Thinking Towards a Pattern Language for Predicate Based Encryption Crypto-Systems

Jan de Muijnck-Hughes and Ishbel Duncan
School of Computer Science,
University of St Andrews,
St Andrews, Fife, UK
{jfdm, ishbel.duncan}@st-andrews.ac.uk

***Abstract*—Predicate Based Encryption (PBE) is a novel family of public key encryption schemes that allows for expressive, and fine-grained, access control to be integrated within the cryptographic process. Providing an efficient means to realise distributed encrypted access control. Security patterns allow for security problems and their solutions to be described concretely and precisely, and be applied directly within the software development process. Pattern languages provide a means to specify how a set of interconnected patterns can be used together to solve a set of related problems. This paper proposes the construction of a pattern language governing the design and deployment of PBE crypto-systems. An overview for the proposed language is given together with a discussion towards issues affecting its specification.**

***Index Terms*—Data Security, Cryptography, Access Control, Web Services, Security Patterns, Pattern Languages**

I. Introduction

Traditional public key encryption schemes provide too fine-a-grained access control over encrypted data. By design, decryption can only be specified on a per entity basis. For each intended decryptor a separate decryption key will be required. There is a gap between the access control policy that is specified and its representation as provided by the encryption scheme. Expressive styles of encrypted access control cannot be implemented efficiently using these traditional schemes.

Predicate Based Encryption (PBE) is family of public key encryption schemes that allows for Attribute Based Access Control (ABAC) to be integrated directly within the cryptographic operations [1]. Generally speaking, within PBE the decryption of a cipher-text is dependent upon the satisfaction of a predicate by a set of attributes, allowing for one-to-many decryption of cipher-texts. PBE schemes allow for an efficient means through which expressive, and fine-grained, access control can be specified over encrypted data. Crypto-systems can make use of PBE to provide both distributed access control, and expressive keyword search over encrypted data. Research towards PBE has concentrated on the construction of PBE schemes, and several different implementations of a PBE scheme have been used to provide encrypted access control [2]–[5]. However, it is not clear what the 'best practises' are regarding the general application, and deployment, of PBE schemes as part of a crypto-system. That is, what are the complete cryptographic measures required, including key management and implementation worries, that an organisation needs to implement to provide guarantees towards confidentiality and access control when using PBE. It is not clear how non-experts in both the security and applied cryptography domain should, and could, use PBE within their crypto-systems.

Within software engineering a pattern represents a specific solution to a recurrent problem. Design patterns can be used to build conceptual models of systems, and security patterns to build secure systems. Patterns can be used to encapsulate best practises for the problem they solve, and be applied during each stage of the software development process [6].

This paper proposes the construction of a pattern language governing how PBE can be deployed, and used as part of a crypto-system to provide distributed access control over encrypted data. By using this pattern language the 'best practises' concerning the deployment of PBE crypto-systems can be captured, formalised, and made accessible to those who lack expertise within the security, and applied cryptography domains.

This paper is organised as follows: Section II introduces PBE Schemes and Crypto-Systems. Section III provides an overview of security patterns. An overview of the proposed pattern language is given in Section IV. Section V presents the proposed research methodology. Section VI discusses several design challenges for the pattern language. The evaluation of security patterns is discussed in Section VII. Related work is discussed in Section VIII. Finally, Section IX concludes the paper and discusses future work.

II. Predicate Based Encryption

PBE is a family of public key encryption schemes that allows for ABAC to be integrated within the decryption process [1]. In this section PBE schemes, and PBE crypto-systems are described.

A. Schemes

In Attribute Based Encryption (ABE) the decryption of a cipher-text is determined by the satisfaction of an access policy, represented as a monotone boolean formula, by a set of attributes [7]. Since a boolean formula can be potentially satisfied by many sets of attributes, a single cipher-text can be decrypted by many decryption keys: Encryption is one-to-many. ABE schemes make use of general predicates only, PBE schemes build upon ABE schemes in that specific

predicates, such as inner-product, can also be used to construct an encryption scheme. PBE is also a known sub class of 'Functional Encryption' [8]. The use of the predicate to construct the decryption key, or encryption key, gives rise to two types of PBE scheme.

