SOFTWARE—PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2010; **00**:1–18
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spe

A Pattern Language for Developing Privacy Enhancing Technologies

Munawar Hafiz

University of Illinois at Urbana-Champaign, 201 N Goodwin Ave, Urbana, IL 61801, USA

SUMMARY

A Privacy Enhancement Technology (PET) is an application or a mechanism which allows users to protect the privacy of their personally identifiable information. Early PETs were about enabling anonymous mailing and anonymous browsing, but lately there have been active research and development efforts in many other problem domains. This paper describes the first pattern language for developing privacy enhancement technologies. Currently, it contains 12 patterns. These privacy patterns are not limited to a specific problem domain; they can be applied to design anonymity systems for various types of online communication, online data sharing, location monitoring, voting and electronic cash management. The pattern language guides a developer when he or she is designing a PET for an existing problem, or innovating a solution for a new problem.

KEY WORDS: Patterns; Pattern Language; Privacy; PET.

1. INTRODUCTION

Privacy is the right of an individual to be left alone [68]. It determines how much information about a person is exposed to others [69]. Privacy is most highly valued by 'public figures'; but it is equally coveted by 'commoners' because they do not want to turn their activities into a public spectacle. The central concept of privacy is anonymity, which refers to the state of an individual's information being publicly unknown. In reality, people do not intend to be entirely anonymous. Rather privacy is considered to be the choice that a person possesses to disclose or hide his or her activities.

Privacy Enhancing Technologies (PETs) enforce this choice by preventing unnecessary or undesired processing of personal data [65, 64]. Early efforts on PET focused on emailing and browsing. These systems allowed a user to anonymously communicate with other users without revealing identity. The design decisions made in these systems follow recurring patterns.

In recent years, PET developers have explored many new and exciting domains in which privacy has become relevant, e.g., privacy-preserved data sharing, location anonymity, anonymity in Radio Frequency IDentification (RFID), etc. The challenges faced by these PET developers appear different; yet a detailed analysis unravels many similarities with the challenges faced by the developers of classic PETs for anonymous emailing and browsing. Patterns can be useful to document these best practices. However, there has been very little work in documenting privacy patterns. The published patterns [58, 59, 53] are about general privacy concepts, and provide insufficient guideline for PET development. Other patterns document how to convey privacy policies to end users during online interactions [54, 53], but they are not directly about developing PETs.

This paper describes a pattern language containing 12 patterns for developing PETs. These patterns are applicable to the design of anonymity solutions for various domains: anonymous messaging, anonymous browsing, zero-knowledge communication, privacy-preserved data sharing, anonymous voting, anonymous digital cash, location anonymity, and anonymity in emerging technologies such as vehicular ad hoc networks (VANET) and Radio Frequency IDentification (RFID). Although each domain has its own PETs, there are design decisions that can be generalized. Understanding these design choices and how they are used to solve a problem in a context benefits PET researchers to develop solutions for new privacy challenges.

Prepared using speauth.cls *[Version: 2010/05/13 v3.00]*

This paper makes three contributions. It contains the *first* set of patterns that address PET design issues. The patterns have been derived from an in depth survey of existing privacy literature and understanding the design decisions that work; they are all *new* patterns that document classic, reusable privacy solutions. Second, it is an early paper that talks about pulling individual patterns from a larger pool to make a language for a specific problem domain. We describe a pattern language that can act as a guide for developers. Third, our pattern language allows practitioners to explore the big picture of privacy problems and solutions. The patterns are *architectural*: they narrate a high level solution and how it can be applied in different PET contexts. Since there has been a lot of interest in designing new and improved PETs, our pattern language eases design efforts by prohibiting developers from 're-inventing the wheel'.

We first survey the domain of privacy problems to identify the scope of our patterns. We describe the threat model in section 3. Then, we describe our pattern language and narrate the patterns in section 4. Finally, we demonstrate some case studies in which these patterns can be used to solve emerging problems in section 5. Section 6 lists other related patterns. Section 7 concludes.

2. SCOPE OF OUR PRIVACY PATTERNS

EPIC distinguishes four models of privacy protections [17]: 1) Having comprehensive laws that govern collection and use of data in private and public sector, 2) Having special purpose laws in specific domains, e.g., financial domain, 3) Companies and services practising self-regulation and comply to established standards, and 4) Building PETs to protect privacy and security.

Privacy, therefore, is not merely a non-functional quality of software—it is the right of every individual. State and legislative institutions have the duty to uphold these rights. There are laws such as the Health Insurance Portability and Accountability Act of 1996 (HIPAA), the Children's Online Privacy Protection Act of 1998 (COPPA), and the Family Educational Rights and Privacy Act of 1974 (FERPA) to protect the way information about an individual is used.

Privacy in software has a lot to do with compliance with standards established by governments and other bodies. These standards aim to bridge the trust gap between users and services. The Platform for Privacy Preferences (P3P) [10] project provides a standard, machine-readable format for online privacy policies and a protocol that enables web browsers to read and process privacy policies automatically. The P3P project enables the development of tools for making informed decisions about when and how to reveal personal information. The tools do not provide anonymity; they only allow a user to understand and negotiate the anonymity service that is provided.

The patterns in this paper are *not* about models to enforce privacy and bridge trust gaps, nor are they about P3P-based tools that allow users to take informed decisions. The patterns in this paper document design decisions *that enable anonymity* in PETs.

Suppose, Alice wants to access some sensitive content online. She does not want anybody to find out about this activity. The service hosting the sensitive content may be P3P-enabled so that Alice's browser is aware that her information will not be mishandled. However, Alice uses the Tor [14] anonymity service to privately access the sensitive content. Mallory, who wants to monitor Alice's activity, is not able to break the anonymity provided by Tor, and therefore fails. The patterns in this paper are about implementing Tor or other similar anonymity services; they describe design decisions that allow an anonymity service to shield a user's privacy.

But what fraction of privacy domain is covered by these patterns? What classes of privacy problems are not covered? Section 2.1 analyzes the problem domain to answer these questions. Section 2.2 describes the applications domains that our patterns concentrate on.

2.1. *Privacy Problems: The Big Picture*

We applied a classification scheme to understand the privacy problem domain. Antón at al. [3] analyzed the privacy requirements of web applications. They identified taxonomy categories that can act as a checklist during web application design. Our classification scheme is based on a more

DOI: 10.1002/spe

general scheme because, as suggested by EPIC's privacy protection models [17], privacy problems appear in many domains other than web applications.

We identify *what* is being protected by PETs, and *who* is it being protected from. PETs aim to protect user *data* as well as user communication *pattern*s that may passively reveal data. The attackers can be classified in 3 groups based on their relationship with an individual: 1) an *insider*, e.g., an application that a user trusts, 2) a *partner*, e.g., an application that a user does not trust but has to depend on for a transaction, and 3) an *outsider*, i.e., an entity that a user does not trust at all.

Table I. Taxonomy of Privacy Issues

Protect from whom? / What to protect?	Insider	Partner	Outsider
Data	Access Control, Privacy legislation	Privacy Preserving Data Sharing (Oblivious Transfer)	Protect Confidentiality
Pattern	Access Control, Privacy legislation	Privacy Preserving Data Sharing	PETs for hiding communication trends

Table I describes the privacy issues. Hiding data and patterns against insiders is the concern of access control mechanisms. From privacy perspective, there are various laws that provide guideline of how to protect information from insider attack. Hiding data and patterns from partners is about enabling users to be involved in a zero-knowledge communication. Hiding data from outsiders is strictly about making data confidential, and is covered by classic security research; it is outside the scope of privacy. On the other hand, the main goal of privacy research is about hiding communication trends from outsiders.

The patterns in this paper concentrate on two anonymity requirements: to hide details about user communication trends from outsiders, and to enable users to involve in an oblivious communication, in which the agents do not receive any additional information about each other. The recurring privacy mechanism to hide communication patterns is a mix based system, which hides a user's messages with messages coming from other users. The concept of a mix based system has been adopted in various application contexts. Most of the patterns in this paper describe various design decisions to implement a mix based system. On the other hand, oblivious communication depends on cryptographic methods; it allows a user to communicate with a server without revealing any details to the server. Section 5 describes how these two requirements overlap in emerging privacy domains, and how a combination of the patterns can be applied.

