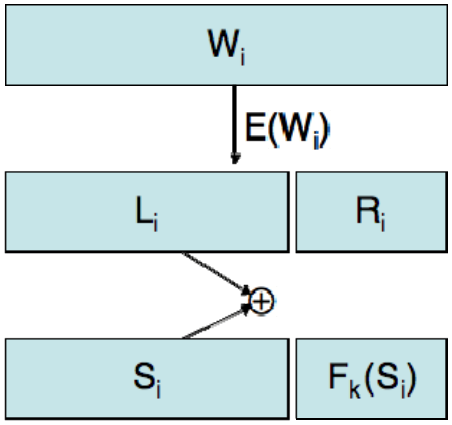

Figure 2: The final encryption scheme for a searchable cipher proposed by Song et. al in [SON00]

Another example of searchable symmetric encryption schemes is proposed by Curtmola et al. in [CUR06]. The proposed algorithm is applicable in the PRISM back end tier in order to store encrypted the files containing the log files allowing only to the allowed people access to the files. The proposed approach, in fact, classifies the entire file containing a specific word and stores this information in a look-up table encrypted with a symmetric algorithm. In the location containing the information for decrypting the i^{th} file containing the searched word is stored the position in the look up table where it is possible to retrieve the information for the $(i+1)^{th}$ file.
A very important theoretical result in the research of symmetric mechanisms that allow the search of words in the encrypted domain has been obtained in [Bel07] by Bellare et. al. In this

paper, in fact, the authors demonstrate that it is possible to build encryption methods that permit fast search in a database by exploiting public-key deterministic encryption schemes. The described approaches are based on the RSA-OAEP and, assuming the plain text is drawn from a space of large min-entropy, it is shown that schemes provide privacy according to the random oracle model.

3.1.2.5 SMC & Regular Expression Matching

In cryptography, secure multi-party computation is a problem that was initially considered by Andrew C. Yao in a 1982 [Yao82]. In secure multi party computation (SMC) protocols, a given number of participants, each owning a set of private data, want to compute a value of a public function using the private data as the input in such a way that the private data does not have to be revealed to the other participants. An SMC protocol is secure if none of the participants learns more from the description of the public function and the result of the global calculation than what he/she can learn from his/her own. One of the most important primitive in order to build an SMC protocol is the oblivious transfer. Example of SMC have been given, e.g., in [Bar05], [Du01], [Du02]. The SMC protocols could be employed in order to allow different PRISM aware entities to compute statistics and to share results without the need to disclose the (sensitive) data possessed by each entity.
Approaches very similar to SMC protocols have also been proposed to allow external entities to carry out a task of matching a regular expression over a given data set without accessing the data directly. Examples of this kind of protocols can be found, e.g., in [Ker06] [Tro07]. The main limitations, both the SMC and the private regular expression matching have, are i) the computational burden and the inefficient use of network resources, and ii) the lack of real implementations.

3.1.3 Hybrid Approach

One-way and two-way techniques cannot only be applied separately on data traffic but also in a joint manner. Such approach aims to combine the strengths of these two types of algorithms or compose them so that a drawback of one or the other algorithm is mitigated. Two semantically different approaches for combination are conceivable: Spatial and Sequential application.
Spatial application of multiple algorithms (e.g. one one-way and one cryptographic) means that the different algorithms are applied independently to different parts of the data packet. For example, packet length information is classified into short/medium/long categories and IP addresses are mapped by applying a cryptographic algorithm which produces a hash value per address.
Sequential application means that multiple algorithms are applied one after the other to the same field or set of fields. For example, IP packet length could be obfuscated by adding a small random number before classifying the length into small/medium/long category in order to hide the exact proportion of short packets in a dataset.
Note that neither the spatial nor the sequential application of multiple algorithms mandate the use of at least one one-way and one two-way algorithm; they just allow combining them in some way. It is also possible to use multiple algorithms of one type in combination and even to combine the spatial and the sequential approach in one anonymisation scheme.
The specific processing steps applied to a data stream depend solely on the targeted application and in which context the resulting trace shall be used. This environment affects the choice, parameters and combination of algorithms applied during the data protection step.
In addition to the previously described approaches, in some situations one could also compute specifically designed metrics or characteristic values for each packet or flow. For example, one could consider some entropy related metrics based on the packet payload, which would later allow (probabilistic) differentiation between encrypted and plain text flows. Another

example is the real-time transport protocol (RTP). RTP packets are encapsulated in UDP packets and there is no explicit way to know if a given UDP packet belongs to an RTP flow or not. However, several computationally light checks can be carried out which a valid RTP header should bypass. Thus again, one can add a single bit of information (metadata) to each packet, which allows a probabilistic identification of RTP flows. This way the actual payload can be excluded from the stored traffic traces, which improves the privacy aspects considerably. (Note that a typical SIP/RTP based VoIP communication is not encrypted, making such traces extremely sensitive).

