Evaluating the implications of attack and security patterns with premortems.

FAILY, S., PARKIN, S. and LYLE, J.

2014

Evaluating the Implications of Attack and Security Patterns with Premortems

Shamal Faily, Simon Parkin, and John Lyle

Abstract Security patterns are a useful way of describing, packaging and applying security knowledge which might otherwise be unavailable. However, because patterns represent partial knowledge of a problem and solution space, there is little certainty that addressing the consequences of one problem won't introduce or exacerbate another. Rather than using patterns exclusively to explore possible solutions to security problems, we can use them to better understand the security problem space. To this end, we present a framework for evaluating the implications of security and attack patterns using *premortems*: scenarios describing a failed system that invites reasons for its failure. We illustrate our approach using an example from the EU FP 7 *webinos* project.

1 Contextualising Patterns for Security Design

Because security knowledge isn't readily available in design situations, there is considerable value in codifying and packaging it. Given both the adversarial nature of security, and the dangers of over or underestimation of security issues when this nature is misunderstood, it also seems useful to package knowledge about attacks as patterns. From a practitioner perspective, it seems surprising that, despite an abundance of examples of how security knowledge can be codified as patterns, e.g. [1], and the claim that building attack patterns is evidence of organisational security maturity [2], there is a dearth of work describing the application of attack patterns in security design.

Shamal Faily
Bournemouth University, Poole UK BH12 5BB, e-mail: sfaily@bournemouth.ac.uk

Simon Parkin
University College London, London UK WC1E 6BT, e-mail: s.parkin@ucl.ac.uk

John Lyle
University of Oxford, Oxford UK OX3 0NH, e-mail: john.lyle@cs.ox.ac.uk

Characterising attack and misuse patterns, e.g. [3], has been helpful in illustrating how patterns can tackle specific problems, but patterns need to be contextualised to be useful. One way of contextualising attack patterns involves better understanding the motives and capabilities of an attacker. Steps towards this goal are being made via profiling techniques [4], and the reuse of open-source intelligence for building attacker personas [5]. However, even when contextualised, such representations remain only partial representation of an attacker's knowledge. Although these provide inspiration for undirected ideation activities, more relevant qualitative data is needed to augment profiles or patterns to understand what specific attacks they might carry out in specific contexts. If such data was available then attacks would be self-evident, thereby eliminating the need for attack patterns.

2 Patterns as an exploratory tool

We may never have the assurances that we would like about a pattern's efficacy; while a pattern may be one possible solution to a problem, we can never be completely sure that this solution itself doesn't introduce further complications we have yet to identify. This is a symptom of the lack of clarity about what it means to secure a system. This, in turn, makes it difficult to devise tests for proving a system is secure, and a grasp of all possible solutions for satisfying a specified security problem [6]. Patterns have the potential to provide this clarity because not only do they help solve security problems, they can also help understand the problem space security is concerned with as well. When we apply patterns, we also make value judgements about how attacker patterns might exploit systems, or how different security patterns might be a satisficing security design solution. These value judgements help us understand and delimit the solution space. Therefore, not only are patterns effective at structuring knowledge, they also explore the consequences of their application. If we can better understand these consequences, we can also better understand what is important to us when thinking about system security.

Interestingly, the value associated with applying patterns to delimit the problem space is obtained whether or not they successfully address the problem we had in mind. While it seems paradoxical that we would apply a security pattern knowing that it will ultimately fail, such an approach is analogous to a *premortem*. In business scenario planning, these operate on the assumption that a solution has failed and, rather than reflecting on what might have gone wrong, designers instead generate plausible reasons for explaining the solution's failure [7]. Although the known structure, motivation, and consequences of security patterns provide some insight into the causes of such a failure, when combined with attack patterns, they allow us to reflect on the motivations of a perceived attacker, and how his capabilities and motivations ultimately led to an exploit. More interestingly, if the mapping between patterns is not clear, the lack of data also provides clues about what additional evidence is needed before the "cause of death" can be established.