There are many gray areas that are yet to be explored, especially about hiding data and patterns from insiders. At the same time, issues such as privacy-preserving data sharing will perhaps involve more patterns as more PETs are being implemented to enable zero-knowledge communication.

2.2. *Problem Domains and Solution Strategies*

We started with Goldberg and colleague's [28, 27] survey on PETs; then, we analyzed research papers describing PETs for other problem domains. The central privacy requirement is anonymity, but each domain has its own interpretation of anonymity and own strategies for achieving it. The problem domains are,

- *Anonymous online communication.* The most common privacy requirement is hiding user activity in online communication. Users typically adopt cryptographic techniques to protect the content, e.g., encrypting data packets, adding hash values to prevent tampering, etc. However, a lot of information can be harvested about a message sender (and maybe a message) by monitoring the sender's messaging pattern. This is the typical threat that motivates anonymous emailing technologies. Anonymous remailers mix a user's data packets with packets from other users and modify the packets so that the incoming packets cannot be correlated with the outgoing ones. They hide sender identity, and/or receiver identity from an active/passive adversary.

Prepared using speauth.cls
DOI: 10.1002/spe

Anonymous web browsers follow mixing strategy, but they also have to provide low-latency service. The low-latency constraint requires different mixing choices. Similar low-latency requirements exist for other services such as interactive chats and remote login sessions.

- *Privacy-preserving data sharing.* A major problem domain in recent years is sharing statistically significant data without revealing individual information. Data sharing systems modify data to hide individual information without tampering the statistics of data. Similar technologies are applicable for log anonymization.
- *Zero-knowledge operations.* Private information retrieval mechanisms allow a user to query a database, and get precise result, without revealing any information about the query to the database server. Another similar zero-knowledge operation is a privacy-preserving set operation, where data owners can involve in a set operation without revealing their individual sets. These cryptographic techniques are the technical backbone of anonymous digital cash.
- *Anonymous voting.* Anonymous electronic voting systems and protocols have been in active development in recent years. The problem in this case is to hide the correlation between a voter and a vote, yet to allow a voter to verify that the vote has been counted. At the same time, the administrators need to be assured that the tallying has been done correctly.
- *Anonymity in location-based services and VANET.* Advances in mobile networks have allowed a lot of location-based services to emerge in recent years. The anonymity requirement in this domain is to hide location information of user agents. Also, the agents should be able to involve in a transaction without compromising their location information. A vehicular ad hoc network (VANET) [19] is a type of location-based network devoted to vehicles and roadside units. In this context, motor vehicle records and personal information are to be protected. However, absolute privacy is not a target of VANET. An active adversary can abuse the absolute anonymity and flood VANET with wrong information. Rather, VANET users need to trust the system so that roadside safety units can monitor and access the information, but attackers can not; nor can they flood the system with garbage information.
- *RFID Privacy.* RFID (Radio-Frequency IDentification), which is used for tracking and tracing, is vulnerable to being monitored by attackers. This happens without the knowledge of the possessor of an RFID label. Implementing privacy in this domain is a different challenge than online privacy. Some RFID issues such as tracking and clandestine inventorying are related to privacy protection, and are covered by privacy patterns. Other issues such as RFID authentication are security problems, with their own security patterns.

The target of anonymity is also different. In online communication, data packets should be anonymized. In data sharing, the target is the information stored about individuals in databases. In location-based services or VANETs, the targets are humans themselves or the cars they are driving.

One major application domain is not covered by patterns in this paper. Anonymous publishing services hide a publisher's identity. However, the main requirement for these services is to uphold the freedom of speech by preventing censorship. These are essentially distributed storage management systems and have their own patterns. A widely-known censorship-resistant storage system is Freenet [8]; the most famous Freenet-based service is WikiLeaks.

3. THREAT MODEL

PETs are designed with a desired degree of anonymity requirement and an attacker model. The patterns in this paper refer to these metrics to illustrate the privacy guarantee they provide.

3.1. Degree of Anonymity

The degree of anonymity required is domain specific, as well as user-specific and context-specific. Conversely, the anonymity provided by a PET may vary based on various constraints.

Reiter and Rubin [49] informally describe anonymity as a six point continuum: absolute privacy, beyond suspicion, probable innocence, possible innocence, exposed, and provably exposed. For

DOI: 10.1002/spe

most PETs, *absolute privacy* is not a realistic goal; rather, PETs require that an attacker can not distinguish an anonymity target from other probable suspects (*probable innocence*). On the other hand, PETs should not *expose* the anonymity targets.

For anonymous communication, Pfitzman and Waidner [47] describe three types of anonymity properties: sender anonymity, receiver anonymity, and sender and receiver unlinkability. *Sender anonymity* means that the identity of a communication initiator is hidden. Encryption hides data content, but not the identity of a communication initiator. A plaintext message that does not have any trace, that can be used to link it back to the sender, does not provide data confidentiality, but provides sender anonymity. *Receiver anonymity* means that the identity of the receiver of a message remains hidden. *Sender and receiver unlinkability* means that though the sender and receiver can be identified as participating in some sort of communication, they cannot be correlated to participate in a conversation (i.e., communicating with each other).

3.2. Attacker Model

Table I lists three classes of attackers. This is based on the trust that a user shares with another entity or an application. The privacy patterns in this paper are about implementing PETs against untrusted (*outsider*) and semi-trusted (*partner*) adversaries.

Pfitzman and Waidner [47] classify the attackers for online communication. A *local eavesdropper* can observe all (and only) communication to and from a user's computer. A more powerful eavesdropper can monitor all data traffic in a local network. A hypothetical *global eavesdropper* is omniscient of all the activities in a network. This classification is based on online communication, but the idea is applicable to adversaries in mix based networks of other domains.

Adversaries can also be classified as active, semi-honest and passive. An *active* adversary manipulates packets flowing through it. A *passive* adversary only monitors. A *semi-honest* adversary follows the network protocol and appears to be honest, but it manipulates the through traffic. Adversaries can work on their own, or they may be colluding.

PETs in different domains have different privacy requirements. For example, low-latency peer-to-peer anonymous communication systems are not designed to resist a global passive adversary. They resist against a set of colluding active and passive adversaries that control a fraction of all the nodes in a network. Hence powerful adversaries, such as governments, can seriously undermine privacy by deploying enough nodes to gain a significant fraction of a peer-to-peer network.

4. A PATTERN LANGUAGE FOR DEVELOPING PETS

We have been maintaining a comprehensive catalog of all published security patterns [31], published by various authors over the last 15 years in various books, catalogs and papers. Currently the catalog contains 96 patterns. The 12 privacy patterns in this paper are from that larger catalog.

The security pattern catalog revealed to us unexplored problem domains. Despite the work done on PETs, the lack of privacy patterns was very obvious. We started with Goldberg and colleague's [28, 27] survey papers and read privacy papers appearing in well-known security conferences. The survey papers are a well-known and widely-accepted source of important problem domains. We augmented it with recent work on privacy in emerging problem domains. For each problem domain, we identified the well-known PETs. Then we analyzed the design decisions of these PETs to find the solution strategies that recur. This led to the 12 privacy patterns listed in this paper. None of the patterns appear in any other pattern catalog.

We have created a pattern language that illustrates the relationship between 96 security patterns from our catalog. Figure 1 shows the part of the larger pattern language that contains the 12 privacy patterns. The patterns are rooted at a more general security pattern INFORMATION OBSCURITY [58]. The security pattern is concerned about protecting information from unauthorized access. The privacy patterns obscure information in two ways: by providing anonymity in a communication scheme and by enabling a zero-knowledge secure multiparty transaction.