3.2 Further Research Results

In [KUO06] the authors present AAPI, a tool for anonymisation of traffic traces which supports small set of general primitives:

- ANONYMIZE field anonymisation)
- BPF FILTER (BPF filtering)
- STR SEARCH (string searching)
- COOK (stream reassembly)
- UNCOOK (splitting a stream to its original form)

As it is evident for the presence of COOK and UNCOOK, this framework allows the modification of both the payload and the header of the packets. Supported data modifications include hashing, randomizing, removing, mapping to sequential values and replacing; prefix preserving anonymisation is supported as well.

In [SEE07] a new framework for formulating anonymisation policies in IDS benchmarking data sets is defined; in particular, several fields included by the HTTP protocol are taken in consideration. Such a framework is based on a “filter in” approach and evaluates whether the anonymisation of a given field is required, optional or useless: the potential of each header field for disclosing sensitive data is investigated.

[XU02] provides an important contribution in that it presents a theorem illustrating the canonical form of a prefix preserving anonymisation scheme. Based on such a result, a new address anonymisation scheme, referred to as Cryptopan, is proposed and an evaluation of its security is performed.

[ZAM06] presents a scrambling scheme for packet payload which aims at making the packet content unintelligible while preserving the possibility of revealing small signature strings.

[ZHA06] presents a prefix preserving scheme where two keys are used in a cascaded way. Users with different privilege levels can see different versions of the anonymised trace. An analysis of the potential threats to such anonymisation scheme is included.

[PAN06] presents another trace anonymisation tool named tcpmkpub, which completely removes the packet payload, while storing a consistent amount of trace information as metadata. The implications of a broad set of anonymisation policies for each field of the packet header (including the link layer header) are extensively discussed.

In [SLA06] the authors describe FLAIM, a modular trace anonymisation tool which can process several kinds of logs and that supports the definition of multi-level anonymisation policies which can be expressed through XML files. For each level and field of the log several anonymisation algorithms are provided.

[UBI06] presents a hardware architecture for on-line packet capturing and anonymisation, which has actually implemented over an FPGA chip. The core component of this architecture is a small special purpose processor called Transformation Unit, which supports a simple instruction set for the definition of anonymisation policies; in particular, it provides special instructions for packet processing such as:

- set to a specified constant
- set to a pseudorandom number

- xor with a specified constant
- table-based hashing
- prefix-preserving mapping (based on Cryptopan algorithm)

Each of the above instructions applies to any 16-bit header field in the packet. Further processing, including payload inspection, is implemented in software running on general purpose PCs.

The contribution of [PEU01] is two-fold. A scheme for the compression of traffic traces, taking advantage from the similarities between consecutive packets, is proposed, together with a prefix preserving address anonymisation scheme, which is based on an electronic codebook. This scheme is not prefix preserving in the formal sense, since the topmost byte of the address is left in clear, while the others are mapped in a non prefix-preserving way.

[SLA05] presents the CANINE tool (Converter and ANonymizer for Investigating Netflow Events), which is appositely designed for the anonymisation of Netflow logs. It supports several file formats and provides multiple methods for anonymising the following fields:

- IP address
- Timestamp
- Port Number
- Protocol number
- Byte count

In [SLA04] an extension of Crypto-Pan is proposed, which provides an efficient passphrase-based key generation algorithm. The performance of the extended Crypto-PAn on Cisco NetFlow logs is evaluated as well.

In [PAN03] a trace anonymisation tool is described, which offers the possibility to describe anonymisation policies in an apposite high level language, thus allowing the user to write scripts to express sophisticated trace transformations. It is possible to parse and modify data referring to the application level, such as HTTP and SMTP; in particular, scripts can be defined for editing headers, replacing the content of Web items with MD5 hashes, or altering filenames or reply codes that match given patterns.