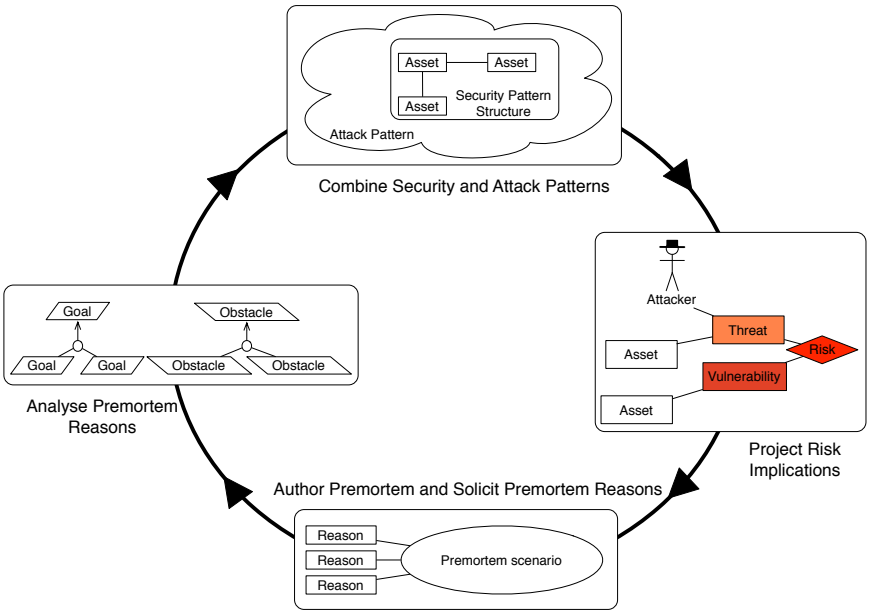

Fig. 1 A Framework for Evaluating Patterns with Premortems

3 Approach

To make the most of the exploratory power of both patterns and premortems, we have devised an approach to evaluate the implications of applying security and attack patterns. This entails combining security and attack patterns, and statically evaluating the risk posed as a result. The risk impact is described using a premortem scenario, and reasons for the failure described are solicited from domain experts or stakeholders. Using KAOS goal and obstacle modelling techniques [8], the underlying causes for these reasons are elicited as implicit requirements that hold when the security pattern is applied. The obstacles and goals elicited are then used to refine the select attack and security patterns respectively.

This approach, which we illustrate in Figure 1, relies on the open-source CAIRIS security requirements management tool[9, 10]. It also relies on goal and risk modelling techniques developed as part of the IRIS framework [11].

This approach is described in more detail in the following sub-sections.

3.1 Combine Security and Attack Patterns

The first step involves selecting a security and attack pattern to form the basis of evaluation. The security pattern represents some aspect of a system architecture

being evaluated, while the attack pattern represents elements of an insecure context of use where the security pattern is being applied.

Both the attack and security pattern are described by XML documents, which are imported into CAIRIS. Compared to the security patterns described by [1], the patterns imported are architectural in nature. They include definitions for components, connectors, and interfaces, as well as security values such as interface privileges, interface and component access rights, and protocols associated with connectors. The pattern structure also models assets associated with components and connectors, and requirements that concern these assets.

The attack pattern structure is based on the classic "Gang of Four" design pattern template [12], but is also aligned with the IRIS meta-model used by CAIRIS [13]. As such, each attack pattern corresponds to a single risk in IRIS [13].

The pattern structures for both are described in more detail in [14].

3.2 Project Risk Implications

Based on the assets, threats and vulnerabilities associated with the security and attack patterns, a risk analysis is performed; this provides an initial evaluation of the risk implications of the pattern combination.

This analysis is automatically carried out by CAIRIS based on the imported security and attack patterns. The results are visualised using CAIRIS' risk modelling notation; this is described in more detail by [15].

3.3 Author Premortem and Solict Premortem Responses

Using inspiration from both the patterns and the risk analysis, a premortem scenario is written to characterise a system failure resulting from this application of both patterns. Reasons for these failures can be elicited from a variety of different people. These range from direct stakeholders, such as customers or project team members, to indirect stakeholders, such as domain experts or members of the public.

Because premortems are not conceptually supported by CAIRIS, this scenario is modelled as a misuse case [16] and associated with the risk resulting from the pattern combination.

3.4 Analyse Premortem Reasons

Using KAOS obstacle models, each premortem reason is modelled as a set of obstacles. Obstacles are abstracted to identify problems giving rise to these reasons; they are also refined to identify conditions that need to be present for the reasons to hold.