DOI: 10.1002/spe

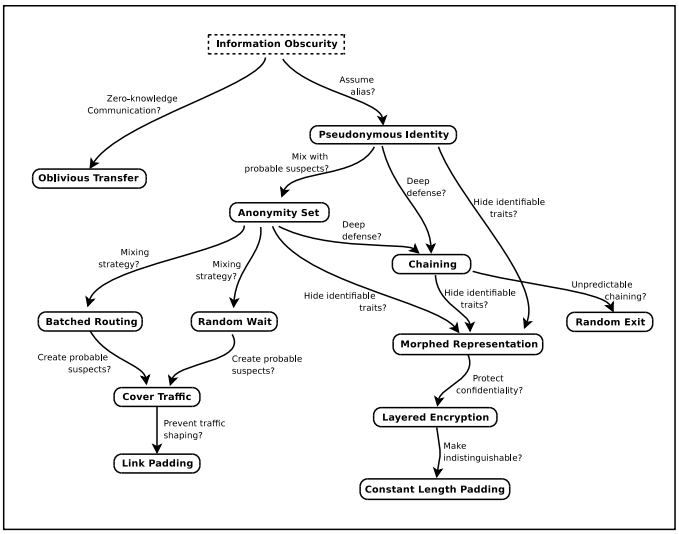

Figure 1. A Pattern Language for Privacy Patterns

PSEUDONYMOUS IDENTITY hides anonymity targets behind a pseudonym. The pseudonymous communication is then blended with other communication and an ANONYMITY SET is created. The defense in depth principle [66] suggests CHAINING anonymity solutions, so that failure of a layer of anonymity does not expose an anonymity target.

The rest of the patterns down the chain are related with a mix based network, which is a core anonymity technology. A mix based network applies MORPHED REPRESENTATION to hide the relation between incoming and outgoing data. In online communication, mix networks adopt LAYERED ENCRYPTION to hide packet content. CONSTANT LENGTH PADDING is used in conjunction with encryption to make all the encrypted packets of the same size.

In a mix based scheme, the anonymity set is created by mixing traffic with other probable suspects. A BATCHED ROUTING scheme stores incoming packets in a pool and only sends them out when the pool reaches a threshold of packets from various senders. A low-latency network may not tolerate this delay; instead it may adopt a RANDOM WAIT strategy, in which a packet sender associates a random delay to a packet. At a mix network, a packet is forwarded when a threshold is reached. A network with low traffic may become vulnerable to traffic analysis, because there are not enough data packets to mix with. COVER TRAFFIC creates dummy packets and adopts LINK PADDING to create a continuous flow of dummies between nodes.

Exit nodes in a mix network forward packets to ultimate recipients. Attackers can choose to monitor exit nodes only. RANDOM EXIT is applied with CHAINING to allow any node in a chain to assume the role of an exit node, thereby complicating traffic analysis.

Another goal of anonymity is to enable zero-knowledge communication. OBLIVIOUS TRANSFER describes cryptographic operations for secure multi-party computation.

The patterns in this paper can guide the design of a new PET, or to refactor an existing PET. Retrofitting is hard, except for a well-designed system, e.g., one that follows SSM [22] [21].

The following sections describe the patterns in the order that they appear in Figure 1.

4.1. PSEUDONYMOUS IDENTITY

PSEUDONYMOUS IDENTITY hides anonymity targets under a pseudonym. It is similar to using an alias in a covert organization; only a chosen few know the actual identity, an outsider does not.

Anonymity targets can be personal or impersonal, but everything has some form of identity. The identity of a person is a name, the identity of a packet is by the information in its header, as well as the payload, the identity of a data entry in a database is by its primary key, etc. *If identities are left exposed, entities are exposed too. How can entities be saved from exposure?*

DOI: 10.1002/spe

Hide identity by adopting a random pseudonym that does not relate to the original. Store the pseudonym assignment because the inverse assignment is also required. Protect the pseudonym assignment, because the success of a pseudonym depends on its secrecy.

Two issues are critical for implementing this pattern: finding appropriate anonymity targets and pseudonyms and protecting the pseudonym mapping. Anonymity targets and pseudonyms vary based on the underlying problem domain. Here are some examples,

- For anonymous web interactions, a packet header is an anonymity target because it contains metadata about packet senders. A proxy can be used between a request sender and a recipient to strip header information (e.g., HTTP_USER_AGENT) and submit on behalf of the sender. This is followed by well-known anonymizing tools such as Anonymizer [2] and iProxy [35].
- Anonymous remailers have to hide senders' identity exposed by the their email address. The Penet remailer (anon.penet.fi) [32] provided anonymous email service by hiding users behind pseudonyms. By stripping a user's name and assigning a pseudonym, the system provided sender anonymity through pseudonymity. It also provided recipient anonymity, because a recipient of an email was also behind a pseudonym.
- In location based systems [4] and VANETs [25], user agents are hidden behind a pseudonym. Some VANETs, e.g., Amoeba [55], form a group containing nearby vehicles and use another layer of pseudonym to identify the group. Another anonymity target is the location of a user agent. Gruteser et al. [29] describe a spatial cloaking mechanism that strips exact location information and provides a fuzzy location.
- For privacy-preserving data sharing, the anonymity targets are identifying informations such as names and SSNs. Various systems [63, 52] use pseudonyms for storing electronic health records. Tcpdpriv [41] and Crypto-PAn [20] applies a prefix-preserving permutation to pseudonymize IP addresses in network logs.
- RFID tags are typically hidden behind a pseudonym. This does not prevent attackers from tracking a product, but prevents attackers from identifying it. To prevent tracking, pseudonyms can be refreshed on a frequent basis [36]. Alternatively, a proxy can control a group of RFID tags and selectively simulate tags upon reader requests. RFID Guardian [51] and RFID Enhancer Proxy [38] adopt this technique.

The privacy guarantee of this pattern strongly depends on the second issue: protecting pseudonym mapping. Thielscher and colleague's [63] electronic health records system store the mapping in a separate table offline, but it may still come under attack from insiders [43]. PIPE [52] applies a threshold scheme to share secrets like keys and protects the mapping from becoming a single target.

The part that stores the mapping may become a single point of failure. Also, maintainer of an anonymizing proxy or a remailer can be forced by law enforcement authorities to divulge this information, thereby undermining the anonymity of users. This is the reason why the Penet remailer was shut down. In 1995, the Church of Scientology forced Penet's owner Johan Helsingius to reveal the identity of a user with pseudonym "-AB-" and the anonymous ID `an144108@anon.penet.fi` by winning a court order. Then in 1996, the Church sued Grady Ward [44] under the suspicion that he posted secret files under the posting title "Scamizdat" in the Penet remailer and forced Johan Helsingius to disclose the identity of two users, `an498608@anon.penet.fi` and `an545430@anon.penet.fi`, posting under the handle "DarkDemonStalker". Johan decided to close the remailer in September, 1996 [33].

Some pseudonyms are ephemeral, e.g., agent IDs in a location-based system; others can last quite long, e.g., pseudonymous email addresses; yet others can be permanent, e.g., an anonymizer proxy always submits user information as itself. For pseudonymous email addresses, users can create a reputation by using the pseudonym for a long time. However, long-term use of a pseudonym can weaken privacy because adversaries can salvage context information from a user's correspondence.

Many ActiveX controls require direct linkage between the sender and the recipient. RealAudio goes around the proxy by establishing direct net connections. Ajaxian applications also require direct connection between the browser and the server. In these cases, a proxy server cannot be used.

Pseudonyms should be changed inside a mix system following MORPHED REPRESENTATION.

DOI: 10.1002/spe

4.2. ANONYMITY SET

ANONYMITY SET is a set of equally probable suspects, that can hide an anonymity target. An attacker cannot distinguish the anonymity target from other members of the set. The larger the anonymity set, the stronger the privacy guarantee.

A pseudonym obfuscates an anonymity target, but it does not hide the target. For example, the header of a data packet can be modified, but the packet itself remains exposed. *How can an anonymity target be hidden?*

Create an anonymity set so that an anonymity target is mixed with other equally probable targets. Make the set of probable suspects as large as possible to provide stronger anonymity.

The anonymity set varies according to the meaning of an anonymity target in a domain. Here are some examples of anonymity sets being constructed in different domains.