[RAM07] presents an enhanced prefix preserving anonymisation scheme which is based on the Canonical form and optimized for online packet processing. Such a scheme achieves a consistent reduction in terms of both memory consumption and processing speed.

In [COU07] the authors present background about inferring sensitive information from anonymised network traces. In the document the authors provide new algorithms for inferring sensitive information from anonymised network traces using state-of-the-art techniques. The work shows that network topology information can be inferred as an artefact of usable network packet traces, and that behaviours of hosts are an important piece of identifying information that which can be leveraged to subvert the anonymisation process. This especially highlights the need to obfuscate behavioural and network topology information which is not a trivial task. This fosters the need to make researchers aware of the compromise they need to find between usability of anonymised data and privacy concerns.

In [FAB06] “Anonymisation of Measurement and Monitoring Data: Requirements and Solutions” the authors we provide a study of the current legal situation in Germany. The show that there are many flaws in the German law concerning the appropriateness of particular monitoring solutions. They also discuss possible approaches for anonymisation of measurement and monitoring data and evaluate their applicability in respect of the applicable laws.

3.3 A Survey on Anonymisation Tools

The following table shows an overview of common (free) tools used in the research community for performing traffic trace anonymisation. In general one can state that tools for this purpose have become quite versatile and mature. However most of them still lack professional documentation and support and are provided in a as-is manner. From the state of the art with respect to anonymisation tools one can see that at first many highly specialised single-purpose tools were developed and later the a trend towards frameworks and policy languages for the specification of the concrete anonymisation emerged. Creation and validation of such policies is up to the user.

These are the characteristics we evaluated during the analysis of the tools:

Prefix Preserving – If a tool supports prefix preserving anonymisation for IP addresses this means that it is capable of separately anonymising the network part and the host part of an IP address, thus preserving vital traffic characteristics. This is supported by almost any tool. However as there are different approaches to doing prefix preserving anonymisation one should check for a used tool which algorithm exactly is used.

Stateless Processing (consistent anonymisation) – The ability to perform stateless anonymisation denotes the following: the result of anonymising a packet X does not depend on any specifics of packets seen before. This characteristic allows in effect performing the anonymisation process in parallel in a distributed manner and afterwards getting a consistent result when concatenating the anonymised traces again. This is not a feature which most anonymisation tools provide but it is becoming more and more available in the more recent tools as it is seen as am important feature.

Vulnerabilities – Depending on the concrete anonymisation algorithm and the configuration parameters in use the resulting output of anonymisation can be vulnerable to reverse engineering attacks. Some of these attacks are generic, other can be tool-specific. Some links to those are listed in the table.

Input – pcap is the de facto standard for input files to anonymisation tools. Some tools support flow reports (netflow v5, v7, or v9) instead of packet traces as input. Rarely some proprietary input format is observed.

Online Processing – Most pcap based tools can also do live capture from an interface, but for some configurations of these tools there are restrictions to live capture in case that the used algorithm requires multiple passes over the packet data.

Output – The output format is usually equal to the input format (pcap or netflow). Sometimes additional statistics are written as ASCII text output to the console too. Output is per default written into a file, no matter if the input was life traffic or a file too.

Reverse Mapping – This denotes the ability of a tool to explicitly output the mapping which was used for anonymisation to a data file. Such mapping needs to be kept strictly secret but it is useful to revert the processing in case that e.g. an attacker needs to be identified.

Table 1: Overview of anonymisation tools.

Tool	Prefix preser-ving	Stateless processi ng	Applied consistently	Vulnerabilities	Input	Online processing	Output	Reverse mapping	Remarks
tcpdpriv	Yes	No	No, except merging pcap files	http://ita.ee.lbl.gov/html/contrib/attack50/attack50.html	Pcap	Yes	pcap	No	
ipsumdump	Yes	No	No, except merging pcap files	Tcpdpriv derived	Pcap, ipsumdump	Yes	Ascii, ipsumdump, pcap	no	Uses click for the actual work. Anonymize IPAddr is based on tcpdpriv
tcpurify	Yes, possible	No	No, except merging pcap files		Pcap	Yes	Pcap	yes	
tcpmkpub	Yes		yes	depends on policy	Pcap	Yes, but might lead to collisions	Pcap	yes	is a framework
anontool	Yes	Yes	Yes		netflow5, netflow9 in pcap	Yes	Pcap		implements an Anonymization API
Crypto-PAn	Yes	Yes	Yes			should be possible via named pipes		yes	works only on IP addresses; is a lib with a sample application

