Engaging Stakeholders in Security Design: An Assumption-Driven Approach

S. Faily

Software Systems Research Centre, Bournemouth University
e-mail : sfaily@bournemouth.ac.uk

Abstract

System stakeholders fail to engage with security until comparatively late in the design and development process. User Experience artefacts like personas and scenarios create this engagement, but creating and contextualising them is difficult without real-world, empirical data; such data cannot be easily elicited from disengaged stakeholders. This paper presents an approach for engaging stakeholders in the elicitation and specification of security requirements at a late-stage of a system's design; this approach relies on assumption-based personas and scenarios, which are aligned with security and requirements analysis activities. We demonstrate this approach by describing how it was used to elicit security requirements for a medical research portal.

Keywords

Personas, Scenarios, Requirements, Risks, Context

1. Introduction

When building systems, it is generally agreed that its stakeholders should be engaged in security as early in the design process as possible. However, all too often, security is considered an after-thought, and security requirements are not properly considered until comparatively late in a project's lifecycle.

Engaging the stakeholders is difficult. The right stakeholders may be heavily in demand, and motivated by innovation rather than security. For example, a software developer may be required to dedicate significant time and resources to understanding the complexity of a problem domain, leaving themselves little time for applying standard security design techniques. Such stakeholders may also find security a distant topic, with media reports on security threats and privacy invasions as somehow irrelevant to a system they are trying to build.

One way of engaging the security unengaged is rely not only on evocation, but also people's natural bias towards personified, rather than anonymous, risk (Schneier, 2012). Software developers may gloss over stories about the loss or public disclosure of patient medical data, but highlighting *their* contribution towards such losses may draw their attention. User Experience (UX) artefacts can evoke by contextualising or personifying these losses, but building them requires real-world empirical data; this can only be collected when stakeholders are engaged, thereby leading to a `chicken-and-egg' situation.

Recent work (Dray, 2014) has highlighted the power of assumptions when engaging developers. Using UX research to challenge assumptions helps developers recognise why such issues need to be addressed, and focus their curiosity towards addressing them. To explore the power of challenging such assumptions, this paper presents an approach for eliciting and specifying security requirements using assumption-based personas, scenarios, and risks to engage system developers to think more about security for a medical research portal, particularly how the portal might be misused. In Section 2, we briefly describe the related work upon which this approach is based, before presenting the approach in Section 3. We describe a case study where this approach was applied in Section 4, before concluding with some implications for security design in Section 5.

2. Related Work

2.1. Personifying Security Expectations

The *personas* technique is a popular UX approach for personifying users to understand their goals and needs (Cooper et al, 2007). Personas are models that provide a specification of archetypical user behaviour. By designing software to satisfy the expectations of these personas, software developers need not rely on their own assumptions about users, which may be unwarranted. In recent years, personas have also been used to support secure system design interventions. For example, (Faily and Fléchais, 2010) found that personas not only provided empathy about the security challenges of hard-to-reach user groups, but were useful for eliciting unforeseen user characteristics if stakeholders felt a persona didn't match reality.

Personas are data-driven, and collecting the empirical data necessary to build them is difficult if stakeholders are not engaged enough to provide or facilitate access to such data. Given these difficulties, (Pruitt and Adlin, 2006) proposes the use of *assumption personas*; these are sketches that articulate assumptions about a user population. Once created, assumption personas allow stakeholders to see the value of personas, and how assumptions may colour their characteristics.

2.2. Contextualising Personas in Secure System Design

Personas build empathy, but their goals and expectations need to be put in context. For this reason, personas are often paired with scenarios; these centre around activities performed by users, rather than around the users themselves. For example, (Rosson and Carroll, 2002) used scenarios to describe how hypothetical stakeholders tackle current practice; these scenarios may be based on empirical data or assumptions. More recently, (Parkin et al, 2010) successfully engaged senior managers using low-fidelity prototypes of security management tools, and a collection of scenarios illustrating their use.

Together, personas and scenarios illustrate security problems, but understanding these problems is not enough to specify solutions that address them: we need to carry out a more formal security and requirements analysis. Personas and scenarios supplement these analyses by illustrating how risks are realised, and how specification decisions are operationalized. In doing so, the human implications of security design decisions in different contexts of use can be better perceived.

To illustrate how these different approaches can work together, the Integrating Requirements and Information Security (IRIS) Framework (Faily and Fléchais, 2010, 2011) was devised; the framework demonstrates how the elements constituting personas, scenarios, requirements, and risks might be aligned, and the application of security, usability, and requirements techniques can complement each other.

3. Approach

Using personas and scenarios, we have developed an approach for eliciting and specifying security requirements that engages stakeholders in security concerns. This approach not only captures information about how usability and security concerns impact requirements, it also accommodates a lack of end-user access, limited access to project stakeholders, and the need to make assumptions about users as transparent as possible during design.