- An anonymous messaging or browsing system requires sender and/or recipient anonymity. To provide sender anonymity, intermediaries (mix networks [5]) are introduced between message senders and recipients that mix the packets coming from one source with packets coming from different other sources. The anonymity set consists of packets from different sources, and provides sender anonymity. Anon.penet.fi [32] is the first famous remailer. It is a mix network that receives an email from a person, mixes it with other user's email stripping away all identity-revealing information, and remails the message to its final destination.
  For recipient anonymity in messaging, broadcast a message or send it to a message pool instead of one single recipient. The recipient views the message like everyone else, but an adversary cannot tell who the message is for. Hordes [60] uses multicast routing, in which every responder gets every message.
- In a location tracking system, a user may want to hide his or her information because he or she is not interested in a location-aware service, or does not trust the service.
  The concept of mixing also applies here. In this case, a location has patches of areas (e.g., Mix Zone [4]), where an agent's identity is mixed with other agents inside the zone. An adversary can track an agent entering the zone, but cannot monitor inside the zone. Agents assume a different identity when they are entering and/or exiting a mix zone. Thus, an adversary cannot correlate an exiting agent with an entering agent. Vehicular ad hoc networks (VANET) also use mix zone technology [16, 25]. Some VANETs, e.g., Amoeba [55], form a group of nearby vehicles to create an additional anonymity set.
- An electronic voting system should protect the privacy of voters by not revealing their votes. In an anonymous voting system, a voter's vote is passed through an obfuscation mechanism, which mixes it with other votes. The Votegrity system based on by Chaum's secret-ballot receipts [6] and the VoteHere system based on Neff's secret shuffle algorithm [42] use the concept of mix networks for mixing the votes.
- Data sharing mechanism has to assure the subjects that their private data cannot be singled out. When releasing private data sets for public use, specific values of sensitive attributes are changed to to more generalized values, e.g., generalizing gender information (male/female to person), age information (specific age storage to storing in an age group), etc. k-Anonymity [61] follows this principle for publishing secret data.
- Unique RFIDs of products can be hidden behind generalized product-type identifiers (traditional barcode) [56]. This hides a unique product among other products of same type. This can be done at the point of sale.

The size of an anonymity set determines how good the obfuscation is. If the set is small, then an adversary can correlate between input and output with higher probability. If the anonymity set has only one element, then the privacy is provably compromised. For anonymous messaging, the mixing mechanism may stall the data traffic to wait for enough data packets. But this delay may be problematic for low-latency networks. Also, the mixing may fail in a real world scenario, such as no other person entering a mix zone in a location anonymity system.

MORPHED REPRESENTATION is used with ANONYMITY SET to change the data passing through a mix. A single mix may not provide sufficient privacy; CHAINING multiple mix nodes is necessary.

DOI: 10.1002/spe

4.3. CHAINING

Anonymity solutions should be applied in layers to have defense in depth. Even if one anonymity solution fails, another mechanism will maintain the anonymity cover.

ANONYMITY SET mixes an entity with equally probable entities. PSEUDONYMOUS IDENTITY cloaks identity behind an alias, which reinforces the anonymity cover. However, a single anonymity solution may not guarantee strong privacy. For example, an attacker may observe the incoming and outgoing traffic of a single mix node, or even perturb the traffic (e.g., blending attack [30]) to compromise anonymity. Users should also not trust a single mix network, because it may be owned by an active adversary. *How can the anonymity solutions be strengthened?*

Adopt multiple instances of an anonymity mechanism in layers. Chain multiple mix nodes instead of one and route traffic through multiple nodes. Adopt multiple pseudonym covers to protect users.

While the other patterns apply on anonymity targets, CHAINING applies to anonymity solutions. It suggests using multiple mix networks or multiple pseudonyms instead of one. All practical mix based networks for emailing and browsing are chained. Popular remailers such as Morphmix [50], Mixmaster [9] and popular browsers such as Tor [14] route user traffic through multiple mixes.

Similarly, using one pseudonym means that the nymserver, that keeps the mapping between a pseudonym and an anonymity target, becomes a single point of failure. A user can use multiple nymservers and adopt a pseudonym cloaked by another pseudonym. In 1996, the Church of Scientology forced the owner of Penet remailer to disclose the identity of two users, `an498608@anon.penet.fi` and `an545430@anon.penet.fi`. This incident eventually led to the closure of the Penet remailer [33]. Ironically, the Church extorted the information of the pseudonym, but it turned out to be anonymized by another pseudonym by the `alpha.c2.org` nymserver. `alpha.c2.org` was a more advanced remailer that obfuscated the mapping of the input and the output, and hence the Church could not get the conviction they were after.

CHAINING makes traffic analysis, a vulnerability of single-point anonymizers, much more difficult. Adversaries can simply observe incoming and outgoing traffic of a single-point anonymizer, and correlate the encrypted incoming traffic to an anonymizer with the unencrypted traffic going to a receiver by timing analysis. If ten times in a row, a sender's communication with an anonymizer is followed milliseconds later by a request from the anonymizer to a particular site, and that site's response to the anonymizer is followed milliseconds later by an encrypted communication to a sender, then it is a good bet that the sender made a visit to that site. CHAINING makes this type of analysis difficult by introducing multiple intermediaries. Furthermore, it protects user anonymity even if a single mix in the entire chain is honest.

CHAINING creates a stronger ANONYMITY SET. LAYERED ENCRYPTION hides data packets when multiple mixes are chained together.

4.4. MORPHED REPRESENTATION

MORPHED REPRESENTATION adds a layer of obfuscation to the anonymity set by modifying the representation of an entity. Thus, an observer cannot correlate data.

ANONYMITY SET hides data among possible suspects. However, if an entity is represented in the same way before and after creating an anonymity set, it is trivially exposed to an attacker. *How can the representation of an entity be obfuscated?*

Modify the representation of an entity when an anonymity set is created.

The representation is different for different anonymity targets, and so is the modification. The most common anonymity solution is a mix network. But only mixing data packets with probable suspects does not ensure anonymity, because an observer can identify incoming and outgoing traffic by matching data content. The problems exist for any system that follow mix network principles, e.g., anonymous remailer, anonymous browser, location anonymity system, anonymous VANET, etc. Here are some examples of the implementation of this pattern in different mix based systems.

- In mix based networks [5], the incoming packets are encrypted using a key shared between a mix node and its previous node. The mix node decrypts a packet and re-encrypts it with the

DOI: 10.1002/spe

key of the subsequent node. Remailers based on mix network principle, e.g., Mixmaster [9], Babel [30], Mixminion [13] etc, follow this pattern.

- Anonymous web browsing systems based on onion routing [62] also modify the representation of incoming and outgoing data through their use of symmetric key encryption (LAYERED ENCRYPTION). Private web browsing systems for peer-to-peer communication, that provide anonymity following this pattern, include Morphmix [50], Tarzan [23] etc.
- In location anonymity systems [4] and VANET anonymity solutions [16, 25], agents are identified by pseudonyms. When a pseudonymous agent enters into a mix zone, its pseudonym is changed.

Changing representation is also adopted to hide RFID tags, e.g., encrypt RFID tags and create a different representation by reencrypting [37]. The algebraic properties of the El Gamal cryptosystem permit creating different cyphertext representations of a single plaintext with a single public key.

A different implementation of MORPHED REPRESENTATION is adopted in anonymous data sharing. When a data set is shared, various meta-properties of database can reveal information even if an anonymity set is created. In k-Anonymity [61] based database sharing, parts of the data is generalized to create an anonymity set. But in addition to that, the sort order of database entries is changed to obfuscate meta-information.

Changing the representation provides sender anonymity against passive and active observers. In order to protect the content against colluding mix networks, the data has to be encrypted end-to-end. Also, data traffic is encrypted with LAYERED ENCRYPTION, so that each mix network only sees information of where it needs to forward the data next. An active adversary controlling a mix can drop packets and create a denial of service attack, but there are techniques to identify and blacklist the misbehaving mixes. Before initiating a route, a user avoids the blacklisted nodes.

This pattern adds performance overhead, especially when cryptography is used to change packet representation. The capability of cryptographic operations is also limited in low-power application domains, e.g., RFID. Symmetric keys are used as opposed to asymmetric keys, but there is an additional overhead of creating symmetric key shares. Also, there are several trade-offs of the chosen lifetime of a symmetric key share: long-lived keys come under brute force attack, while ephemeral keys have setup overhead. Public keys are durable, but they involve a high computational overhead.

MORPHED REPRESENTATION is used with ANONYMITY SET to hide correlation. Sometimes the data traffic is encrypted with LAYERED ENCRYPTION, where the encryption layers change the representation, as well as protecting confidentiality.