Ciphertext-Policy (CP) schemes use predicates as encryption keys, and decryption keys are derived from a set of attributes [9]. The functionality afforded by CP schemes directly mimics that of ABAC. CP-PBE schemes offer a means to provide distributed encrypted access control within a system. Subjects i.e. decrypting entities, can only access the resource (cipher-text) if their attributes (decryption key) satisfies the access policy (encryption key) associated with the resource.

Schemes that use predicates to construct decryption keys, and sets of attributes to construct encryption keys, are known as Key-Policy (KP) schemes [7]. KP-PBE schemes offer a means to provide expressive keyword search over encrypted data. During encryption cipher-texts are *tagged* with a set of keywords (the encryption key) and decryption keys act as search queries over cipher-texts.

A general PBE scheme consists of the following four operations:

- **Setup**: initialises the crypto-scheme and generates a master secret key MSK, used to generate decryption keys, and a set of public parameters MPK.

$$(\mathsf{MSK}, \mathsf{MPK}) := \mathrm{Setup}() \tag{1}$$

- **KeyGen**: generates a decryption key $\mathsf{Dec}(entity)$ based upon the master secret key and some entity supplied input.

$$\mathsf{Dec}(entity) := \mathrm{KeyGen}(\mathsf{MSK}, \mathrm{input}) \tag{2}$$

- **Encrypt**: encrypts a plain-text message M using the public parameters and supplied encryption key for an entity.

$$CT := \mathrm{Encrypt}(\mathsf{Enc}(M), \mathsf{MPK}, M) \tag{3}$$

- **Decrypt**: decrypts a cipher-text if and only if the attributes held by the entity can satisfy the access policy.

$$M := \mathrm{Decrypt}(\mathsf{Dec}(entity), \mathsf{MPK}, CT) \tag{4}$$

where 'input' and $\mathsf{Enc}(M)$ will either be a predicate or set of attributes depending on the type of PBE scheme. Several implementations of PBE schemes are listed within Boneh et al. [8].

Remark 1: The two types of PBE scheme each beget a different style of access control. CP schemes provide a form of discretionary access control, and KP schemes provide a form of mandatory access control.

B. Crypto-Systems

PBE has already be used to provide guarantees towards the confidentiality of data as part of a crypto-system [2]–[5]. Each of these systems represents a specific, and application dependent, deployment of a PBE based crypto-system. The systems presented in references [2]–[5] do not address the generic deployment of PBE crypto-systems. To address this problem de Muijnck-Hughes [10] introduced five generic deployment scenarios for PBE crypto-systems.

When used to provide public key encryption services, a PBE crypto-system will consist of three types of entity: the *Key Authority*—the entity responsible for creating, and assigning, decryption keys; *Encryptors*—entities that encrypt data; and *Decryptors*—entities that decrypt data. Both encryptors and decryptors will interact with the key authority to obtain both the MPK, and information to construct encryption keys (encryptor) and $\mathsf{Dec}(entity)$—decryptor. This is the prototypical deployment scenario for a public key encryption crypto-system. Other deployment scenarios can be specified by varying: a) ownership of the key authority, either a service provider or service user; b) the type of entity owning the key authority, either a decryptor, encryptor, or neither i.e. independent; and finally c) the type of PBE scheme used either Ciphertext-Policy or Key-Policy. These variations result in the following five generic deployment scenarios.

- **PBE-within-a-Service**: A service provider is the Key Authority and embeds a CP-PBE scheme within their own service offerings. Users of the service, acting as Encryptors and Decryptors, can use the PBE scheme to selectively share their data amongst each other.
- **PBE-as-Service**: A service provider is the Key Authority and provides a service offering CP-PBE. This affords service users, acting as Encryptors, and Decryptors, the ability to selectively share their data outwith the confines of a particular service.
- **'Database' Access**: A service provider acts as the Key Authority and Encryptor, encrypting their data using KP-PBE. Service users, acting as Decryptors, can then be provided decryption keys as a means to provide expressive keyword search over encrypted data.
- **'Database' Submission**: Using KP-PBE a service provider can be the Key Authority and employees/affiliates of the service provider as Decryptors. Service users, acting as Encryptors, can submit content to the service provider. Within the service providers domain access to submitted content can be restricted using the decryption keys.
- **Distributed Security**: Service users act as the Key Authority and use CP-PBE to selectively control access to their encrypted data, accross multiple services to other service users, and providers. Decryptors i.e. other service providers and users, can be provided decryption keys as a means to restrict their access to the encrypted data.