The approach is tool-supported by the open-source Computer Aided Integration of Requirements and Information Security (CAIRIS) requirements management tool (Faily, 2013). CAIRIS was developed to support the IRIS Framework; it allows the capture of security, usability, and requirements data as the techniques are applied, guides the creation of personas, and automatically evaluates risks for different contexts of use (Faily and Lyle, 2013; Faily and Fléchais 2010b).

3.1. Assumption Persona Development

The first stage of the approach involves specifying the expectations held about a system's prospective user-community. Implicit assumptions in the available documentation are identified, and used to form the basis of assumption personas. Not only do these assumption personas clarify expectations about end-users, subsequent discussion around these confirm a useful scope of analysis for the subsequent stage.

For each role relevant to the scope of analysis, the available documentation is reviewed to elicit factoids for each role. These are used to establish persona characteristics and, based on these, assumption persona narratives. Once the assumption personas are developed, these are presented to the project team for review. Any issues raised by the team are used to revise the assumption personas or correct any misinterpretations held about the system. The process for building these personas is described in more detail in (Faily and Fléchais, 2010a).

3.2. Design Sessions

This stage entails holding small focus groups with project team members. Each session focuses on the use of scenarios, requirements, or risk analysis.

A scenario session involves modelling scenarios carried out by the assumption personas in their respective contexts. Like the personas, these scenarios are grounded in assumptions identified from project documentation, or from analysis undertaken during other design sessions. Some of these scenarios focus on *misusability*, by illustrating how unintentional misuse of the system might lead to security problems.

A requirements session involves using the KAOS goal-oriented requirements engineering approach (van Lamsweerde, 2009) to elicit and specify requirements needing to be satisfied in order for the scenarios to be realised. Requirements are modelled as goal trees and, in addition to being refined to sub-goals, goals may conflict with obstacles: conditions representing undesired behaviour and preventing an associated goal from being achieved (van Lamsweerde and Letier, 2000). Such obstacles may arise from intentional, as well as accidental, obstacles, thereby making it possible for them to model threats (van Lamsweerde, 2004).

Risk analysis sessions involve using AEGIS (Appropriate and Effective Guidance for Information Security): a participative design process (Fléchais et al, 2007). This entails the team members jointly modelling the system's assets in different contexts; these assets are modelled using UML class diagrams, where classes represent different assets. The assets are evaluated according to values held by the participants about them. Vulnerabilities, threats, and risks affecting these assets are elicited, before possible security controls mitigating these risks are selected. Although one of many risk analysis processes, AEGIS's diagrammatic notation is useful for engaging stakeholders about security, providing useful discussion about asset values, and eventually yielding relevant security requirements (Fléchais, 2005).

In all sessions, assumption personas are used as an authority for user expectations; these are modified if aspects of the analysis challenge their characteristics.

During the sessions, elicited requirements and security analysis elements are specified within CAIRIS, and the resulting models are discussed with the session participants. After the final session, each requirement is examined and assigned a responsible role. Following this, a specification document is generated and sent to project team members for their review.

4. Case Study

This approach was evaluated by using it to elicit security requirements for a portal that facilitated the sharing of medical study data. The study data consisted of long-running, longitudinal studies of people sharing some specific characteristic. By providing an accessible interface to such studies, the portal ensured that research data and meta-data could be re-used by researchers, thereby reducing the need for running unnecessary and expensive long-term studies.

Two particular user roles dominated the design of the portal. The first of these were academic researchers; these would use the portal to find datasets of interest. The project sponsors were keen to maximise take-up by the researcher community, and initiatives encouraging this would be looked upon kindly. The second class of users were data managers; these were responsible for curating data sets, which would be available via the portal. The perception held by the project team was that data managers were portal's key user community.

The portal design was dominated by two contexts of use. The *Research* context was concerned with researchers interacting with the portal as part of their day-to-day research. The *Study* context was concerned with data managers' interaction with the portal to curate their datasets, and managing requests for accessing them.

Although empirical data from representative stakeholders would have made an invaluable contribution to the usability and security analysis, there was no scope for collecting data from research end-users. Similarly, time constraints meant that data managers from the study exemplars would also not be available. Fortunately, the development team agreed to act as user proxies because of the time they had spent working with data managers from the exemplar studies, and their domain knowledge based on previous, related research. However, given that the development team had little direct experience of the researcher community, it was important that any assumptions that were made about both data managers and researchers needed to be as transparent as possible.

Unfortunately, limited access to stakeholders also extended to the development team. Despite the project team being small with only 4 developers, it faced several tight deadlines; this made opportunities to work with the project team severely limited. Consequently, it was essential to be parsimonious with regards to team member access, while at the same time ensuring that the intervention had an impact on the development of the portal.