4.5. LAYERED ENCRYPTION

LAYERED ENCRYPTION pattern encrypts mix network data packets in layers, so that each intermediary mix in a chain only sees the information it has to forward.

In a mix network, MORPHED REPRESENTATION pattern modifies packet header and content by decrypting and re-encrypting data. A packet's payload can be encrypted end-to-end, but its header information has to be revealed to a mix. An active adversary controlling a mix may see the routing information from a header and eventually compromise sender and receiver identity. *How can the routing of mix network be modified to retain privacy against an active adversary?*

Before sending data, create a route beforehand and establish a symmetric key share between the intermediaries. Send the information encrypted end-to-end and then encrypted in multiple layers (like an onion skin).

Figure 2 illustrates the onion-like layered encryption scheme between a sender and a receiver. Alice wants to send a message to Bob (address B). Alice creates a route through 3 intermediaries, R_1, R_2 and R_3, in that sequence. Suppose their addresses are A_1, A_2 and A_3, and the symmetric keys shared between Alice and the nodes are K_1, K_2 and K_3, respectively. Alice creates a layered message, encrypting the innermost layer with the symmetric key used in the last hop before the server, the next layer with the symmetric key used in the preceding hop and so on.

Each remailer is able to decrypt the bundle it receives, but it cannot itself look more than one link ahead, let alone determine the final destination. Moreover, after the first link, the sender's identity

DOI: 10.1002/spe

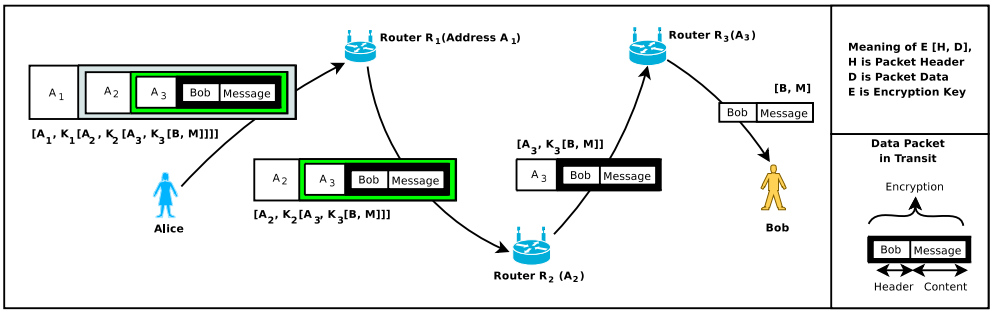

Figure 2. Communication between Alice and Bob using Layered Encryption

has been removed. The first remailer R_1 is connected with the sender but when it receives the message, it has no way to determine whether its previous node is the sender or just another mix node in the remailer chain.

Onion Routing [62] systems are based on mixes with layered encryption. Onion Routing systems have better latency values than mix networks and therefore are more applicable in a web browsing scenario. Private web browsing systems for peer-to-peer communication such as Morphmix [50] and Tarzan [23] follow layered encryption mechanism.

The main overhead of layered encryption is the path setup cost. Typically, it is much less than one second, and it appears to be no more noticeable than other delays associated with normal web connection setup on the Internet. Computationally expensive public key operation is only used during the connection setup phase. By using dedicated hardware accelerators on the routers, the burden of public key operations can be relaxed. Additional overheads of key setup and use are discussed in context with MORPHED REPRESENTATION.

LAYERED ENCRYPTION can be used with proxy-aware applications, as well as several non-proxy-aware applications. It supports various protocols, e.g., HTTP, FTP, SMTP, rlogin, telnet, finger, whois and raw sockets. Proxies can be used with NNTP, Socks 5, DNS, NFS, IRC, HTTPS, SSH and Virtual Private Networks (VPN).

LAYERED ENCRYPTION follows MORPHED REPRESENTATION by performing cryptographic operations at each nodes in the path.

4.6. CONSTANT LENGTH PADDING

CONSTANT LENGTH PADDING suggests making the size of data packets the same in a mix network.

A mix network uses LAYERED ENCRYPTION to protect packet headers and data. However, meta-information about packet content can be leaked from packet size. *How can an adversary be prevented to exploit the size variation in data traffic?*

Make the length of all packets the same. If needed, add padding to data packets.

Mixmaster [9], Mixminion [13], Babel [30] and Tarzan [23] breaks data into uniform-sized packets, and pads the packets to a uniform size.

LAYERED ENCRYPTION uses a mixture of asymmetric and symmetric key cryptography. In such a hybrid key cryptosystem, the size increase is particularly noticeable due to the need to include a random session key in addition to the ciphertext. Conversely, decryption results in shorter messages. Thus the length of an email message will decrease after each decryption as it travels through mixes.

In theory, a cryptosystem, in which encrypted messages are the same size as cleartext message, can be devised, e.g., CFB mode of DES with a pre-distributed initialization vector. In practice, cyphertext is usually somewhat longer than cleartext. Chaum suggests breaking data into fixed size blocks and pad them. Keeping the data chunks of fixed length is important. Otherwise, the padding will vary and reveal information to an active adversary operating a mix node. Fixed length cell with padding is a central idea of onion routing [62].

CONSTANT LENGTH PADDING pattern follows LAYERED ENCRYPTION. By keeping the size similar, it contributes to the ANONYMITY SET.

DOI: 10.1002/spe

4.7. BATCHED ROUTING

BATCHED ROUTING pattern suggests collecting incoming packets in a mix network, and dispatching them in a batch.

ANONYMITY SET in a mix network is created by mixing a sender's incoming packets with packets from other sources, and morphing data representation. However, there remains a strong causal relationship between between incoming and outgoing messages. *Unless this relationship is hidden, an adversary can correlate input with output messages. How can the output of a mix node prevent timing analysis attacks?*

Collect the input data packets, and when the collection reaches a limit, output all the data packets together in a batch.

The main issue is identifying the batching strategy for an application domain. Some examples,

- In a mix network, different batching strategies can be adopted. A threshold mix collects n messages and flushes everything. A timed mix flushes the entire queue every t seconds. A mix can combine the batching strategy, e.g., flushing when the timeout expires and only if a threshold has been reached. Mixmaster [9] and Mixminion [13] follow this strategy, but instead of flushing the entire queue, they flush a fraction of the queue.
- Crowds [49] is a web browser that does not follow mix based technology. Crowds proxies forward web requests, and the final proxy in the path forwards a request to a server. Since a typical web request creates more than one HTTP GET requests, an adversary can launch a timing attack. Crowds end nodes scrape a web page and find all page requests for the page. It batches all the GETs and passes them together.
- Some location-based systems use temporal cloaking [29] mechanism. In this mechanism, a system hides the location information of an agent, until k agents have visited the mix area. The elapsed time makes the spatial coordinates of an agent less accurate.

Batching increases the cost of blending attack [30] and intersection attack [30]. On the other hand, it opens up other issues because it subjects anonymity to be dependent on the behavior of other users. Messages in an idle mix network may be delayed for a long time. An idle mix network may drop a message or route it with a risk of anonymity violation.

BATCHED ROUTING is used with COVER TRAFFIC.

4.8. RANDOM WAIT

RANDOM WAIT suggests adding delays to incoming data traffic of a mix to thwart timing attacks.

BATCHED ROUTING works best for systems with stable traffic patterns. But many application domains do not have stable traffic. Low-latency mixes cannot indefinitely wait for incoming traffic. A VANET or a location-based anonymity solution has no control on the movement of an agent. Systems with variable number of users and with changing traffic conditions may result in low levels of anonymity. An attacker can also take advantage by launching a denial of service attack on the incoming channel of a mix, and force it to drop its stale traffic. *How can a node be implemented to operate without depending on external sources to create anonymity cover?*

Add random delays to make a system non-deterministic. Keep a packet in the pool, and forward it immediately after the delay is passed.

The delay is typically generated by the message sender, who adds it to message header. The random delay can be assigned from a memoryless exponential distribution, or from a uniform distribution. In Stop-and-Go mixes [39], users generate random delay from an exponential distribution, and add this delay to the packet header. The messages are mixed by the randomness of the distribution of delays. The mix never batches a packet, it only follows wait strategy.

