Chapter 6
Assessing Security and Privacy Behavioural Risks for Self-Protection Systems

Yijun Yu, Yoshioka Nobukazu, and Tetsuo Tamai

Abstract Security and privacy can often be considered from two perspectives. The first perspective is that of the attacker who seeks to exploit vulnerabilities of the system to harm assets such as the software system itself or its users. The second perspective is that of the defender who seeks to protect the assets by minimising the likelihood of attacks on those assets. This chapter focuses on analysing security and privacy risks from these two perspectives considering both the software system and its uncertain environment including uncertain human behaviours. These risks are dynamically changing at runtime, making them even harder to analyse. To compute the range of these risks, we highlight how to alternate between the attacker and the defender perspectives as part of an iterative process. We then quantify the risk assessment as part of adaptive security and privacy mechanisms complementing the logic reasoning of qualitative risks in argumentation (Yu et al., J Syst Softw 106:102–116, 2015). We illustrate the proposed approach through the risk analysis of examples in security and privacy.

6.1 Introduction

Security properties are often described using confidentiality, integrity, availability, authentication and authorisation according to the ISO/IEC 9126 standard [5], which highlight the protection of asset values from malicious attacks. Privacy properties, as understood by "the rights to be left alone" [13], concern the control of sharing

Y. Yu (✉)
The Open University, Milton Keynes, UK
e-mail: y.yu@open.ac.uk

Y. Nobukazu
National Institute of Informatics, Tokyo, Japan
e-mail: nobukazu@nii.ac.jp

T. Tamai
Hosei University, Tokyo, Japan
e-mail: tamai@hosei.ac.jp

Y. Yu et al. (eds.), *Engineering Adaptive Software Systems*,
https://doi.org/10.1007/978-981-13-2185-6_6

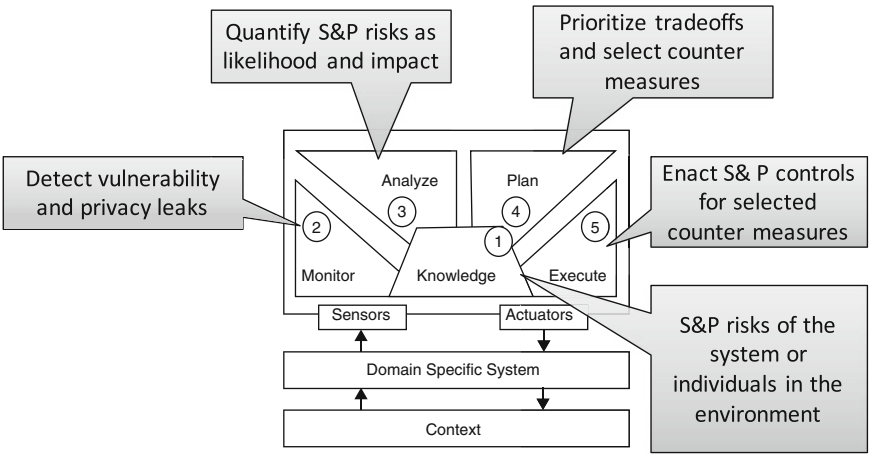

Fig. 6.1 Security and privacy concerns on the MAPE-K architecture [4] for self-protection

the identity of individuals or groups to prevent potential harms to their life and can be represented using selective disclosure [12], contextual integrity [1], etc.

Both security and privacy properties are adaptive in nature. From the dimensions of self-adaptive systems, known as MAPE-K feedback loops [2], both security and privacy can be seen as cross-cutting concerns to all these five dimensions at runtime. Self-adaptive security and privacy mechanisms, or *self-protection*, instantiate the MAPE-K dimensions [4] as follows (see Fig. 6.1).