Canine	Yes	No	Yes		Netflow v5, v7,nfdump, CiscoNCSA, ArgusNCSA,	No		Should be possible	not maintained anymore, successor is FLAIM
scrub netflows	Yes, modified CryptoPAN		Yes, Keyed randomization		Netflow	maybe, but not built for	Netflow		
flowmon probe	No	No	No		Wire	Thats its only intention			Hardware device
ruler	Yes	Yes	Yes	depends on policy	Pcap		Pcap		Its a rewriting language
FLAIM	Yes				pcap headers, netfilter, NetFlows	Yes			
Tcpdump Anonymizer	Yes	No	No		Pcap, wire	Yes		No	Undocumented (no documentation found)

4 Complexity and Performance

With regard to computational complexity the anonymisation techniques presented spread across a wide range of numbers. This is mainly due to the pure difference in mathematically required computations for each algorithm.
We can differentiate the effort needed in two dimensions: (a) stateless (see section for a definition 3.3) versus stateful processing and (b) cryptographic versus non-cryptographic.
With regard to (a) we can note that stateless processing (e.g. replacement of an IP address by applying a fixed hash function) generally has an effort of O(1), because it only takes into account the data from one packet at a time and some function or mapping table. Consider in contrast stateful IP address remapping where each newly found IP address will be assigned a new anonymised IP address. This requires a lookup of each IP address per packet to check if this address has already been observed, thus requiring a worst-case performance of O(log2(n)) where n is the number of addresses already seen. However this does not mean that stateful methods are always faster.
With regard to (b) one can state that cryptographic functions which can be used for hashing purposes (such as those used for generating secure fingerprints for cryptographic signatures) are generally much slower than non-cryptographic functions.
A very broad overview and comparison for the computation of normal as well as cryptographic hash functions based on IP packet data can be found in [HEN07].

5 Conclusions

This deliverable has documented the State of the Art on data protection techniques applicable for the PRISM project and for the to-be-developed PRISM system. The information herein covers a broad scope from techniques such as trusted computing which governs hardware access, via security in databases, up to the classical datagram field anonymisation techniques and cryptographic schemes.

The information collected has shown us that in order to build a system which is privacy-preserving as well as secure (and still convenient to use) one needs to have a clear goal in mind and must peek at the details of the targeted solution in order to select a set of appropriate and matching techniques for the realisation.

The future PRISM system will build on the knowledge gained in this deliverable. For the system we can select techniques which are able to balance the performance bottlenecks between different system components whilst maintaining data protection by using the matching cryptographic and/or anonymisation techniques.

We have seen that a certain level of data protection will be available at the network processor card, directly applied after packet capture. More advanced techniques can be applied at later stages of the PRISM font-end.

For connecting the PRISM font-end with the back-end we can choose from several (mostly cryptographic) techniques which allow a safe data export using the IPFIX protocol.

In the back-end system, we plan to make use of traditional methods (e.g. for database management security) in order to implement secure access to the PRISM database plus some advanced schemes to make the protected data searchable.

The data protection of the packet data itself can be performed by a suited combination of the anonymisation techniques documented in this deliverable. For each scenario we will need to select an appropriate selection of packet fields, techniques, and analysis tools in order to maximize the data protection while retaining the data usefulness for the intended purpose.

In addition to protecting the data itself the PRISM system will then govern the access to (and in some scenarios the use of) the protected data, including a safe user authentication plus provisioning of selected and adapted analysis tools.

This document will help us with the information about techniques, tools, and the knowledge of the risks of releasing network data, to select the right algorithms and components for the future PRISM system.

References

[AGR02] R. Agrawal, J. Kienrnan, R. Srikant and Y. Xu, "Hippocratic Databases", in Proc. of the 28th International Conference on Very Large Databases (VLDB 2002), Hong Kong, China, Aug. 2002.

[AGR04] AGRAWAL, R., KIERNAN, J., SRIKANT, R., AND XU, Y. 2004. Order-preserving encryption for numeric data. In Proceedings of the ACMSIGMOD 2004 Conference, Paris, France. ACM Press, New York.

[AMD] A. M. Devices. AMD Secure Virtual Machine Architecture Reference Manual. AMD, 2005

[ASH03] P. Ashley, S. Hada, G. Karjoth, C. Powers, M. Schunter, "The Enterprise Privacy Authorization Language (EPAL), EPAL 1.2 Specification", IBM Research Report, 2003, http://www.zurich.ibm.com/security/enterprise-privacy/epal/.