In a system following random wait strategy, the delay is independent of the network traffic. Hence, tight delay constraints can be implemented by a mix, regardless of the current load. It is useful for systems for which low latency is more important than providing a high level of anonymity, e.g., web browsers. Unlike a batched mix, the anonymity level may become low when traffic level drops. However, there is always a tradeoff between delay and anonymity for low-latency systems.

DOI: 10.1002/spe

Babel [30] avoids directly using random delay in packets, but it adopts a comparable delaying strategy. It introduces inter-mix detours, a scheme where mix nodes can choose to rewrap a message and send it through a few randomly chosen new hops. After the detour, a message goes back to its original path. This increases the latency, but enhances the anonymity of the messages. For the detour, the mixes use their own encryption technology.

A VANET has no control over the agents. However, it surrounds a mix zone with another region, typically a road intersection, where the agents go through a silent period [55]. The transmission is disabled for a random time in the silent zone.

RANDOM WAIT is used with COVER TRAFFIC.

4.9. COVER TRAFFIC

COVER TRAFFIC pattern suggests generating dummy entities to create a better anonymity set.

ANONYMITY SET requires the presence of other equally probable suspects. In some cases, the physical constraints of a domain threatens to undermine this cover. *How can a better anonymity set be created in domains with practical constraints?*

Generate false entities (cover traffic) and mix them with target entity to create an anonymity set. Make it easy for other anonymizing solutions to detect and discard the false entities, but ensure that adversaries cannot detect them.

Mix-based remailers typically do not need cover traffic because there is no latency constraint. However, Mixmaster [9] and Mixminion [13] use cover traffic to thwart blending attacks [30]. Low latency systems, on the other hand, generate spurious traffic to create anonymity set. Babel [30], under low load, sends out random-looking decoy messages to random destinations. Tarzan [23] uses constant-rate cover traffic between pairs of mixes, sending traffic only between covered links.

Location-anonymity systems [4] introduce dummy traffic following various algorithms [40]. These algorithms target three constraints that make dummy-generation difficult. First, serving dummy users similar to dummy messages is a waste of resources. Although the overhead might be acceptable in the realm of bits, the cost might be too high in real world services. Second, realistic dummy user movement are much more difficult to construct than dummy messages. Third, dummy users have to simulate real world users in controlling physical objects, e.g., opening and closing doors, purchasing services with electronic cash, etc.

For protecting sensitive information in a database, data perturbation approaches [15] are similar to a cover traffic mechanism. In an output perturbation approach, a database first computes an 'exact' answer, but returns a 'noisy' version of it. Data perturbation methods include swapping [48] in which portions of data are replaced with data taken from the same distribution, and fixed perturbations [1] in which a random perturbation is added to every data entry.

Cover traffic creation policy may depend on the load of a mix network, but an attacker can also exploit this. A cover traffic policy, that is independent from traffic load, is more secure. The generated traffic can be inserted to the mix pool, or to the output of a mix. Adding to the output is a more secure option, since it prevents cover traffic to bias the load of a mix network.

COVER TRAFFIC comes with a cost. Crowds [49] avoids this cost, because it only intends to provide plausible deniability. Morphmix [50] does not use cover traffic because it operates in an environment with unreliable nodes, i.e., nodes come and go often. Maintining cover traffic flow between nodes is difficult in this case.

COVER TRAFFIC creates a better ANONYMITY SET. The cover traffic flow in mix networks should be constantly padded between the links (LINK PADDING).

4.10. LINK PADDING

LINK PADDING pattern suggests keeping a constant traffic on each outbound link in a mix.

A passive adversary may monitor the input and output of an anonymity preserving node and be able to find the input and output correlation by monitoring the traffic flow variation in outgoing nodes. Generating cover traffic does not stop traffic analysis, if the cover traffic is only added to some links. Suppose a mix node has three outbound links A, B and C. The node generates cover

DOI: 10.1002/spe

traffic on each node that it sends payload. An adversary may monitor that sometimes link A is busy, on another time links A and C are busy, yet another time all three links are busy and so on. The adversary can shape the traffic and undermine anonymity. *How can a mix base node prevent attackers from detecting traffic variations?*

Keep traffic flowing on all outbound links. Pad the links with cover traffic, even when actual payload is absent.

LINK PADDING hides information about payload traffic, because the statistics of the total output traffic becomes significantly different from that of the payload traffic. ISDN-Mixes [46] and Pipenet [11] use a constant-rate cover traffic along the length of the path, i.e., by sending messages at a constant rate through each path. Tarzan [23] nodes maintain a bidirectional packet stream with a fixed number of other nodes (mimics). A node only sends data through one of its mimics, hence data is always sent among a constant stream of cover traffic. Location-anonymity systems constantly generate dummy traffic in some fixed directions between mix zones.

Keeping a constant rate of packet flow between links may come under attack in practical scenarios, because operating system timers may fail to keep a constant rate when there is heavy load. This clock skew reveals payload information. Link padding with variable inter-arrival time between packets provides better protection [26]. Again, keeping a constant flow has high cost, especially when packet flow is low. Adaptive link padding techniques [67] tackle this issue.

LINK PADDING follows COVER TRAFFIC pattern. It affects the cover traffic generation policy.

4.11. RANDOM EXIT

RANDOM EXIT suggests adding a variation to packet routing mechanism by allowing mix networks to forward packet to a recipient instead of forwarding to the next hop.

The exit node of an anonymity network forwards a packet to its final recipient. Exit nodes often come under abuse. Having a few exit nodes reduces the number of points an adversary needs to monitor. *How can an anonymity service prevent exit abuse?*

Allow traffic to exit an anonymity network not only at the endpoints a circuit, but also in the middle of a circuit.

Tor [14] initiators can direct traffic to exit partway down the circuit, by using in-band signaling within the circuit. This frustrates traffic shape and volume attacks based on observing the end of the circuit. Crowds [49] proxies on the path of a web request can locally decide, based on a probability of forwarding (p_f), whether to forward traffic through another proxy, or become the last node on the path and communicate with the recipient directly.

4.12. OBLIVIOUS TRANSFER

OBLIVIOUS TRANSFER is a transaction between an initiator and a respondent without revealing the initiator's private information to the respondent.

Messaging is amenable to anonymity because if a message sender or a recipient hides its identity and does not reveal any traits in the message, he or she can remain hidden. For transactions that require users to share identifying attributes between them, it is much harder to remain private. Suppose, Alice and Bob are two millionaires who want to find out who is richer without revealing the precise amount of their wealth [70]. *How can a user involve in a zero-knowledge communication?*

Adopt an oblivious transfer protocol, in which a sender transfers one of potentially many pieces of information to a receiver, but remains oblivious as to what piece (if any) has been transferred.

In 1-out-of-2 oblivious transfer, one party, the sender, has input composed of two strings ($M0$,$M1$), and the input of a second party, the chooser, is a bit a. The chooser should learn Ma and nothing regarding $M\bar{a}$ while the sender should gain no information about a.

Similarly, in private information retrieval [45], a database (sender) transmits some of its items to a user (chooser), in a manner that preserves mutual privacy. The database has assurance that the user does not learn any information beyond what he or she is entitled to. The user has assurance that the the database is oblivious or unaware of which particular information is consumed by the user.

DOI: 10.1002/spe

Private matching and private set intersection schemes [24] consider the problem of computing an intersection of two private sets but not revealing anything beyond that.

Oblivious transfer has practical applications other than privacy. Shady domain name services were involved in a scheme called front running [34]. When a user searches for a domain name, a domain name service registers the domain name for itself and charges a user more to get the domain. A private information retrieval method effectively hides the domain name query from a malicious domain name service.

5. SOLVING EMERGING PROBLEMS

Emerging problems are often similar to old problems and can be solved by existing patterns. This section describes how patterns in our catalog may assist users in designing emerging PETs. We explore 3 PETs for pseudonymous mail management, privacy-preserving face recognition and private instant messaging with stronger anonymity guarantee.