The five deployment scenarios each represents a solution to a different recurring deployment problem regarding PBE crypto-systems. However, the descriptions by de Muijnck-Hughes [10] are not well documented nor are they formal. There is a need for well documented, formal descriptions of the deployment of PBE crypto-systems.

III. Security Patterns

Patterns represent a well documented solution to a recurrent problem within a particular context. Software patterns present

a 'separation of concerns' between solution conception and solution application. Design patterns can be used to document the *tried and tested* design decisions made by software engineers and will encapsulate the designer's experience within the pattern [11]. Patterns provide non-domain experts a means to access a domain experts knowledge. Patterns can be applied at all levels of the software development lifecycle. Moreover, a single pattern may not be enough to describe the solution to a set of problems. Pattern languages provide a means to specify how a set of interconnected patterns can be used together to solve a set of related problems [12].

Security patterns are patterns that describe solutions to recurring security related problems [6]. Not only can security patterns be used to describe well known security concepts but also security mechanisms within software engineering. Pattern-oriented approaches are increasingly being used to develop secure systems [13]. Security pattern languages are pattern languages within the security domain. By using security patterns complex security concepts, and mechanisms, can be expressed concisely and explicitly such that non-domain experts can understand them, and consequently use them [14]. Moreover, Cuevas et al. [15], [16], use a pattern-based approach to model *Untraceable Secret Handshakes*, a cryptographic protocol that allows two parties within the same group to share a secret key and also prove group membership. Patterns can be used to represent advanced crypto-systems.

The five scenarios, presented in Section II-B can be used to realise different security concepts. In fact these five scenarios are reminiscent of several existing security patterns. The two database-oriented patterns could be used to implement the asynchronous secure channel pattern (both directions) [6], and the two *as-a-service* patterns could be used to realise Brokered (PBE-as-a-Service) and Direct Authentication—PBE-within-a-Service [17]. However, these existing patterns represent application areas in which PBE can be used to provide guarantees. These patterns do not address the fundamental cryptographic operations, and key management operations, required for PBE crypto-systems; they were created before the advent of PBE.

Given the view from Delessy et al. [14], and the work from Cuevas et al. [15], [16], it naturally follows that a pattern-based approach can be used to describe advanced crypto-systems that utilise state-of-the-art cryptographic techniques. Namely, a pattern-language can be constructed that can be used to specify, deploy, and use PBE crypto-systems. The proposed pattern-language is described in the next section.

IV. Pattern Language Overview

The pattern language is envisaged as a multi-layered pattern diagram that details how each of the five deployment scenarios can be realised. Fig. 1 contains the pattern diagram for the proposed language. Each layer in the diagram represents a different layer of abstraction: Higher layers represent more abstract concepts, and lower layers more concrete realisations. Within each layer there will be patterns that provides solutions to the problems associated with each layer.

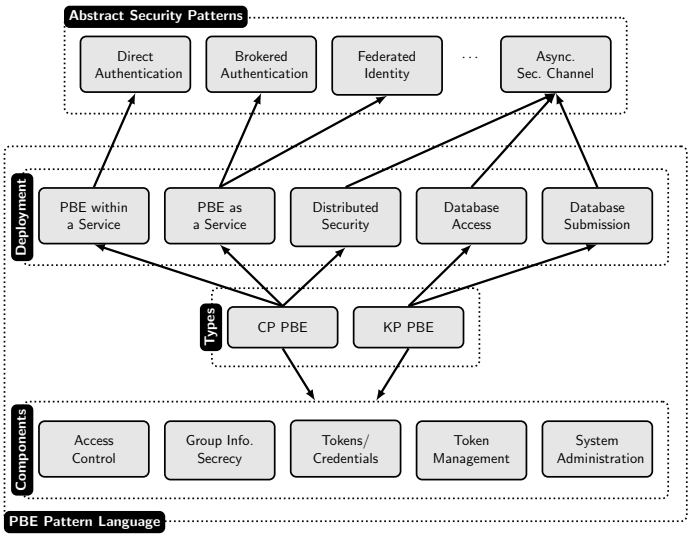

Fig. 1. Pattern Diagram for PBE Crypto-Systems

Recall from Section III, the five deployment scenarios for PBE can be used to provide solutions for known security patterns. Examples of these known patterns are given in the top most layer of Fig. 1. The highest layer in the language will represent each of the five deployment scenarios presented as individual patterns. This represents the main interface to the language.