Monitoring aims to detect system vulnerability and privacy leaks. *Analysis* requires a quantification of risks in terms of assessing of the likelihood and impact of runtime incidents. *Planning* involves the ranking, prioritisation and trade-offs of incident responses in order to select the best countermeasures at runtime. *Execution* enacts the defence to control the managed system or individuals and implement the countermeasures. Throughout these activities, the *knowledge* about the system and the individuals also change over time, which could change the predefined boundaries between attackers and defenders.

According to our earlier studies on security risks [16] and privacy arguments [12], it is necessary to analyse contextual factors in order to identify and assess the risk factors. These approaches proposed to use problem-oriented analysis on the context of a system in its running environment, in order to elicit the risk factors from a prepared knowledge base (e.g. common vulnerability exposures[1] and common vulnerability scoring system[2]).

In addition to MAPE-K feedback loops for self-protection, system and individuals also need to quantify the security and privacy risk factors at runtime. Such quantified risks could help tune the *set points* [3] for runtime security and privacy feedback loop controls. However, runtime properties of the system and environment

[1] https://cve.mitre.org

[2] https://nvd.nist.gov/vuln-metrics/cvss

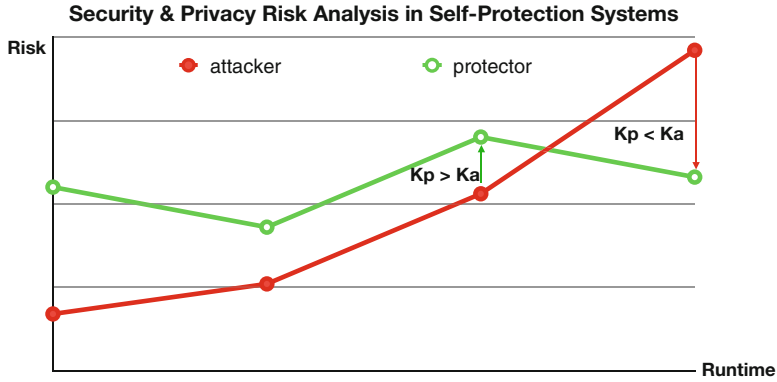

Fig. 6.2 Knowledge adaptation centric to assessing security and privacy risks

typically involve dynamic behaviours; therefore, it is also necessary to consider the behaviour models explicitly.

The uncertainty in self-adaptive systems can be classified into *unknown knowns*, *known unknowns* and *unknown unknowns* [11], where the known unknowns shall be addressed at the runtime, leaving unknown knowns and unknown unknowns to the approaches that perform machine learning, which is beyond the scope of this chapter. Specifically, in this chapter, we would like to address the following research questions:

- Can behavioural models be modified at runtime to reflect the new changes of the known unknowns at design time?
- Can the known unknowns be explicitly defined as parameters on top of the behavioural models at runtime?
- When an attacker can adapt their behaviour according to their knowledge superior to what the protector knows, would self-adaptation capability be misused to hurt security?
- If it cannot be prevented to misuse self-adaptive systems, how can we exert runtime control into the design against this possibility of misuses?

Conceptually, Fig. 6.2 depicts the relationship between the knowledge of defender (K_p) and the attacker (K_a). In principle, when the attacker has more knowledge than the protector, the security and privacy risks of self-adaptive systems are likely to increase. When the defender knows more, the risks can be controlled better. However, the knowledge boundary between the defender and attacker is not always explicitly defined and can change over time. Therefore, the reassessment of the security and privacy risks needs to be performed continuously. Taking the ancient analogy of *spears* (矛) for attacks and *shields* (盾) for protections, the best means for meeting security goals and anti-goals are not staying constant. When the two sides are confronting each other at runtime, the winner is not always predictable unless there is a systematic way to manage the changes.

The key question to ask is, if such analogy holds, whether there is a way to quantitatively access the impact of self-adaptation on security and privacy? When the frontier knowledge is changed, both self-adaptive system and security controls try their best to deal with uncertain behaviours. How to ensure or maintain the level of security when systems are facing such uncertain adaptive behaviours?