Pseudonymous Mail Retrieval. A pseudonymous mail retrieval system allows a sender to send and receive messages at a pseudonymous address. Pynchon Gate [57] supports pseudonymous mail retrieval by combining private information retrieval primitives with mix network and pseudonymous servers. Nymservers create PSEUDONYMOUS IDENTITY for subscribed users. The mails are routed through a mix network, which provides sender anonymity. Technically, any mix network can be used as a transport; the mix networks follow the standard patterns for mix network. In order to create ANONYMITY SET at the receiver, a collator component is introduced at the nymservers. The collator component follows BATCHED ROUTING for mixing, and forwards the received messages into a pool. It does not employ COVER TRAFFIC, since the threshold for batched forwarding is typically long enough to ensure a strong ANONYMITY SET. The actual receivers employ an OBLIVIOUS TRANSFER mechanism to query the message pool and select the messages for them.

Privacy-preserving Face Recognition. Face recognition technology, that has been deployed in public places, pose a threat to the privacy of individuals. People do not want to be profiled. Yet, they are willing to let law-enforcement agencies adopting these techniques, because they believe that this will ensure security in the long run. In a real world scenario, a face recognition technology is often run by a private organization, e.g., a bank, a train station, or a shopping mall. They typically match their collected data with the data stored in a confidential database, which keeps facial information of individuals. The untrusted organization running the technology should be able to identify whether a perpetrator is in the locality, but they should not be able to use the entire database for some other purpose, e.g., profiling. Erkin et al. [18] implemented a PET for face-recognition, which allows to efficiently hide both the biometrics and the result from the server that performs the matching operation, by using OBLIVIOUS TRANSFER pattern. In their system, the private organization provides a face image, while a trusted party, e.g., the police, has access to a database of facial templates. Their PET allows to jointly run the standard Eigenfaces recognition algorithm in such a way that the untrusted party cannot learn more than basic parameters of the database, while the trusted party cannot learn the input image or the result of the recognition process.

Anonymity for real-time instant messaging and voice-over-IP Communications. We have described various mix based technologies to protect communication pattern against adversaries with limited capability. Drac [12] is the first system to be designed to withstand a global passive adversary to protect communication pattern in real-time instant messaging and voice-over-IP communication. It is a peer-to-peer based mix technology, in which a circuit for a communication is created using a friend-of-a-friend network. The ANONYMITY SET is created by keeping COVER TRAFFIC and a LINK PADDING scheme. The data is encrypted using LAYERED ENCRYPTION pattern. The nodes also assume pseudonyms following PSEUDONYMOUS IDENTITY pattern. Drac gives away the

DOI: 10.1002/spe

identity of a users friends to guarantee the unobservability of actual calls, while still providing anonymity when talking to untrusted third parties. The use of LINK PADDING means that the architecture is suitable for regular, low-volume traffic, i.e., for real-time instant messaging and voice-over-IP communication. The low-volume link padding and friend-of-a-friend network create a smaller anonymity set, but they are harder than random anonymity sets, because an adversary has to infiltrate the social circle of a user to perform insider attacks.

6. PRIVACY PATTERNS FROM OTHER SOURCES

There has not been a lot of work on privacy patterns. Schumacher [58] described two privacy patterns: PROTECTION AGAINST COOKIES pattern describes how to control the cookies in a web client, and PSEUDONYMOUS EMAIL pattern describes the mechanism of a pseudonymous email delivery system. Till Schummer, in his paper on information filtering in collaborative systems [59], described patterns that block the transmission of personal information. Sadicoff et al. [54] described one privacy pattern: a privacy proxy that helps inform users of a website's privacy policies. INFORMED CONSENT FOR WEB-BASED TRANSACTIONS pattern in Romanosky and colleague's [53] paper is similar in concept. Their paper described two privacy patterns for online interactions. MASKED ONLINE TRAFFIC pattern suggests that identifiable information should be stripped off online traffic. MINIMAL INFORMATION ASYMMETRY pattern suggests that users should have more information about how the websites handle private data. Chung et al. [7] described 15 patterns that focused on privacy in ubiquitous computing. These patterns originated from their collective experience in human computer interaction. Their intent was not to describe privacy patterns, but to find out how patterns helped users design better systems. Their results [7] suggested that the privacy patterns that they identified were not useful, since they were relatively abstract.

All the previous privacy patterns were too general. For example, MASKED ONLINE TRAFFIC pattern [53] suggests hiding online traffic, but it does not describe how to build PETs for hiding. Many of these patterns are about bridging the trust gap between users and online services by allowing users to efficiently collect information about how services handle their private information.

The patterns in this paper document concrete design decisions made by expert PET designers. These patterns can be reused in various domains for designing and implementing PETs.

7. CONCLUSION

This paper describes 12 privacy patterns and a pattern language that shows their relationships. These patterns can help developers design PETs in various domains; we have also described how the patterns can be useful in designing PETs for emerging problems.

We expect the catalog to grow, primarily because emerging problems will warrant new solutions and new patterns. However, experts for specific PET domains may identify domain-specific patterns and relate them to the patterns in the existing catalog. We listed architectural patterns only; a different scope may have resulted in a different catalog. We also omitted patterns, especially cryptographic patterns, that are the building blocks of many of the patterns in this catalog. Although they are used in PETs, they are not privacy patterns, but are security patterns and are part of our larger security pattern catalog. We will continue to explore new privacy patterns keeping these issues in mind. However, our catalog is a first step in understanding and documenting the design decisions of PETs. It allows developers to have a deep understanding of the privacy domain as they implement their solutions.

Acknowledgement

This paper has been written incrementally over the last 5 years. I am grateful to Paul Adamczyk, Nikita Borisov, Nicholas Chen, Eduardo Fernandez, Ralph Johnson, and the anonymous reviewers.

DOI: 10.1002/spe

REFERENCES

1. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, SIGMOD '00, pages 439–450, New York, NY, USA, 2000. ACM.
2. Anonymizer.com. Online privacy services.
3. I. Antón and B. Earp. A requirements taxonomy for reducing web site privacy vulnerabilities. *Requir. Eng.*, 9:169–185, August 2004.
4. A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE PerCom*, 2(1):46–55, 2003.
5. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
6. D. Chaum. E-voting: Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, 2(1):38–47, Jan./Feb. 2004.
7. E. S. Chung, J. I. Hong, J. Lin, M. K. Prabaker, J. A. Landay, and A. L. Liu. Development and evaluation of emerging design patterns for ubiquitous computing. In *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques*, DIS '04, pages 233–242, New York, NY, USA, 2004. ACM.
8. I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009:46–??, 2001.
9. L. Cotrell. Mixmaster & remailer attacks, 1995.
10. L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall, and J. Reagle. *The Platform for Privacy Preferences 1.0 (P3P1.0) Specification*. W3C, April 2002.
11. W. Dai. Pipenet 1.1. Usenet post, August 1996.
12. G. Danezis, C. Diaz, C. Troncoso, and B. Laurie. Drac: an architecture for anonymous low-volume communications. In *Proceedings of the 10th international conference on Privacy enhancing technologies*, PETS'10, pages 202–219, Berlin, Heidelberg, 2010. Springer-Verlag.
13. G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *Proceedings of the 2003 Symposium on Security and Privacy*, pages 2–15. IEEE Computer Society, 2003.
14. R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320, 2004.
15. I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM Symposium on Principles of Database Systems (PODS-03)*, pages 202–210, New York, 2003. ACM Press.
16. F. Dötzer. Privacy issues in vehicular ad hoc networks. In G. Danezis and D. Martin, editors, *Privacy Enhancing Technologies*, volume 3856 of *Lecture Notes in Computer Science*, pages 197–209. Springer, 2005.
17. EPIC. Privacy & human rights, An international survey of privacy laws and developments. *Electronic Privacy and Information Centre and Privacy International*, 2002.
18. Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, PETS '09, pages 235–253, Berlin, Heidelberg, 2009. Springer-Verlag.
19. J. Evans, W. Holfelder, and M. Kawamoto, editors. *Proceedings of the First International Workshop on Vehicular Ad Hoc Networks, VANET 2004, Philadelphia, PA, USA, October 1, 2004*. ACM, 2004.
20. J. Fan, J. Xu, M. H. Ammar, and S. B. Moon. Prefix-preserving IP address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. *Computer Networks (Amsterdam, Netherlands: 1999)*, 46(2):253–272, Oct. 2004.
21. M. Fayad and S. Wu. Thinking objectively: Merging multiple conventional models in one stable model. *Communications of the ACM*, 45(9):102–106, 2002.
22. M. E. Fayad and A. Altman. An introduction to software stability. *Commun. ACM*, 44(9):95–98, 2001.
23. M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, DC, November 2002.
24. M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2004.
25. J. Freudiger, M. Raya, M. Félegyházi, P. Papadimitratos, and J.-P. Hubaux. Mix-zones for location privacy in vehicular networks, 2007.
26. X. Fu, B. Graham, R. Bettati, and W. Zhao. On effectiveness of link padding for statistical traffic analysis attacks. In *23th International Conference on Distributed Computing Systems (23th ICDCS'03)*, pages 340–, Providence, RI, May 2003. IEEE Computer Society.
27. I. Goldberg. Privacy-enhancing technologies for the Internet, II: Five Years Later. In *International Workshop on Privacy Enhancing Technologies (PET), LNCS*, volume 2, 2002.
28. I. Goldberg, D. Wagner, and E. Brewer. Privacy-enhancing technologies for the Internet. In *Proc. of 42nd IEEE Spring COMPCON*. IEEE Computer Society Press, Feb. 1997.
29. M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st international conference on Mobile systems, applications and services*, MobiSys '03, pages 31–42, New York, NY, USA, 2003. ACM.
30. C. Gülcü and G. Tsudik. Mixing e-mail with BABEL. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS '96)*, San Diego, California, Feb. 1996. Internet Society.
31. M. Hafiz, P. Adamczyk, and R. E. Johnson. Organizing security patterns. *IEEE Software*, 24(4):52–60, 2007.
32. J. Helsingius. The `anon.penet.fi` anonymous server. help file, 1995.
33. J. Helsingius. Johan Helsingius closes his Internet remailer, 30 Aug 1996.
34. ICANN Security and Stability Advisory Committee. Report on domain name front running, SAC 024, Feb. 2008.
35. iProxy.net. iProxy anonymizer service. `http://iproxy.net/`.
36. A. Juels. Minimalist cryptography for low-cost rfid tags (extended abstract). In *Security in Communication Networks*, volume 3352 of *Lecture Notes in Computer Science*, pages 149–164. Springer Berlin / Heidelberg, 2005.