Where security pattern requirements mitigate these obstacles, obstacle resolution links are added to the obstacle model.

To capture the insights resulting from this evaluation, the obstacle model is incorporated into the structure of the CAIRIS attack pattern. To facilitate this, the Data Type Definition for attack pattern XML documents includes an implementation element. This element allows an attack to be described textually and visually using an obstacle model.

The implicit requirements that mitigate these obstacles are also incorporated into the security pattern. These requirements are represented as goals trees, which are associated with individual components in the pattern.

4 Example

We illustrate our approach by describing how it was used to evaluate aspects of the security architecture of *webinos*. *webinos* is a software infrastructure for running web applications across smartphones, PCs, home media centres, and in-car devices [17]. *webinos*' software architecture includes policy management components for facilitating cross-device access control [18].

The process used to carry out an architectural risk analysis process was described in [14]. Our approach complements that process by providing a framework for moving from the identification of general flaws and knowledge about known attacks, to the discovery of risks resulting from ambiguity and inconsistency in a system design.

4.1 Combine Security and Attack Patterns

We wanted to consider how resistant the policy management architecture might be to attempts to *footprint*, i.e. attempts to gather potentially exploitable information about the system.

The security pattern is characterised using the *Context Policy Management* architectural pattern [19]. This pattern realises the policy management requirements specified by [18].

The initial attack pattern illustrated in Figure 2 was constructed by searching for keywords associated with footprinting activities using open-source intelligence. In this example, attacks and exploits were found by searching the Common Attack Pattern Enumeration and Classification (CAPEC) and Common Weakness Enumeration (CWE) repositories [20, 21].

Name	Test footprinting
Likelihood	Occasional
Severity	Critical
Intent	Glean an understanding of what resources are available on a device by eavesdropping on requests.
Motivational Goal	Accountability (High)
Motivation	Ethan looks for test code which provides unauthorised resource access.
Applicable Environment	Complete
Structure	
Attack	Locate and Exploit Test APIs
Exploit	Allocation of Resources without Limits or Throttling
Participant	Ethan
Motives	Data theft
Responsibilities	Technology (Medium), Software (Medium), Knowledge/Methods (Medium)
Collaboration	
Target	Application Data
Exploit	Access Request

Fig. 2 Initial attack pattern structure

4.2 Project Risk Implications

When the patterns were imported into CAIRIS, a risk model was automatically generated and a "Test footprinting" risk added to characterise the attack pattern.

When this risk model, which is illustrated in Figure 3, was automatically generated, the risk score seemed surprisingly low based on colour of the risk diamond. In the risk model, the shade of red is used to indicate its seriousness; the darker the shade, the more serious the risk is. Based on the initial lightness of the shade, the risk was not considered significant. On further investigation, we discovered that this was because no security properties were associated with the exploited Application Data asset, despite an attack that compromised its accountability. This minimised the impact of the threat, thereby lowering the overall risk rating. However, initially contextualising the attack pattern with the *Ethan* attacker persona [22] indicated that both integrity and accountability of this data needed to be maintained. Adding these