[BOU02] BOUGANIM, L. AND PUCHERAL, P. 2002. Chip-secured data access: Confidential data on untrusted servers. In Proceedings of the 28th International Conference on Very Large Data Bases, Hong Kong, China. Morgan Kaufmann, San Mateo, CA, 131–142

[BOY03] BOYENS, C. AND GUNTER, O. 2003. Using online services in untrusted environments—a privacypreserving architecture. In Proceedings of the 11th European Conference on Information Systems (ECIS '03), Naples, Italy

[BYU04] J. Byun, E. Bertino and N. Li, "Purpose Based Access Control for Privacy Protection in Relational Database Systems", CERIAS Technical Report 2004-52, Purdue University, 2004.

[BYU05] J. Byun, E. Bertino and N. Li, "Purpose Based Access Control for Privacy Protection in Relational Database Systems", in Proc. of the 10th ACM symposium on Access control models and technologies, Stockholm, Sweden, June 2005.

[COU07] S Coull, C Wright, F Monrose, M Collins, M Reiter, "Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces", in proceedings of the Network and Distributed System Security Symposium (2007)

[CUR06] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky, *"Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions"*

[DAM04] DAMIANI, E., DE CAPITANI DI VIMERCATI, S., PARABOSCHI, S., AND SAMARATI, P. 2004. Computing range queries on obfuscated data. In Proceedings of the Information Processing and Management of Uncertainty in Knowledge-Based Systems, Perugia, Italy

[DAV81] DAVIDA, G.,WELLS, D., AND KAM, J. 1981. A database encryption system with subkeys. ACMTrans. Database Syst. 6, 2 (June), 312–328

[DISCREET] FP6 IST DISCREET (Discreet Service Provision in Smart Environments), home page: http://www.ist-discreet.org.

[DOM96] DOMINGO-FERRER, J. 1996. A new privacy homomorphism and applications. Inf. Process. Lett. 60, 5 (Dec.), 277–282.

[DOM98] DOMINGO-FERRER, J. AND HERRERA-JOANCONMART´ı, J. 1998. A privacy homomorphism allowing field operations on encrypted data. Jornades de Matem` atica Discreta i Algor´ısmica.

[FAB06] Fabian Haibl and Falko Dressler, Anonymisation of Measurement and Monitoring Data: Requirements and Solutions, in "Praxis der Informationsverarbeitung und Kommunikation" collection #29, issue #4, pages 208–213, ISSN 0930-5157, october-december 2006

[FER01] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn and R. Chandramouli, "Proposed NIST Standard for Role-Based Access Control", ACM Transactions on Information and System Security (TISSEC), Vol. 4, No. 3, pp. 224-274, August 2001.

[GOL04] O. Goldreich, "Foundations of Cryptography: a two-volume textbook", Volume II, Cambridge University Press 2004.

[HAC02] HACIG ¨UM¨US, H., IYER, B., LI, C., AND MEHROTRA, S. 2002a, "Executing SQL over encrypted data in the database-service-provider model", in proceedings of the ACM SIGMOD'2002, Madison, WI. ACM Press, New York

[HAC04] HACIG ¨UM¨US, H., IYER, B., AND MEHROTRA, S. 2004. Efficient execution of aggregation queries over encrypted relational databases, in proceedings of the 9th International Conference on Database Systems for Advanced Applications. Springer, Jeju Island, Korea

[HEN07] Henke, C., Schmoll, C., Zseby, T.: Empirical evaluation of hash functions for multipoint measurements. Technical Report TR-2007-11-01 (Available upon request)

[INTEL] D. Grawrock. The Intel Safer Computing Initiative Building Blocks for Trusted Computing. Intel Press, http://www.intel.com/intelpress/sum secc.htm , 2005

[JEN00] JENSEN, C. 2000. "Cryptocache: a secure sharable file cache for roaming users". In Proceedings of the 9th Workshop on ACM SIGOPS European Workshop: Beyond the PC. New Challenges for the Operating System, Kolding, Denmark. 73–78

[KAR02a] G. Karjoth and M. Schunter, "A Privacy Policy Model for Enterprises", in Proc. of 15th IEEE Computer Foundations Workshop (CSFW '02), June 2002.