In this chapter, we will use example behaviour models to illustrate these challenges and demonstrate the need to expand the knowledge boundary for a self-protection system.

6.1.1 Motivating Examples

To illustrate the problem and the proposed solutions, we show two example systems briefly here to give the context.

6.1.1.1 A PIN Entry Device System

PINs are used for ATM and various smartphones to authenticate users. They are not as hard as online banking protection because the password allowed to use is limited to a few digits. However, such systems are widely used because it demands little memory from users, hence offering a bit more usability. Since it is widely used and simple, we use this example to illustrate the basic concepts in our risks analysis.

6.1.1.2 A Social Media System

Social media such as Facebook are widely used to connect people by posting messages to friends who can pass them onwards to the friends of friends. Privacy, however, it is a key asset to protect so that the information is not passed on the unintended audience. Since the users of social media systems follow their instinct to share posts, it is likely such privacy concerns are violated. The user behaviour-based risk analysis approach proposed in this chapter will be exemplified, again using a simplified behaviour model of a social media system of Facebook.

6.2 Abstract Goal Behaviour Models

In order to perform such an analysis on security and privacy risks, we first introduce the behaviour models for agents, including both human and machine, according to Jackson's abstract goal behaviour models [6].

Definition 6.1 (Behaviour Model) The behavioural model of a domain (or a machine) is represented by a state machine, denoted as a tuple $< S, s_0, T, G, A >$ where S is the set of states, $s_0 \in S$ is an initial state, A is the set of actions on an

alphabet, $T : S \times A \times S$ is a set of A labelled transitions between the states, and $G : T \times \mathcal{B}$ is the set of Boolean guard conditions, indicating whether a transition can be fired.

We assume that the agents have goals that determine their interpretations of the current states of the domains in the world.

Definition 6.2 (Goals) A goal g can be defined as a certain property that holds on the desired states. In other words, given an initial state s_0, the goal of a system can be described by a set of states $s \in S$ such that $g(s)$ is true.

Consider the satisfaction of goals; according to van Lamsweerde [8], there are four typical modes. An *ACHIEVE* goal is described by $\neg g(s_0) \wedge g(s)$, indicating that the goal property was not initially true; a *MAINTAIN* goal is described by $g(s_0) \wedge g(s)$, indicating that initially established property is maintained to be true; a *CEASE* goal is described by $g(s_0) \wedge \neg g(s)$, indicating that the initially established "anti-goal" is no longer true; and an *AVOID* goal is described by $\neg g(s_0) \wedge \neg g(s)$, which avoids the satisfaction of an anti-goal. All these modes can be mapped nicely to security and privacy goals, where the *ACHIEVE* goal of a protector can be regarded as the *CEASE* goal of an attacker and vice versa.

Definition 6.3 (Abstract Goal Behaviours) Since requirements goals are prescriptive on the machine and domains, we can establish the following basic requirements satisfaction argument according to [17]:

$$W, S \models R \tag{6.1}$$

where W and S are nothing but the properties of behaviour models with respect to world context domains and specification of the machine, respectively, while R is the abstract goal behaviours desired by composing these behaviour models.

The intermediate concept of abstract goal behaviours connects the requirements properties with respect to those of the machine domains. When problem domains are biddable or uncertain (e.g., human actors are non-deterministic), we need to handle the uncertainty by considering probabilistic behaviours. In order to quantify these abstract behaviour models, we introduce the notion of risks in terms of likelihood and impact, as follows.

6.3 Risks in Behaviour Models

In this section, we give the definitions of R-DTMC behavioural models and their extension for modelling transparency. Then we provide the technical details of algorithms to compute the security risks by composing the models and the adaptive transparency of risk functions.