DOI: 10.1002/spe

37. A. Juels and R. Pappu. Squealing euros: Privacy protection in rfid-enabled banknotes. In *Financial Cryptography*, volume 2742 of *Lecture Notes in Computer Science*, pages 103–121. Springer Berlin / Heidelberg, 2003.
38. A. Juels, P. F. Syverson, and D. V. Bailey. High-power proxies for enhancing RFID privacy and utility. In *Privacy Enhancing Technologies*, volume 3856 of *Lecture Notes in Computer Science*, pages 210–226. Springer, 2005.
39. D. Kesdogan, J. Egner, and R. Bschkes. Stop- and- go-mixes providing probabilistic anonymity in an open system. In *Information Hiding*, volume 1525 of *Lecture Notes in Computer Science*, pages 83–98. Springer, 1998.
40. H. Kido, Y. Yanagisawa, and T. Satoh. Protection of location privacy using dummies for location-based services. In *ICDE Workshops*, page 1248, 2005.
41. G. Minshall. "Tcpdpriv", release 1.1.10, Aug. 1997.
42. C. A. Neff. A verifiable secret shuffle and its application to e-voting. In *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS 2001)*, pages 116–125. ACM Press, November 2001.
43. T. Neubauer and A. Ekelhart. An evaluation of technologies for the pseudonymization of medical data. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 857–858. ACM, 2009.
44. R. Newman. The Church of Scientology vs. Grady Ward, 24 Jul 1996.
45. F. Olumofin and I. Goldberg. Privacy-preserving queries over relational databases. In M. Atallah and N. Hopper, editors, *Privacy Enhancing Technologies*, volume 6205 of *Lecture Notes in Computer Science*. 2010.
46. A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-MIXes: Untraceable communication with very small bandwidth overhead. In *Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, pages 451–463. Springer-Verlag, 1991.
47. A. Pfitzmann and M. Waidner. Networks without user observability - Design options. In F. Pichler, editor, *Advances in cryptology: Proceedings of a Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT '85)*, volume 219 of *LNCS*, pages 245–253, Linz, Austria, Apr. 1985. Springer.
48. S. P. Reiss. Practical data-swapping: the first steps. *ACM Trans. Database Syst.*, 9:20–37, March 1984.
49. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
50. M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-peer based anonymous Internet usage with collusion detection. In *WPES '02: Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 91–102, New York, NY, USA, 2002. ACM Press.
51. M. R. Rieback, B. Crispo, and A. S. Tanenbaum. RFID guardian: A battery-powered mobile device for RFID privacy management. In C. Boyd and J. M. G. Nieto, editors, *ACISP*, volume 3574 of *Lecture Notes in Computer Science*, pages 184–194. Springer, 2005.
52. B. Riedl, V. Grascher, S. Fenz, and T. Neubauer. Pseudonymization for improving the privacy in e-health applications. In *Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, HICSS '08, pages 255–, Washington, DC, USA, 2008. IEEE Computer Society.
53. S. Romanosky, A. Acquisti, J. Hong, L. F. Cranor, and B. Friedman. Privacy patterns for online interactions. In *Proceedings of the 2006 conference on Pattern languages of programs*, PLoP '06, 2006.
54. M. Sadicoff, M. M. Larrando-Petrie, and E. B. Fernandez. Privacy aware network-client pattern. In Proceedings of the 12th Conference on Patterns Language of Programming (PLoP'05), 2005.
55. K. Sampigethaya, M. Li, L. Huang, and R. Poovendran. AMOEBA: Robust location privacy scheme for VANET. *IEEE Journal on Selected Areas in Communications*, 25(8):1569–1589, 2007.
56. S. Sarma, S. Weis, and D. Engels. RFID systems and security and privacy implications. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin / Heidelberg, 2003.
57. L. Sassaman, B. Cohen, and N. Mathewson. The Pynchon Gate: A secure method of pseudonymous mail retrieval. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, WPES '05, 2005.
58. M. Schumacher. Security patterns and security standards - with selected security patterns for anonymity and privacy. In Proceedings of the European Conference on Patterns Language of Programming (EuroPLoP'02), 2002.
59. T. Schummer. The public privacy - patterns for filtering personal information in collaborative systems. In Proceedings of CHI workshop on Human-Computer-Human-Interaction Patterns, 2004.
60. C. Shields and B. N. Levine. A protocol for anonymous communication over the Internet. In *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security*, pages 33–42, 2000.
61. L. Sweeney. k-Anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
62. P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy*, pages 44–54, Oakland, California, 4–7 May 1997.
63. C. Thielscher, M. Gottfried, S. Umbreit, F. Boegner, J. Haack, and N. Schroeders. Patent: Data processing system for patient data. *International Patent, WO 03/034294 A2*, 2005.
64. G. van Blarkom, J. Borking, and J. Olk, editors. *Handbook of Privacy and Privacy-Enhancing Technologies: The case of Intelligent Software Agents*. College bescherming persoonsgegevens, 2003.
65. H. van Rossum, H. Gardeniers, J. Borking, A. Cavioukian, J. Brans, N. Muttupulle, and N. Magistrale. Privacy Enhancing Technologies – The path to anonymity. Technical report, Registratiekammer, The Netherlands and Inofrmation and Privacy Commissioner, Ontario, Canada, 1995.
66. J. Viega and G. McGraw. *Building Secure Software: How to Avoid Security Problems The Right Way*. Addison-Wesley, 2002.
67. W. Wang, M. Motani, and V. Srinivasan. Dependent link padding algorithms for low latency anonymity systems. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, pages 323–332, New York, NY, USA, 2008. ACM.
68. S. D. Warren and L. D. Brandeis. The right to privacy. *Harvard Law Review*, 4(5):193–220, December 1890.
69. A. F. Westin. *Privacy and Freedom*. Atheneum, NY, 1967.
70. A. C. Yao. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, pages 160 –164, 1982.

DOI: 10.1002/spe