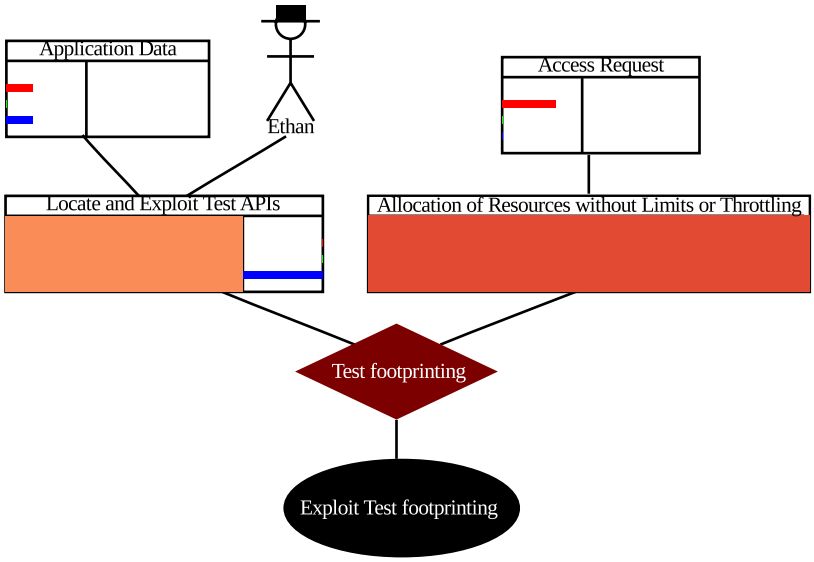

Fig. 3 CAIRIS Risk Analysis model of security and attack pattern elements

security properties to the Application Data asset subsequently lead to the criticality of risk being increased.

4.3 Author Premortem and Solict Premortem Responses

Inspired by the risk model, we developed the premortem scenario below to characterise the implications of an access control failure.

The Register picked up a story based on blog reports by irate mobile users complaining about increased email spam since they started using webinos applications; this spam is sufficiently targeted that it evades spam filters. This has led to lots of irate twitter posts appearing on the twitter feed on the webinos home page, especially from angry developers who users blame for this traffic. As the bad press grew and open-source advocates began to criticise the project, major partners began to leave the project, and EC funding was cut; this lead to the project being halted due to lack of money.

This premortem scenario was emailed to all members of the *webinos* project team, who were asked to reply to the email with their reasons for the described failure. Although the most active members of the project team were web app developers, the team also included participants with backgrounds in handset and car manufacturing, and mobile market analysis.

4.4 Analyse Premortem Reasons

Several possible reasons for this failure were proposed. These ranged from the re-selling of private application data, through to attackers actively footprinting the network infrastructure hosting the cloud-based hubs for *webinos* personal zones. For reasons of brevity, we will consider only one of these: one team member suggested that *webinos*' access control policies might be open to ambiguity, and that this could lead to information being shared against the wishes of end users.

Using the ambiguity analysis techniques described in [14], it was possible to derive the obstacle model illustrated in Figure 4. The Test Footprinting attack pattern was revised to incorporate this obstacle model as rationale for the attack [23]. Additionally, insights about these reasons lead to revisions to the access right, protocol, and privilege values associated with the Context Policy Management architectural pattern.

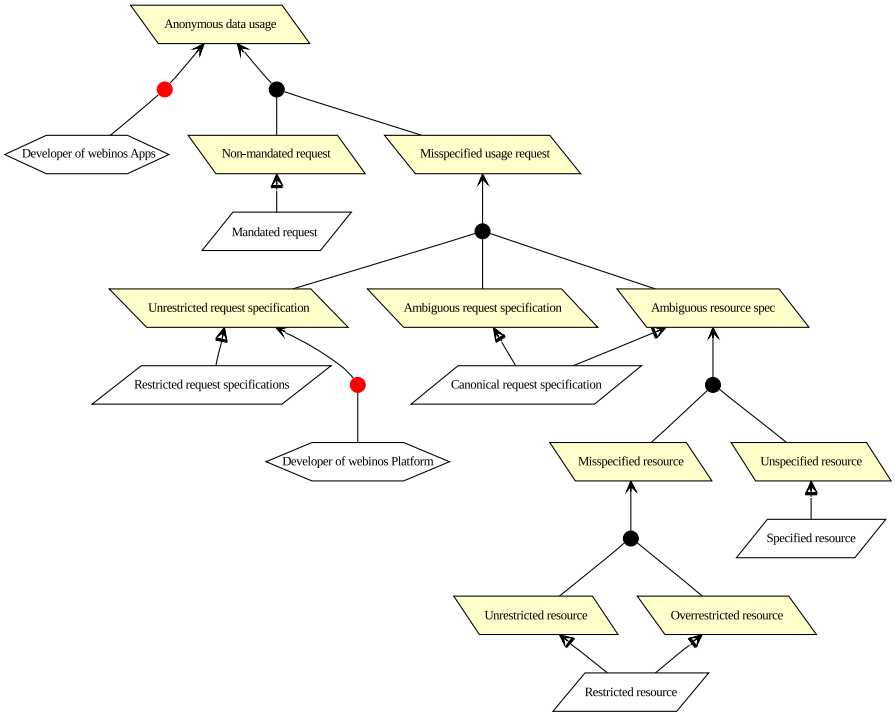

Fig. 4 Obstacle model associated with premortem reason

5 Implications

Devising and applying this premortems-based approach to pattern evaluation has broader implications on the design and evaluation of security. We describe some of these in the following sub-sections.