[KAR02b] G. Karjoth, M. Schunter and M. Waidner, "Platform for Enterprise Privacy Practices: Privacy-enabled Management of Customer Data", in Proc. 2nd Workshop on Privacy Enhancing Technologies, Lecture Notes in Computer Science, Vol. 2482, Springer-Verlag, 2002.

[KAR02c] G. Karjoth, M. Schunter and M. Waidner, "Privacy-enabled Services for Enterprises", in Proc. International Workshop on Trust and Privacy in Digital Business (TrustBus 2002), Aix en Provence, France, Sep. 2002.

[KLE89] KLEIN, S., BOOKSTEIN, A., AND DEERWESTER, S. 1989. Storing text retrieval systems on CD-ROM: compression and encryption considerations. ACM Trans. Inf. Syst. 7, 3 (July), 230–245

[KUO06] D. Koukis, S. Antonatos, D. Antoniades, E.P. Markatos, P. Trimintzios, "A Generic Anonymisation Framework for Network Traffic"

[LEF04] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu and D. J. DeWitt, "Limiting Disclosure in Hippocratic Databases", in Proc. of the 30th International Conference on Very Large Databases (VLDB 2004), Toronto, Canada, Aug. 2004.

[MAS05] F. Massacci, J. Mylopoulos and N. Zannone, "Minimal Disclosure in Hierarchical Hippocratic Databases with Delegation", in Proc. of 10th European Symposium on Research in Computer Security (ESORICS 2005), Milan, Italy, Sep, 2005.

[MAS06] F. Massacci, J. Mylopoulos and N. Zannone, "Hierarchical hippocratic databases with minimal disclosure for virtual organizations", The International Journal on Very Large Data Bases, Vol. 15, No. 4, pp. 370-387, Springer, Nov. 2006.

[MIL03] Ethan L. Miller Geoff Kuenning, "Anonymisation Techniques for URLs and Filenames"

[MIN97] G Minshall, "TCPdpriv: Program for Eliminating Confidential Information from Traces"

[MON05a] M, Casassa Mont, R. Thyne and P. Bramhall, "Privacy Enforcement with HP Select Access for Regulatory Compliance", Hewlett-Packard Labs Technical Report, HPL-2005-10, 2005.

[MON05b] M, Casassa Mont, "A System to Handle Privacy Obligations in Enterprises", Hewlett-Packard Labs Technical Report, HPL-2005-180, 2005.

[MON06] M, Casassa Mont and R. Thyne, "A Systemic Approach to Automate Privacy Policy Enforcement in Enterprises", in Proc. of 6th Workshop on Privacy Enhancing Technologies, Lecture Notes in Computer Science, Vol. 4258, Springer-Verlag, 2006.

[MOS05] T. Moses (ed), "OASIS Privacy Policy Profile of XACML v2.0", OASIS Standard, Feb. 2005.

[OASIS] Organization for the Advancement of Structured Information Standards (OASIS), "OASIS eXtensible Access Control Markup Language (XACML) TC", 2004, http://www.oasis-open.org/committees/xacml/.

[OPENTC] 6th FWP IST OPEN_TC (Open Trusted Computing), home page: http://www.opentc.net.

[PAN03] Ruoming Pang, Vern Paxson, "A High-level Programming Environment for Packet Trace Anonymisation and Transformation"

[PAN06] Ruoming Pang, Mark Allmanz, Vern Paxsonz, Jason Lee, "The Devil and Packet Trace Anonymisation"

[PEU01] Markus Peuhkuri, "A Method to Compress and Anonymize Packet Traces"

[PRIME] FP6 IST PRIME (Privacy and Identity Management for Europe), Home page: https://www.prime-project.eu/.

[RAM07] Ramaswamy Ramaswamy, and Tilman Wolf, "High-speed prefix-preserving IP address anonymisation for passive measurement systems"

[RIV78] RIVEST, R., ADLEMAN, L., AND DERTOUZOS, M. 1978. Data banks and privacy homomorphisms. In Foundations of Secure Computation. Academic Press, Orlando, FL. 169–179

[SCH02] M. Schunter and P. Ashley, "The Platform for Enterprise Privacy Practices", in Proc. Information Security Solutions Europe (ISSE 2002), Paris, France, Oct. 2002.