Definition 6.4 (Discrete Time Markov Chains (DTMC), Reward DTMC) A *DTMC* extends a state machine by a function $\pi : \delta \to [0, 1]$ that is the probability for a transition in T to be successfully fired. For every state s, the sum of the probability of its outgoing transitions is $\sum_{s'|(s,s')\in\delta} \pi(s, s') = \{0, 1\}$. When the sum is zero, the state is an *absorbing* or *final* state. Furthermore, an R-DTMC extends a DTMC with an impact function $\mathcal{I} : S \to [0, \infty)$ as the reward for reaching a state $s \in S$, typically it indicates the impact of damage on the assets.

Note that in its general form, R-DTMC could associate an impact on transitions as well. In this work, we do not require this level of generality because we have been focusing on the risks of damaging assets at these states, rather than the risks of certain actions on the transitions. By associating the impact with the source state of a transition, our state-only representation of impact is equivalent to associating any impact with the transition.

Definition 6.5 (Traces and Risks) From Definition 6.4, a *trace* $\langle s_0, s\rangle$ from the initial state s_0 to a state s is defined by a sequence of n transitions $(s_k, s_{k+1}) \in \delta$ where $n > 0, k = 0, \ldots, n-1$, and $s_n = s$. From the same pair of states s_0 and s, there could be more than one trace, and these traces may have different lengths. For a given trace $\langle s_0, s_n\rangle$ of length n, the *likelihood* $p(s)$ is defined as follows:

$$p(s) = \prod_{k=0}^{n-1} \pi(s_k, s_{k+1}) \tag{6.2}$$

and the associated *risk* $r^n(s)$ is defined as the product of impact $I(s)$ and likelihood $p(s)$:

$$r^n(s) = I(s) \times p(s), \tag{6.3}$$

which measures how the impact could take effect when the state at the end of the trace is reached from the initial state, at certain likelihood. Considering all possible traces from s_0 to s, the aggregate risk on the state s is given as

$$r^*(s) = \sum_{n=1}^{\infty} \sum_{\langle s_0,s\rangle\in\delta^n} r^n(s). \tag{6.4}$$

One can use a naïve Algorithm 1 to simulate a stochastic decision process which walks on a random transition of each state with respect to the probability distribution of the outgoing transitions. The input also contains two thresholds, t for the total number of traces to simulate and n for the maximal length of the traces before a final state is reached. When there could be infinite length of a trace due to cycles, the simulation forces a trace to terminate when its length is larger than a certain threshold (Lines 2) or when it already has no further transitions (Lines 3–5). Otherwise, the random walk is based on a uniformly distributed random number

Algorithm 1: Compute risks by simulating a stochastic decision-making process through random walks

Data: An R-DTMC (S, δ, s_0, π, $\mathcal{I}$) per Definition 6.4;
t, maximal number of traces to simulate through random walks;
n, maximal length of traces to simulate;
Result: Approximated risks $r^n(s)$ per Definition 6.5 from the simulated traces

1 **for** c:=1 *to* t **do**
2 **for** $l := 1..n$ **do**
3 **if** $\sum_{(s_{l-1},s_k)\in\delta} \pi(s_{l-1}, s_k) = 0$ **then**
4 **break**;
5 **end**
6 $r :=$ random(0, $\sum_{(s_{l-1},s_k)\in\delta} \pi(s_{l-1}, s_k)$) where k indexes outgoing transitions of s_{l-1};
▷ uniformly distributed random number
7 Let $s_l = s_k$ where k is the minimal number so that $r < \sum_{(s_{l-1},s_k)\in\delta} \pi(s_{l-1}, s_k)$;
8 $r^*(s_l) := r^*(s_l) + i(s_l)$;
9 **end**
10 **end**
11 $r^n(s) := r^n(s)/t$ for each state s;
12 return $r^n(s)$;

generator (Line 6). When it falls into the slot by the probabilistic distribution of the outgoing transitions from the previous state s_{l-1}, the corresponding outgoing transition will be assumed (Line 7). On that transition, the risk of reaching the current state s_l will be updated by adding its impact (Line 8).