5.1 Analysing incomplete risk models

The Test Footprinting attack pattern sketched in Figure 2 made assumptions about both vulnerable assets, and assets used as part of a broader exploit. Because the IRIS meta-model assumes these assets are known when specifying risks, these assumptions needed to be made early. However, while indications about vulnerable assets may be known at an early stage, it's unlikely that details about exploited assets will be known. If inappropriate assets, or asset security properties are elicited at this early stage, the visual risk models can help identify this. However, inappropriate assets or asset values may also inspire inappropriate premortems; these are less easy to identity, and may lead to the exploration of inappropriate parts of the problem space. While there is no easy remedy to this problem, it does suggest that risk models may benefit from being less specific at an early stage; this has implications on how risks might be appropriately analysed and visualised using incomplete information.

5.2 Measuring quality concerns

As well as evaluating the implications of attack and security patterns, this approach also helps us measure quality concerns. Responses given by project team members may be complicated, simple, pessimistic, or optimistic. In each case, however, they say something about how confident team members are about either the system or the context it is situated in. This suggests that premortem responses are a useful tool for managers to keep in touch with the security reality of a project.

We can also use this approach to understand why security systems may not satisfy quality requirements we expect to hold. For example, we may expect a combination of security patterns to fail fast and predictably for different attack patterns. If we write a premortem describing a system that fails slowly and unpredictably then responses might help us to identify what we need to do to fix these problems.

5.3 Engagement with hard to reach groups

Premortems allow organisational stakeholders with security interests, but a non-IT background, to engage in security design. While time-constrained stakeholders,

such as senior managers, may find it difficult to take time out to engage in design activities, they might it easier to provide short, thought-provoking feedback to scenarios that capture their imagination. When these responses are recorded, they can formally acknowledge their contribution to the design process. However, because premortems are successful because contributors don't feel any responsibility for their contribution, acknowledging these responses may lead to less than honest responses.

5.4 Promoting Open Innovation

Although our approach relied on a predefined list of stakeholders for premortem responses, we could also use *Open Innovation* to collect responses from a broader base of external respondents. Open Innovation is a paradigm where both ideas and paths to market are generated from both within and outside an organisation [24]. This can be made possible by using premortems to describe contextual information; this is collected while combining patterns and projecting risks, particularly when risks are rated at a lower than expected value. Using premortems, this contextual information could be shared without disclosing sensitive information about the nature of the assets being exploited, potential attacks and attackers.

6 Conclusion

In this chapter we described an approach where premortems were used to better understand the problem and solution space associated with attack and security patterns. We illustrated this approach using an example where this exploration lead to more refined pattern descriptions.

We are currently exploring ways that CAIRIS can be updated to better address the lessons learned applying this approach. For example, we are currently investigating different ways of making the IRIS risk meta-model more permissive given the issues raised in Section 5.1. We are also considering more effective ways of supporting the premortem technique within CAIRIS and the broader IRIS framework. These include devising more effective ways of supporting the elicitation of premortem scenarios, and soliciting reason responses from appropriate stakeholders.

In future work, we will explore how this technique might be used by security testers, in particular ethical hackers, to support a deep-dive analysis of a system. In particular, we will consider how this approach provides a framework for taking fresh thinking that ethical hackers can provide, and aligning this thinking with both exploiting and mitigating system models.

Acknowledgements The work described in this chapter was funded by the EU FP7 *webinos* project (FP7-ICT-2009-05 Objective 1.2).