The next layer in the diagram presents patterns for the two types of PBE crypto-systems: Ciphertext-Policy and Key-Policy. These two types of crypto-system will each represent a more generalised deployment of a PBE crypto-system based upon the underlying type of PBE scheme, and access control required. Within each deployment pattern there will always be a need for a key authority, an encryptor, and a decryptor. It will be the location of these entities and their possible interactions that will change. Although these two patterns will use different access control styles, they can nonetheless be described as a different deployment of an even more generic PBE crypto-system.

The remaining layers of the PBE pattern language will consist of patterns that when combined can be used to model, and also implement, a generic PBE crypto-system. These patterns represent the common components found within PBE crypto-systems. Within this layer the patterns will be a combination of existing patterns and patterns specific for implementing PBE. These different patterns can be grouped into five areas: a) Group Information Secrecy—patterns modelling the cryptography; b) Access Control—patterns modelling the access control; c) Tokens/Credentials—representing cryptographic keys; d) Token Management—patterns for key management; and finally e) System Administration—patterns modelling the Key Authority. Section VI discusses the issues forseen in pattern selection and creation for this layer.

V. Methodology

The main goal for our pattern language is to provide a separation of concerns concerning the deployment of PBE

crypto-systems. To achieve this goal the pattern language outlined within Section IV will be constructed and later evaluated.

Before the construction of the pattern language, the existing security pattern landscape will be surveyed to identify any existing patterns suitable for inclusion within the pattern language. Bunke et al. [18] recently studied the security pattern landscape, enumerating and classifying over 300 different security patterns. This work will be used as a basis for our own investigation.

A bottom-up approach will be taken towards the actual development and construction of the pattern language. Patterns representing the five different component areas of a generic PBE crypto-system will be addressed first—The component layer from Fig. 1. This layer provides an ideal separation between the patterns required for implementing PBE in software, and patterns concerned with deployment. The challenges forseen with this layer are discussed further in Section VI. Next, the layer representing the two types of PBE crypto-system will be addressed. Finally, the patterns representing the deployment layer of the language will be constructed.

To determine the success of the resulting pattern language it will be evaluated using techniques taken from previous studies of the pattern language landscape [19]–[21]. Section VII discusses further our intended direction concerning pattern language evaluation.

VI. Challenges Affecting Construction

Central to the pattern language is the construction of a generic PBE crypto-system. Section IV introduced five areas that deal with different functional units within a generic PBE crypto-system. This section discusses the challenges arising from selecting/designing patterns for each of these areas. Although, each of the five areas are to be discussed individually, another challenge is the combination of patterns when specifying patterns for a generic PBE crypto-system. Moreover, this paper is concerned with the overall design of the pattern language. Lower level patterns that address the implementation in software of a generic PBE crypto-system will not be included in this discussion.

A. Group Information Secrecy

Patterns exist that present patterns for the use of public key cryptography to provide guarantees towards data confidentiality [22], [23]. However, they were specified before the advent of PBE. These patterns are not suitable for modelling PBE schemes. These patterns do not address the one-to-many encryption and relation to ABAC that is inherent within PBE. New patterns need to be constructed that describes how to achieve *Group Information Secrecy*. Furthermore, these new patterns need to take into account both KP-PBE and CP-PBE schemes.

B. Access Control

Although *Group Information Secrecy* will model the confidentiality aspect of a PBE scheme, the style of access control also needs to be modelled. Patterns addressing access control, especially within distributed systems, have been presented [24]–[26]. The challenge here will be to see what, and how, patterns addressing ABAC can be incorporated with the patterns from the other four areas. For instance, KP-PBE and CP-PBE both promote two different styles of access control: discretionary, and mandatory. In fact these styles of access control could be modelled by the Capability pattern [25], and ABAC pattern [26].

C. Token/Credentials

Within PBE the cryptographic keys will be constructed from sets of attributes or predicates. These cryptographic keys could be represented using the models for attributes and access policies that are mentioned in many of the access control patterns. However, the problem is in deciding which representation to use when modelling either CP-PBE and KP-PBE schemes.