Finally, the risk is computed as dividing the aggregated impact by the number of simulated traces through random walks, t (Line 11). Note that the chance of selecting an outgoing transition of the previous state is proportional to the probability of the outgoing transitions.

The time complexity of Algorithm 1 in terms of the number of random decisions is $O(tn)$. To get more precision, both t and n need to be larger. Yet when the machine contains cyclic transitions, it is impossible to enumerate all traces.

Depending on the probabilities assigned to the cyclic transitions, the risks in Algorithm 1 are an approximation on the threshold of n, which may not converge to constants when n increases. To illustrate, consider any transitions that form a self-cycle $(s, s) \in \delta$ with $\pi(s, s) = 1$. In such a trace, the state s will be visited n times with the likelihood of 1. Its risk, computed by the simulation, $n \times I(s)$, will increase proportionally to n.

When the cyclic exploration machine gets more complex, however, it is no longer obvious whether the risk computation by simulation converges or not. Even when it converges, a large number of enumerations could be taken to approximate the

risks in order to achieve high precision. The challenge is, given an R-DTMC, there should be a way to tell whether the risks converge or not without lengthy simulations. Furthermore, is there a way to compute the converging risks precisely and efficiently, without all the simulations?

In a matrix form, the likelihood computation can be rewritten as solving the likelihood vector $\mathbf{p}$ to a system of recurrence equations:

$$\begin{aligned} \mathbf{p} &= P\mathbf{p} + \mathbf{c} \\ \mathbf{p} &\geq \mathbf{0} \end{aligned} \tag{6.5}$$

where P is a $n \times n$ transition probabilities square matrix and $\mathbf{c} = (1, 0, \ldots)$ and $\mathbf{p}$ are $1 \times n$ vectors of nonnegative real numbers.

Rewriting this as a linear equation where I stands for the identity matrix of dimension $n \times n$, i.e. $\mathbf{p}I = \mathbf{p}$, we have:

$$(I - P)\mathbf{p} = \mathbf{c} \tag{6.6}$$

The solution of $\mathbf{p}$ can be obtained as:

$$\mathbf{p} = (I - P)^{-1}\mathbf{c} \tag{6.7}$$

When the probability matrix P and the impact vector $\mathbf{i}$ have elements of non-numeric expressions, we call them *symbolic*. When P is not lower-triangular matrix, we call the model *cyclic*, which can still be solved into a risk profile function by applying symbolic LDU decompositions recurrently. Limited by space, we have put the details of algebraic computation of the risk into a technical report,[3] with a proof that when the behavioural model converges (i.e. the necessary and sufficient condition requires a single exit state which does not have any outgoing transitions), the resulting risk profile function can be obtained without simulating on every combination of values in Algorithm 1.

6.4 Running Examples

In [16], we introduced a systematic approach to elicit risk factors from the context diagrams of a software system. The example we used is PED (PIN entry device), where certain security risks have been identified "quantitatively". However, since the quantification was based on natural language processing and CVSS records, it is not yet associated with the behavioural models, hence cannot be applied at runtime.

Here let us first simplify the example so that it is easy to see how risk assessment can be quantified onto the behavioural models.

[3] https://github.com/yijunyu/demo-riskexplore/tree/master/doc

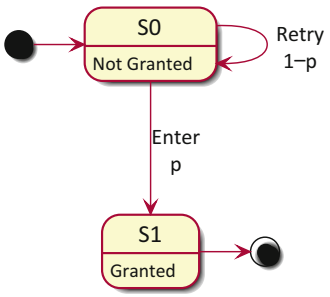

Fig. 6.3 A simple behaviour model for security risks assessment

6.4.1 Security Risks in PIN Access Control

Figure 6.3 illustrates how attackers could gain access to the account after infinite number of trials. Such cyclic behaviour model is quite common in real life; however, existing simulation-based model checkers are not able to detect some flaws in the model.