4.1. Assumption Persona Development

When the study began, only two documents were available for eliciting assumptions: a requirements specification for the portal, and a technical annexe to the portal's contract signed by all project partners.

After a review of the documentation, three roles were evident: researchers, data managers, and gateway administrators. Based on these roles, the documentation was analysed to elicit assumptions. Assumptions were elicited about behaviour, which could be reasonably assumed if the documentation accurately represented the concerns of the particular stakeholder role. For example, a requirement indicating that first-line support to the portal would be provided between the hours of 9 am to

5.30pm might reasonably suggest that the authors believe researchers work only during commercial office hours.

Three skeleton assumption personas were created for each of these roles: Alex (an academic researcher), Brian (a data manager), and Colin (an administrator for the Data Gateway). For each persona's characteristic, a narrative was written commensurate with it. For example, based on persona characteristics summarised by *In no hurry* and *Looking to apply data-set once discovered*, the following narrative describing Alex's motivation was written:

Alex is looking to use a dataset as soon as he discovers it is suitable. He isn't in a particular hurry, so is prepared to wait for his registration to the Data Gateway and the respective data set to be approved.

The results of this initial analysis to date were presented to the project team. This presentation described the scope assumed for the analysis, provided an overview of the work carried out, and Alex, Brian, and Colin were presented. Each persona selected particular persona characteristics. The third characteristic was chosen as the most divisive, in order to stimulate lively discussion about the persona. At the end of this session, it was agreed that Colin's activities were not relevant to the scope of analysis, and this persona was dropped from the remainder of the intervention.

Despite the nature of the documentation used, it was possible to elicit a surprising amount of data about both the possible activities and attitudes of personas. Moreover, many persona characteristics were elicited during the design sessions. As such, the personas evolved throughout the design sessions, concurring with best practice in the use of personas, which suggests that personas should be fostered throughout a project’s lifecycle (Pruitt and Adlin, 2006).

Although identifying the basis for characteristics was straightforward, justifying them was more difficult. Prior to their initial validation, many of the characteristics were based exclusively on individual pieces of empirical data. As such, value judgements about the source data and context were directly reflected in these characteristics.

Although the initial workshop surfaced a number of these issues, it was usually not until the personas were directly written into scenarios in design sessions that certain invalid characteristics were identified. Applying the personas within a specific context did, however, help identify missing data about behaviour not identified during their creation and initial presentation to the team.

4.2. Design Sessions

Four design sessions were held over the course of a month. The first was a scenario session followed, a day later, by a requirements session. A risk analysis session took place the following week, followed by a final risk analysis session the week after.

Due to project deadlines, rather than having access to multiple developers per session, only a single developer was available. The same developer was consistently used for each session, and was available for email clarification when queries arose outside at other times. In addition to the allocated portal developer, a non-project domain expert participated in the second session. Although this participant was only partially knowledgeable in the on-going project, she was aware of the problem domain in general.

The approach taken during each session was more flexible than originally envisaged. In practice, multiple techniques were used when the situation deemed it useful. For example, KAOS was used in each of the first sessions when it was felt most appropriate. Similarly, elements of AEGIS were also used in the first two sessions to elicit assets, their relationships, and concerns arising from goals and scenarios. Switching from the use of one technique to another did not seem to hinder the thought processes of participants in the sessions.

During the sessions, the personas were progressively refined and embellished with further characteristics from the documentation as new insights were gleaned.

The amount of data elicited from the risk analysis sessions was comparatively small. This was mainly due to the resolution of many security and usability problems during the requirements and scenarios sessions. Another reason for the small number of explicit risks was a tendency by the project team to dismiss security issues deemed out of scope. On more than one occasion, assets identified as in scope, such as portal documentation about the use of some functionality, was de-scoped. This issue of passing responsibility for out-of-scope issues was also apparent from the threats and vulnerabilities highlighted in both contexts of use.

The issue of scope deferral was also contextual. Most of the risk analysis elements were elicited from the Research context; these were associated with assets deemed out of the project scope. The few risk analysis elements not concerning the Study environment were also marginalised. For example, of four risks elicited, only one - a Man-in-the-Middle attack - concerned the Study environment. Upon discussing resolutions to this, it was agreed that the portal relied on a secure channel between some of its components. Consequently, responsibility for mitigating more general Man-in-the-Middle attacks was delegated to the administration team responsible for one of these components.

Although the project team were reluctant to take a defence-in-depth approach to tackling security problems, security concerns were eventually identified. This was possible by focusing attention on goal obstruction within the Study environment; unlike the Research environment, this environment concerned concepts that *were* within the project scope. This allowed threats and vulnerabilities to be mitigated at the design level when considered in context with other portal requirements. This was especially useful because, besides the generic internet-facing threats and vulnerabilities, it was not entirely clear what the threat model facing the portal might be.