D. Token Management

A separate yet related issue to the representation of cryptographic keys is their management. Several patterns exist that look at cryptographic key management [27], [28]. These patterns provide solutions to problems found when providing a Public Key Infrastructure including key generation, key exchange, key revocation, and key storage. However, these patterns were designed using traditional public key encryption schemes and mechanisms. They are not suitable for direct use with PBE schemes. In this area of the pattern language, patterns need to be constructed that provide public key infrastructure solutions but use PBE schemes instead of traditional public key schemes.

E. System Administration

Although Section VI-D discussed more technical aspects of this administration, the managerial aspects also need to be modelled. Within PBE schemes the cryptographic keys are not *numbers*. Keys are derived from attributes and policies both of which posses some meaning. The keys created will be used to represent existing security policies, and describe subjects, resources, and the environment in which these policies are used. Patterns need to be constructed that take into account the access control administration and the cryptographic key generation.

VII. Evaluating Security Patterns

The security pattern landscape has been analysed before [19]–[21]. However, these analyses were not all aimed at the evaluation of individual patterns. There is a need for further investigation into the evaluation of individual security patterns. Below the existing research is described and briefly critiqued.

Halkidis et al. [19] performed a qualitative analysis of several security patterns according to: a) how well the pattern adheres to ten guiding principles for building secure software [29]; b) how well the pattern deters the software developer from building an insecure system; and c) how the pattern responds

to different types of attack. Similar work was also performed by Konrad et al. [30]. The use of qualitative evaluation criteria as used by Halkidis et al. is inherently problematic. Halkidis et al. [19, Section 4] mention that some of the criteria specified can only be used to assess the patterns implementation and not the pattern itself. For a qualitative analysis of patterns, criteria assessing the patterns and not implementation need to be specified.

Heyman et al. [21] used a quantitative approach to the evaluation of the security pattern landscape. During the evaluation Heyman et al. evaluated patterns according to three criteria. First was the pattern's *appropriateness* that is, was the pattern specified suitable for use when designing systems. The second criterion was towards the pattern's *quality of documentation*. This criterion looked at how well the problem, solution, and associated forces were described within the pattern. The final criterion investigated the distribution of security patterns over a list of security requirements taken from Firesmith [31]. Although, the efforts of Heyman et al. were primarily aimed at assessing the pattern landscape rather than individual patterns, the first two criterion, appropriateness and quality of documentation, are suitable criteria for evaluating individual patterns.

Finally, Yoshika et al. [20] provided a discussion towards the use of security patterns for security engineering. Yoshika et al. discussed patterns in terms of their: ease of use, effectiveness, and sufficiency. A pattern's *ease of use* is dependent upon the quality of documentation and the pattern's appropriateness cf. the first and second criteria from Heyman et al. [21]. The *effectiveness* of a pattern can be established using security metrics. Measurements can be taken before, and post pattern adoption. The final discussion point was towards the *sufficiency* of security patterns for security engineering. As with Heyman et al. the criteria were developed without individual pattern evaluation in mind. Yoshika et al. were primarily discussing the use of security patterns in relation to security engineering. Only the first two criterion from Yoshika et al. [20] are suitable for evaluating individual patterns.

VIII. Related Work

The use of PBE schemes as part of a crypto-system is not new. As mentioned in the introduction several systems [2]–[5] have been developed that used PBE within their operation. These systems looked at specific uses of PBE, and these deployments of PBE are analogous to the deployment patterns mentioned in Section IV. In Bobba et al. [2], the authors provide an attribute-messaging service, corresponding to the PBE-within-a-Service deployment pattern. The system developed in Zhao et al. [3] can be seen as an implementation of the Database Access pattern. Pirretti et al. [4] provide systems that can be seen as uses of the PBE-within-a-Service pattern. Finally, Akinyele et al. [5] present a system that uses a combination of the PBE-within-a-Service, and Database Access patterns.

Describing cryptography and key management as security patterns has been addressed before [22], [27], [28], [32]. Section VI discused how the patterns presented in References [22], [27], [28] were defined using traditional public key encryption schemes. They do not take into account encrypted access control for groups; they were designed before the advent of PBE. Finally, Cuevas et al. [32] provides patterns for encrypted access control within *Sensor Networks*. Although, these patterns describe encrypted access control for groups they have been specified for use within the application domain of sensor networks. The pattern language proposed will be application domain agnostic.