When the probabilities of the transitions and the impact of the states are unknown, we need to change the way of looking at them as numeric values, but as algebraic symbols (i.e. known unknowns) instead.

Assume that the overhead for login was $-O$, which rewards the attacker by the value of bank account V, then the risk of loss is estimated to be

$$-O/p + V \tag{6.8}$$

When p is small enough, the following condition could provide some relative assurance of the system security:

$$O/p > V \tag{6.9}$$

This explains why an effective policy for preventing denial-of-service attacks is to introduce some overhead to the users while logging in to the system, so that it is not worthwhile to try indefinitely.

6.4.2 Privacy Risks in Social Networks

While social networks systems are used by individual users, they may choose to share posts to the friends, with a non-negligible probability that the friends may share the posts further to unwanted or undesirable audience. The trade-offs between sharing and not sharing, with respect to social benefits such as likes and resharing, are frequent decisions to be made by the individual. The rationale of such decisions are typically risk assessments on the basis of simulating the effect of leaking the private information to unintended audience [14].

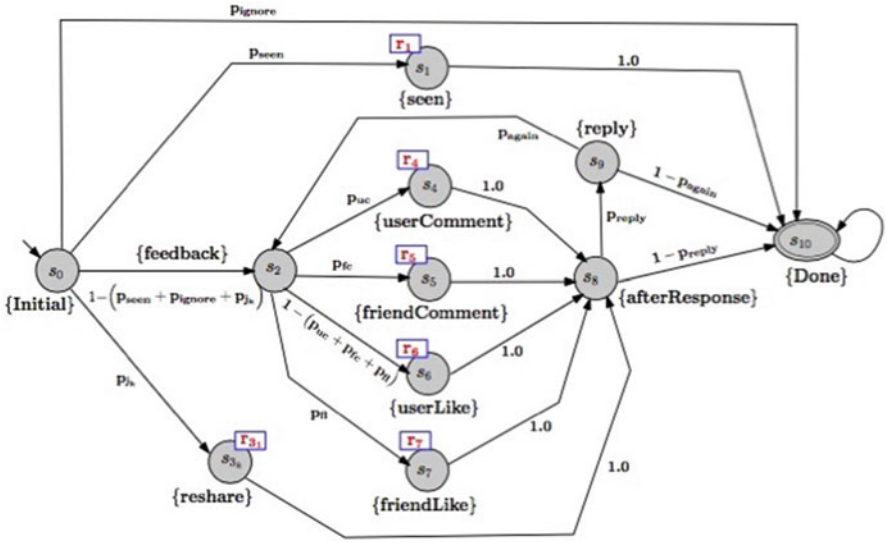

Fig. 6.4 A privacy risk assessment model taken from [10]

Recently, the original work in [14] has been extended to introduce a much more complex behavioural model of sharing [10] by introducing inductive machine learning techniques similar to those of recommendation systems. In other words, patterns of groups emerging from the social circles are learnt by different individuals, while they are making similar decisions.

However, simulation-based approaches are inherently incomplete. For example, the risk assessment model in Fig. 6.4 is a little bit more complicated than the security one we discussed earlier. It is based on the individual's decisions to share post on a social network and the estimation of the risk exposure to the audience if the information is sensitive.

The behaviour model was built using PRISM [7], but the algebraic symbols are computed differently here. After applying our risk explorer tool,[4] the risk profile function can be obtained as such:

```
pseen*r1+pjk*r3-(r4*puc*(pjk-(1-pseen-pignore)
-pagain*preply*pjk))/(1-pagain*preply*pfl
+pagain*preply*(pfl-(1-puc-pfc))-pagain*preply*pfc
-pagain*preply*puc)-(r5*pfc*(pjk-(1-pseen-pignore)
-pagain*preply*pjk))/(1-pagain*preply*pfl
+pagain*preply*(pfl-(1-puc-pfc))-pagain*preply*pfc
-pagain*preply*puc)+(r6*(pfl-(1-puc-pfc))*(pjk-
(1-pseen-pignore)-pagain*preply*pjk))/(1
-pagain*preply*pfl+pagain*preply*(pfl-(1-puc-pfc))
-pagain*preply*pfc-pagain*preply*puc)-(r7*pfl*(pjk
```