References

1. Schumacher M, Fernandez E, Hybertson D, Buschmann F. Security Patterns: Integrating Security and Systems Engineering. John Wiley & Sons; 2005.
2. McGraw G, Chess B, Migues S. Building Security in Maturity Model; 2011.
3. Fernandez EB, Yoshioka N, Washizaki H, Jürjens J, VanHilst M, Pernul G. Using Security Patterns to Develop Secure Systems. In: Mouratidis H, editor. Software Engineering for Secure Systems: Industrial and Research Perspectives. IGI Global; 2011. p. 16–31.
4. Chiesa R, Ducci S, Ciappi S. Profiling Hackers: The Science of Criminal Profiling as Applied to the World of Hacking. 1st ed. Boston, MA, USA: Auerbach Publications; 2008.
5. Atzeni A, Cameroni C, Faily S, Lyle J, Flechais I. Here's Johnny: A Methodology for Developing Attacker Personas. In: Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security. ARES '11. Washington, DC, USA: IEEE Computer Society; 2011. p. 722–727. Available from: http://dx.doi.org/10.1109/ARES.2011.115.
6. Faily S, Flechais I. To boldly go where invention isn't secure: applying security entrepreneurship to secure systems design. In: Proceedings of the 2010 workshop on New security paradigms. NSPW '10. New York, NY, USA: ACM; 2010. p. 73–84. Available from: http://doi.acm.org/10.1145/1900546.1900557.
7. Klein G. Performing a Project Premortem. Harvard Business Review. 2007;85(9):18–19.
8. van Lamsweerde A. Requirements Engineering: from system goals to UML models to software specifications. John Wiley & Sons; 2009.
9. Faily S, Fléchais I; IGI Global. Towards tool-support for Usable Secure Requirements Engineering with CAIRIS. International Journal of Secure Software Engineering. 2010 July-September;1(3):56–70.
10. Faily S. CAIRIS web site; 2013. http://github.com/failys/CAIRIS.
11. Faily S, Fléchais I. Eliciting Policy Requirements for Critical National Infrastructure using the IRIS Framework. International Journal of Secure Software Engineering. 2011;2(4):114–119.
12. Gamma E, Helm R, Johnson R, Vlissides J. Design patterns: elements of reusable object-oriented software. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.; 1995.
13. Faily S, Fléchais I. A meta-model for usable secure requirements engineering. In: Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems. SESS '10. New York, NY, USA: ACM; 2010. p. 29–35. Available from: http://doi.acm.org/10.1145/1809100.1809105.
14. Faily S, Lyle J, Namiluko C, Atzeni A, Cameroni C. Model-driven architectural risk analysis using architectural and contextualised attack patterns. In: Proceedings of the Workshop on Model-Driven Security. MDsec '12. New York, NY, USA: ACM; 2012. p. 3:1–3:6. Available from: http://doi.acm.org/10.1145/2422498.2422501.
15. Faily S, Flechais I. Analysing and Visualising Security and Usability in IRIS. 2012 Seventh International Conference on Availability, Reliability and Security. 2010;0:543–548.
16. Sindre G, Opdahl AL. Eliciting security requirements with misuse cases. Requir Eng. 2005 Jan;10(1):34–44. Available from: http://dx.doi.org/10.1007/s00766-004-0194-4.
17. Fuhrhop C, Lyle J, Faily S. The webinos project. In: Proceedings of the 21st international conference companion on World Wide Web. WWW '12 Companion. New York, NY, USA: ACM; 2012. p. 259–262. Available from: http://doi.acm.org/10.1145/2187980.2188024.
18. Lyle J, Monteleone S, Faily S, Patti D, Ricciato F. Cross-Platform Access Control for Mobile Web Applications. Policies for Distributed Systems and Networks, IEEE International Workshop on. 2012;0:37–44.
19. webinos Consortium. Context Policy Management Architectural Pattern; July. https://github.com/webinos/webinos-design-data/blob/master/architecture/architecturalpatterns/ContextPolicyManagement.xml.

20. The MITRE Corporation. Common Attack Pattern Enumeration and Classification (CAPEC) web site; 2012. http://capec.mitre.org.
21. The MITRE Corporation. Common Weakness Enumeration (CWE) web site; 2012. http://cwe.mitre.org.
22. webinos Consortium. Ethan Attacker Persona; 2011. https://github.com/webinos/webinos-design-data/blob/master/personas/ethan.xml.
23. webinos Consortium. Test Footprinting Attack Pattern; July. https://github.com/webinos/webinos-design-data/blob/master/architecture/attackpatterns/TestFootprinting.xml.
24. Chesbrough HW. Open innovation: the new imperative for creating and profiting from technology. Harvard Business School Press; 2003.