IX. Conclusion

This paper proposes the design of a pattern language for PBE crypto-systems. PBE is a novel family of public key encryption systems that facilitates expressive and fine-grained encrypted access control. Although, PBE schemes have been employed to provide encrypted access control the general deployment of such schemes as part of a crypto-system has yet to be formalised. A pattern-based approach provides a separation of concerns between the design of a system, and its implementation by developers. The proposed pattern language would be used as a tool by system designers when looking to include encrypted access control within their system.

Current focus has been towards developing the pattern language itself, and investigating further the requirements for each of the component areas required for modelling both CP-PBE and KP-PBE crypto-systems. The most recent efforts have been towards modelling the Group Information Secrecy patterns. However, there is still much work left. Futurework, will be two-fold. First the remaining areas of the pattern language will be developed. This will include providing patterns for each of the five component areas, and higher layers within the pattern language. Development of the pattern language will also include investigating more concrete patterns that can be used during the implementation stage of software development to realise these crypto-systems. Secondly, an evaluation of the constructed pattern language will take place, this will also require identifying suitable evaluation criteria.

Acknowledgement

This research has been supported through an EPSRC Doctoral Training Grant. This grant provides funding for 3.5 years of PhD research.

References

[1] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," *Advances in Cryptology — EUROCRYPT 2008*, pp. 146–162, 2008. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-78967-3_9

[2] R. Bobba, O. Fatemieh, F. Khan, A. Khan, C. A. Gunter, H. Khurana, and M. Prabhakaran, "Attribute-based messaging: Access control and confidentiality," *ACM Trans. Inf. Syst. Secur.*, vol. 13, no. 4, 2010.

[3] F. Zhao, T. Nishide, and K. Sakurai, "Realizing fine-grained and flexible access control to outsourced data with attribute-based cryptosystems," in *Information Security Practice and Experience*, ser. Lecture Notes in Computer Science, F. Bao and J. Weng, Eds. Springer Berlin / Heidelberg, 2011, vol. 6672, pp. 83–97. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-21031-0_7

[4] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," in *Conference on Computer and Communications Security: Proceedings of the 13th ACM conference on Computer and communications security*, 2006.

[5] J. A. Akinyele, C. U. Lehmann, M. D. Green, M. W. Pagano, Z. N. J. Peterson, and A. D. Rubin, "Self-protecting electronic medical records using attribute-based encryption," 2010, version 20101118:220821. [Online]. Available: http://eprint.iacr.org/2010/565

[6] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security patterns: integrating security and systems engineering*, ser. Wiley series in software design patterns. John Wiley & Sons, 2006.

[7] V. Goyal, A. Sahai, O. Pandey, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," 2006, Conference proceedings (article).

[8] D. Boneh, A. Sahai, and B. Waters, "Functional Encryption: Definitions and Challenges," 2010. [Online]. Available: http://eprint.iacr.org/2010/543

[9] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption." IEEE Computer Society, 2007, Conference proceedings (article), pp. 321–334.

[10] J. de Muijnck-Hughes, "Data protection in the cloud," Master's thesis, Radboud Universiteit Nijmegen, March 2011.

[11] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[12] F. Buschmann, K. Henney, and D. Schmidt, *Pattern-oriented software architecture: On patterns and pattern languages*, ser. Wiley series in software design patterns. John Wiley & Sons, 2007.

[13] E. B. Fernández, N. Yoshioka, H. W. J. Jürjens, M. V. Hilst, and G. Pernul, *Using Security Patterns to Develop Secure Systems*. IGI Global, 2011, ch. 2, pp. 16–31.

[14] N. Delessy, E. Fernandez, and M. Larrondo-Petrie, "A pattern language for identity management," in *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*, march 2007, p. 31.

[15] A. Cuevas, P. El Khoury, L. Gomez, A. Laube, and A. Sorniotti, "A security pattern for untraceable secret handshakes," in *Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on*, june 2009, pp. 8–14.