[4] https://github.com/yijunyu/demo-riskexplore

```
-(1-pseen-pignore)-pagain*preply*pjk))/(1
-pagain*preply*pfl+pagain*preply*(pfl-(1-puc-pfc))
-pagain*preply*pfc-pagain*preply*puc)
```

where the following determinant condition has to be satisfied; otherwise the behavioural model will not converge to a solution:

```
0<1-pagain*preply*pfl+pagain*preply*(pfl-(1-puc-pfc))

  -pagain*preply*pfc-pagain*preply*puc
```

By applying an optimisation algorithm to minimise the risk profile function, e.g. differential evolution optimisation [9], it is possible to obtain a near-optimal solution in less than 1 min.

For example, when $r1 = r2 = r3 = r4 = r5 = r6 = r7 = 1$, the lowest risk of 0.012 can be achievable when $pseen = 0.006239582$, $pignore = 0.987266657$, $pjk = 0.003001324$, $puc = 0.115949677$, $pfc = 0.446085095$, $pfl = 0.131866686$, $preply = 0.003728548$ and $pagain = 0.048901284$.

6.5 Discussions

Of course, this combination of the known unknowns for "minimum" risks is derived without considering any constraints. With more knowledge at runtime, either for the protector or for the attacker, the minimal risk would not look the same because they would see these unknowns differently.

In principle, both normal user and attackers can be modelled as biddable domains, in which not all decisions are deterministic and not all states are explicit. In other words, unless we are the attackers, such models are just intellectual guesses. Nonetheless, having a model allows us to estimate the risks of actions of individual agents.

We assume that the attackers and the defenders have different knowledge of the system. In other words, through observations, the attacker could realise some vulnerabilities before the defenders knows, and the defenders certainly could know some internal designs that the attacker may not know.

In the following, we show an example where the attacker knows a vulnerability before it manifests to the public.

Suppose the protector initially assume that the PIN code protection used in the system is uniformly distributed and to guess correctly the 4 digits one would have to try 10,000 combinations in the worst case and 5000 in the average case. However, through key loggers or other means, attackers could estimate the distribution of probability so that p increases to 0.5. In that case, the current behavioural model could no longer offer sufficient protection since the risk increases dramatically. Similarly, if the protector knows that the account has $0 in value, while the attacker does not, it becomes easier for the protector to set up a trapping "honey pot" in order to catch such reckless attackers. In this case attackers would face higher risks.

6.6 Summary

In this chapter, we have articulated the need to quantify the risks for self-protection, i.e. offering both protectors and attackers' perspectives in assessing the risks. The known unknowns, in this work, manifest as symbolic probabilistic variables appearing on the guard condition of transitions in behavioural models. We have also used two examples from security and privacy application domains to illustrate the advantage of such quantified risk exploration.

Note that the work of risk exploration is an ongoing research effort, where we have developed open-source tools for colleagues to use and compare with our results https://github.com/yijunyu/demo-riskexplore. A guide tour of risk exploration can be found in the tutorial [15].

In the future, we hope to improve the efficiency of our quantitative risk exploration tool so that self-protection systems could be armed with the runtime behaviour models to define the set points for efficient self-adaptation.