5. Conclusion

In this paper, we presented an approach where assumptions were used to ground an approach for engaging project team members when eliciting and specifying security requirements. The case study example demonstrated that, in lieu of research on a target system's users, assumptions afford the creation of personas that engage developers in security while they building a system. Moreover, these personas can be used in design sessions, which elicit or draw attention to security requirements without disrupting a project's on-going development activities. Based on the lessons learned applying this approach, three security design implications can be taken away.

First, engagement can follow by focusing on the indirect, rather than direct, consequences of security. One of the difficulties in completing this study arose from the lack of engagement with the project team. Although the project team appeared to be genuinely interested in the approach and the analysis being carried out, their time was too limited to properly integrate this analysis into the day-to-day running of the project. The case study showed that focusing on the impact of non-security design decisions is a more effective technique for engaging developers in security issues rather than relying on fear of more generic threats, particular when these threats may or may not be relevant (or perceived relevant) to the scope of analysis.

Second, as much as security should be considered at the outset of a project, we may need to develop design approaches for treating it at a comparatively late stage. This study reinforced the need for innovative thinking to ensure that important security issues were built into the system. This is an important finding as the individual techniques used were designed on the basis that they would be used early in the design process. In contrast, not only were they applied at a comparatively late stage; they were used in parallel with other design activities.

Finally, our findings lead us to conclude that when a security design process is devised, its techniques should scale to working with less than optimal input data. Moreover, the process should attempt to carry out as much analysis as reasonably possible carried out without disrupting other project activities.

6. References

Cooper, A., Reimann, R., and Cronin, D. (2007). About Face 3: The Essentials of Interaction Design. John Wiley & Sons.

Dray, S. M. (2014). Questioning Assumptions: UX Research That Really Mat- ters. interactions, 21(2):82–85.

Faily, S. (2013). CAIRIS web site. http://github.com/failys/CAIRIS.

Faily, S. and Fléchais, I. (2010). A Meta-Model for Usable Secure Requirements Engineering. In Proceedings of the 6th International Workshop on Software Engineering for Secure Systems, pages 126–135. IEEE Computer Society.

Faily, S. and Fléchais, I. (2010). Barry is not the weakest link: eliciting secure system requirements with personas. In Proceedings of the 24th BCS Interaction Specialist Group Conference, BCS '10, pages 124–132. British Computer Society.

Faily, S. and Fléchais, I. (2010a). The secret lives of assumptions: Developing and refining assumption personas for secure system design. In Proceedings of the 3rd Conference on Human-Centered Software Engineering, volume LNCS 6409, pages 111–118. Springer.

Faily, S. and Fléchais, I. (2010b). Towards tool-support for Usable Secure Requirements Engineering with CAIRIS. International Journal of Secure Software Engineering, 1(3):56–70.

Faily, S. and Fléchais, I. (2011). Eliciting Policy Requirements for Critical National Infrastructure using the IRIS Framework. International Journal of Secure Software Engineering, 2(4):114–119.

Faily, S. and Lyle, J. (2013). Guidelines for integrating personas into software engineering tools. In Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems, EICS '13, pages 69–74. ACM.

Fléchais, I. (2005). Designing Secure and Usable Systems. PhD thesis, University College London.

Fléchais, I., Mascolo, C., and Sasse, M. A. (2007). Integrating security and usability into the requirements and design process. International Journal of Electronic Security and Digital Forensics, 1(1):12–26.

Parkin, S., van Moorsel, A., Inglesant, P., and Sasse, M. A. (2010). A stealth approach to usable security: helping it security managers to identify work- able security solutions. In Proceedings of the 2010 workshop on New security paradigms, NSPW '10, pages 33–50. ACM.

Pruitt, J. and Adlin, T. (2006). The persona lifecycle: keeping people in mind throughout product design. Elsevier.

Rosson, M. B. and Carroll, J. M. (2002). Usability engineering: scenario-based development of human-computer interaction. Academic Press.

Schneier, B. (2012). Liars & Outliers: Enabling the Trust That Society Needs to Thrive. John Wiley & Sons.

van Lamsweerde, A. (2004). Elaborating security requirements by construction of intentional anti-models. In Proceedings of the 26th International Conference on Software Engineering, pages 148–157. IEEE Computer Society.

van Lamsweerde, A. (2009). Requirements Engineering: from system goals to UML models to software specifications. John Wiley & Sons.

van Lamsweerde, A. and Letier, E. (2000). Handling obstacles in goal-oriented requirements engineering. IEEE Transactions on Software Engineering, 26(10):978–1005.