[16] A. Cuevas, A. Laube, A. Sorniotti, P. E. Khoury, and L. Gomez, "Security patterns for untraceable secret handshakes with optional revocation," *International Journal On Advances in Security*, vol. 3, no. 1&2, pp. 68–79, sept 2010. [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=sec_v3_n12_2010_6

[17] J. Hogg, D. Smith, F. Chong, D. Taylor, L. Wall, and P. Slater, *Web Service Security: Scenarios, Patterns, and Implementation Guidance for Web Services Enhancements (WSE) 3.0*, N. Delgado, Ed. Microsoft, 2005, outdated in relation to Microsoft technologies depicted. [Online]. Available: http://msdn.microsoft.com/en-us/library/aa480545.aspx

[18] M. Bunke, R. Koschke, and K. Sohr, "Application-domain classification for security patterns," in *PATTERNS 2011, The Third International Conferences on Pervasive Patterns and Applications*. ThinkMind, 2011, pp. 138–143.

[19] S. Halkidis, A. Chatzigeorgiou, and G. Stephanides, "A qualitative evaluation of security patterns," in *Information and Communications Security*, ser. Lecture Notes in Computer Science, J. Lopez, S. Qing, and E. Okamoto, Eds. Springer Berlin / Heidelberg, 2004, vol. 3269, pp. 251–259. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30191-2_11

[20] N. Yoshioka, H. Washizaki, and K. Maruyama, "A survey on security patterns," *Progress in Informatics*, no. 5, pp. 33–47, 2008.

[21] T. Heyman, K. Yskout, R. Scandariato, and W. Joosen, "An analysis of the security patterns landscape," in *Proceedings of the Third International Workshop on Software Engineering for Secure Systems*, ser. SESS '07. Washington, DC, USA: IEEE Computer Society, 2007, p. 3. [Online]. Available: http://dx.doi.org/10.1109/SESS.2007.4

[22] A. M. Braga, C. M. F. Rubira, and R. Dahab, "Tropyc: A pattern language for cryptographic software," in *Pattern Languages of Programs PLoP 1998*, 1998. [Online]. Available: http://www.dcc.unicamp.br/ic-tr-ftp/1999/99-03.ps.gz

[23] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, "Security patterns repository version 1.0," 2003, online [Accessed 2012-02-27]. [Online]. Available: http://scrypt.net/~celer/securitypatterns/repository.pdf

[24] E. B. Fernandez and G. Pernul, "Patterns for session-based access control," in *Proceedings of the 2006 conference on Pattern languages of programs*, ser. PLoP '06. New York, NY, USA: ACM, 2006, pp. 8:1–8:10.

[25] N. Delessy, E. B. Fernandez, M. M. Larrondo-Petrie, and J. Wu, "Patterns for access control in distributed systems," in *Proceedings of the 14th Conference on Pattern Languages of Programs*, ser. PLOP '07. New York, NY, USA: ACM, 2007, pp. 3:1–3:11.

[26] T. Priebe, E. Fernandez, J. Mehlau, and G. Pernul, "A pattern system for access control," in *Research Directions in Data and Applications Security XVIII*, ser. IFIP International Federation for Information Processing, C. Farkas and P. Samarati, Eds. Springer Boston, 2004, vol. 144, pp. 235–249.

[27] S. Lehtonen and J. Pärssinen, "A pattern language for key management," in *Pattern Languages of Programs PLoP 2001*, 2001.

[28] ——, "Pattern language for cryptographic key management," in *Proceedings of the 7th European Conference on Pattern Languages of Programs (EuroPlop'2002)*, 2002.

[29] J. Viega and G. Mcgraw, *Building Secure Software: How to Avoid Security Problems the Right Way (Addison-Wesley Professional Computing Series)*. Addison-Wesley Professional, Oct. 2001.

[30] B. H. Cheng, S. Konrad, L. A. Campbell, and R. Wassermann, "Using security patterns to model and analyze security requirements," *Security*, 2003. [Online]. Available: http://www.cse.msu.edu/~konradsa/Publications/rhas03.pdf

[31] D. G. Firesmith, "Engineering security requirements," *Journal of Object Technology*, vol. 3, no. 1, January 2004. [Online]. Available: http://www.jot.fm/issues/issue_2004_01/column6

[32] A. Cuevas, P. E. Khoury, L. Gomez, and A. Laube, "Security patterns for capturing encryption-based access control to sensor data," *Emerging Security Information, Systems, and Technologies, The International Conference on*, vol. 0, pp. 62–67, 2008.