References

1. Barth, A., Datta, A., Mitchell, J.C., Nissenbaum, H.: Privacy and contextual integrity: framework and applications. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy, SP'06, pp. 184–198. IEEE Computer Society, Washington, DC (2006). https://doi.org/10.1109/SP.2006.32
2. Brun, Y., Di Marzo Serugendo, G., Gacek, C., Giese, H., Kienle, H., Litoiu, M., Müller, H., Pezzè, M., Shaw, M.: Engineering Self-Adaptive Systems through Feedback Loops, pp. 48–70. Springer, Berlin/Heidelberg (2009). https://doi.org/10.1007/978-3-642-02161-9_3
3. Chen, B., Peng, X., Yu, Y., Zhao, W.: Requirements-driven self-optimization of composite services using feedback control. IEEE Trans. Serv. Comput. **8**(1), 107–120 (2015). https://doi.org/10.1109/TSC.2014.2298866
4. Iglesia, D.G.D.L., Weyns, D.: Mape-k formal templates to rigorously design behaviors for self-adaptive systems. ACM Trans. Auton. Adapt. Syst. **10**(3), 15:1–15:31 (2015). https://doi.org/10.1145/2724719
5. ISO/IEC: Iso/iec 25010 system and software quality models. Technical report (2010)
6. Jackson, M.: System behaviours and problem frames: concepts, concerns and the role of formalisms in the development of cyber-physical systems. In: Dependable Software Systems Engineering, pp. 79–104 (2015). https://doi.org/10.3233/978-1-61499-495-4-79
7. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) Proceedings of 23rd International Conference on Computer Aided Verification (CAV'11), Snowbird. LNCS, vol. 6806, pp. 585–591. Springer (2011)
8. van Lamsweerde, A.: Goal-oriented requirements engineering: a guided tour. In: 5th IEEE International Symposium on Requirements Engineering (RE 2001), 27–31 Aug 2001, Toronto, p. 249 (2001). https://doi.org/10.1109/ISRE.2001.948567
9. Mullen, K., Ardia, D., Gil, D., Windover, D., Cline, J.: DEoptim: an R package for global optimization by differential evolution. J. Stat. Softw. **40**(6), 1–26 (2011). http://www.jstatsoft.org/v40/i06/
10. Rafiq, Y., Dickens, L., Russo, A., Bandara, A.K., Yang, M., Stuart, A., Levine, M., Calikli, G., Price, B.A., Nuseibeh, B.: Learning to share: engineering adaptive decision-support for

online social networks. In: Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE '17), IEEE Press, Piscataway, pp. 280–285 (2017)
11. Sutcliffe, A., Sawyer, P.: Requirements elicitation: towards the unknown unknowns. In: 2013 21st IEEE International Requirements Engineering Conference (RE), Rio de Janeiro, pp. 92–104 (2013). https://doi.org/10.1109/RE.2013.6636709
12. Tun, T.T., Bandara, A.K., Price, B.A., Yu, Y., Haley, C., Omoronyia, I., Nuseibeh, B.: Privacy arguments: analysing selective disclosure requirements for mobile applications. In: 2012 20th IEEE International Requirements Engineering Conference (RE), Chicago, pp. 131–140 (2012)
13. Warren, S.D., Brandeis, L.D.: The right to privacy. Harvard Law Rev. **4**(5), 193–220 (1890). http://www.jstor.org/stable/1321160
14. Yang, M., Yu, Y., Bandara, A.K., Nuseibeh, B.: Adaptive sharing for online social networks: a trade-off between privacy risk and social benefit. In: 13th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2014, Beijing, 24–26 Sept 2014, pp. 45–52 (2014). https://doi.org/10.1109/TrustCom.2014.10
15. Yu, Y.: Risk assessment using early requirements models: a guided tour. In: 25th International Requirements Engineering Conference, Tutorial, Lisbon (2017)
16. Yu, Y., Franqueira, V.N.L., Tun, T.T., Wieringa, R., Nuseibeh, B.: Automated analysis of security requirements through risk-based argumentation. J. Syst. Softw. **106**, 102–116 (2015). https://doi.org/10.1016/j.jss.2015.04.065
17. Zave, P., Jackson, M.: Four dark corners of requirements engineering. ACM Trans. Softw. Eng. Methodol. **6**(1), 1–30 (1997). https://doi.org/10.1145/237432.237434