[SEE07] Vidar Evenrud Seeberg, Slobodan Petrovic, "A New Classification Scheme for Anonymisation of Real Data Used in IDS Benchmarking"

[SLA04] Adam Slagell, Jun Wang, William Yurcik, "Network Log Anonymisation: Application of Crypto-PAn to Cisco Netflows"

[SLA05] Adam J Slagell, Yifan Li and Katherine Luo, "Sharing Network Logs for Computer Forensics: A New Tool for the Anonymisation of NetFlow Records"

[SLA06] Adam Slagell, Kiran Lakkaraju, and Katherine Luo, "FLAIM: A Multi-level Anonymisation Framework for Computer and Network Logs"

[SON00] Song, D., Wagner, D., and Perrig, A., "Practical techniques for searches on encrypted data. In Proceedings of the 2000 IEEE Symposium on Security and Privacy", Oakland, CA. IEEE Computer Society Press, Los Alamitos, CA. 44–55

[TAPAS] 5th FWP TAPAS (Trusted and QoS-Aware Provision of Application Services), home page: http://www.newcastle.research.ec.org/tapas/

[TCG] Trusted Computing Group (TCG), https://www.trustedcomputinggroup.org/specs/

[UBI06] Sven Ubik, Petr Žejdl, Jiří Halák, "FPGA-based Packet Header Anonymisation"

[XU02] Jun Xu Jinliang Fan Mostafa H. Ammar Sue B. Moon, "Prefix-Preserving IP Address Anonymisation: Measurement-based Security Evaluation and a New Cryptography-based Scheme"

[YAV00] R. Yavatkar, D. Pendarakis and R. Guerin, "A Framework for Policy-based Admission Control", IETF RFC 2753, January 2000.

[ZAM06] José Zamora Ponce, Martin Loebl, Lukas Kencl, "Packet Content Anonymisation by Hiding Words"

[ZHA06] Qianli Zhang and Xing Li, "An IP Address Anonymisation Scheme with Multiple Access Levels"

Appendix A – Glossary

DATA CATEGORIES

Data in the PRISM context is classified into five different categories:

PACKET DATA

- any field from layer 2,3,4 data or payload of packets seen on the wire e.g. EtherType, IP addresses, TCP sequence number, Total Length, etc.
- for such fields the defined names from wireshark Display Filters are useful, e.g. eth.type, ip.src, ip.dst, tcp.seq, ip.len, http.last_modified etc. See also in the wireshark Display Filter Reference

MEASUREMENT DATA

- based on the results of the measurement but not on bits from inside the packet data
- timestamps, link load (if taken from the device, e.g. from the Router MIB via SNMP), CPU load

TASK META-DATA

- any 'setting' used for performing the measurement and information about the measurement setup
- e.g. link capacity, meas. task initiator, router type, start time of task, monitored AS, filter expression
- may include also the anonymisation algorithms and parameters used (but it might be not allowed to disclose all of them!)

DERIVED DATA

- based on statistical analysis of the measurement data and/or packet data ; e.g. flow bandwidth, mean packet size, inter-packet delay distribution, TCP connects per second...

EXTERNAL DATA

- required by monitoring applications to perform analysis and evaluation of the measurement data and packet data ; e.g. IP address -> AS number mapping, IDS signatures

ANONYMISATION OPTIONS:

We can differentiate two basic functional approaches to changing information:

ONE-WAY FUNCTION:

- The term one-way function refers to a mapping which is hard or impossible to reverse.
- Typical usages include hashes (integrity, authentication) and anonymisation.

TWO-WAY FUNCTION (CIPHER):

- The term two-way cipher is a reversible function f from plaintext M to cipher text C parameterised with one or multiple keys.

- A strong requirement for ciphers is that it is computationally infeasible (apart from brute-force attack) to determine a plaintext message M from cipher text C=f(M) without the knowledge of the relevant key material.

DATA FORMATS

Data can appear in ***plaintext***, ***encrypted*** and/or ***anonymised***. In principle, encryption and anonymisation can be applied to all of the above data categories, while not all possible combinations are useful in practice.

PLAINTEXT

- Data in ASCII or binary form.

ENCRYPTED

- data encrypted by a one-way function or a cipher

ANONYMIZED

- data where private information about users is reduced
- during the anonymisation process, private information can either be altered or deleted
- the data format is expected to be semantically the same as the original data used as input (e.g. for packet data the tcpdump format can be kept during the anonymisation process)
- anonymised data does not include the description of the applied anonymisation (would be task meta-data)