Analysis of Social Engineering Threats with Attack Graphs

Kristian Beckers[1], Leanid Krautsevich[2], and Artsiom Yautsiukhin[2](✉)

[1] Paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen, Duisburg, Germany
kristian.beckers@paluno.uni-due.de

[2] Istituto di Informatica E Telematica, Consiglio Nazionale Delle Ricerche, Via G. Moruzzi 1, 56124 Pisa, Italy
{leanid.krautsevich,artsiom.yautsiukhin}@iit.cnr.it

Abstract. Social engineering is the acquisition of information about computer systems by methods that deeply include non-technical means. While technical security of most critical systems is high, the systems remain vulnerable to attacks from social engineers. Social engineering is a technique that: (i) does not require any (advanced) technical tools, (ii) can be used by anyone, (iii) is cheap.

While some research exists for classifying and analysing social engineering attacks, the integration of social engineering attackers with other attackers such as software or network ones is missing so far. In this paper, we propose to consider social engineering exploits together with technical vulnerabilities. We introduce a method for the integration of social engineering exploits into attack graphs and propose a simple quantitative analysis of the graphs that helps to develop a comprehensive defensive strategy.

Keywords: Social engineering · Threat analysis · Attacker modelling · Attack graph

1 Introduction

A study[1] of 2011 from Dimensional Research considered 853 IT professionals from United States, United Kingdom, Canada, Australia, New Zealand, and Germany concluded that: (i) 48 % of large companies and 32 % of small companies were victims of 25 or more social engineering attacks in the past two years, (ii) an average cost per incident is over \$25 000 and (iii) 30 % of large companies even cite a per incident cost of over \$100 000. In addition, the SANS institute report in a white paper[2] about social engineering that cyber attacks cost U.S.

[1] Dimensional Research Study about Social Engineering http://www.checkpoint.com/press/downloads/social-engineering-survey.pdf.

[2] SANS Institute InfoSec Reading Room http://www.sans.org/reading-room/whitepapers/engineering/threat-social-engineering-defense-1232.

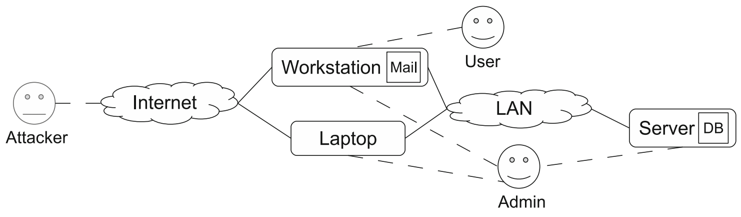

Fig. 1. An infrastructure model

companies $266 million every year and that 80 % of all attacks are caused by authorised users that are either disgruntled employees or non-employees that have established some form of trust within a company. The study also cites Kevin Mitnick a famous hacker, who stated in a BBC interview: "The biggest threat to the security of a company is not a computer virus, an unpatched hole in a key program or a badly installed firewall. In fact, the biggest threat could be you. What I found personally to be true was that it's easier to manipulate people rather than technology. Most of the time organizations overlook that human element." Thus, we can conclude that social engineering is still an important security issue to address.

Social engineering threats have been classified and analysed in the past [1–4]. Research into social engineering, e.g., [5], showed that the attacks often follow a simple process: gather information about the target, develop and exploit a trust relationship, and utilise the gathered information. Possible countermeasures are organisational rules in the form of security policies and staff training to recognise and prevent social engineering attacks, so-called awareness training.

One of the problem with tackling the social engineering threat is that security analysis of social engineering attackers is often isolated c.f. [1,3,6] from a technical security analysis, e.g., network vulnerability analysis. In this work, we investigate how social engineering exploits may completely or partially substitute technical vulnerabilities in an attack. We aim at enhancing our existing threat analysis methodology [8] with a social engineering threat analysis. Our methodology uses a detailed attack graph that represents systems as connected sets of vulnerabilities and that helps to analyse what steps the attacker needs to execute in order to achieve his goal. We focus on the integration of social engineering exploits into a combined attack graph for a further quantitative security analysis.

Our main contributions are: (i) formalisation of social engineering threats using threat patterns; (ii) (semi-automatic) identification of the existing social engineering vulnerabilities for a concrete system using access control rules and some information about the system; (iii) consolidation of social engineering vulnerabilities with network vulnerabilities in a combined attack graph; (iv) quantitative analysis of possible attacks considering the combined attack graph.

The remainder of the paper is structured as follows: Sect. 2 shows our research gap with respect to the related work. Section 3 describes several types of social engineering threats. Our methodology is shown in Sects. 4 and 5 concludes our paper and provides directions for future research.

Running example. We consider the Huntsville Consortium that sells energy to customers (see [8] for more details). There is an infrastructure which is responsible for storage of the metering data and issuing bills (see Fig. 1). The information is stored on a server `serv`, which runs under FreeBSD OS, in a database `db` (Oracle MySQL). The operator (`user`) uses a workstation `ws`, which runs under Windows OS, to process the data from the database. There is also an administrator `adm` of the network who uses a laptop `lap` to manage the server and the workstation. The laptop runs under Linux OS. The internal firewall allows access to a server from the workstation and the laptop only through `LAN`. The workstation and the laptop are connected to the `Internet`. The operator and the administrator have access to `e-mail` for interaction with clients and between themselves. Finally, this organisation is located in the same `building` without (properly) guarded access into it. The aim of the attacker is to change the data in the database to avoid paying for electricity.

2 Related Work

The only work we are aware of that tries to combine social engineering attackers with network attackers in a threat analysis is Kvedar et al. [7]. The authors used social engineering techniques successfully in a case study to acquire knowledge about network vulnerabilities as part of training exercise in the U.S. military. In contrast to our work the authors do not consider social engineering as separate exploits, but use it simply as a mean to acquire domain knowledge, e.g., firewall configurations. To show this gap in research we provide an overview of existing work in the areas of social engineering threat taxonomies and countermeasures for social engineering. We use attack graph analysis to analyse threats to provide an overview of this field as well.

Analysing Social Engineering Threats. Krombholz et al. [6] and Ahmadi et al. [9] presented overviews of social engineering threats, the communication channels used to conduct these attacks, e.g., social networks or instant messaging. Mills [10] focused on the problem of social networking offers with regard to the risk of information leakage. Chitrey et al. [11] conducted a survey using a questionnaire with IT practitioners in India with the aim of understanding the awareness of practitioners of social engineering threats. Furthermore, Laribee et al. [4] created models to describe social engineering threats in general. While Algarni and Xu [2] described two different attacker models for the domain of social networking sites. In addition, Dimkov et al. [3] contributed methodologies and guidelines for physical penetration testing using social engineering.

Countermeasures against Social Engineering Attacks. Severals works proposed social engineering countermeasures that include security awareness trainings, policies, incident reporting systems, and penetration testing. Winkler et al. [12] and Mitnick [5] described a large number of attacks and showed how to mitigate them using their catalogs of countermeasures. Peltier [1] and Gonzales et al. [13] based their collections of countermeasure on human behavioural patterns, which can be exploited by social engineering attackers. Twitchell [14] provided recommendations of how to teach people about social engineering threats and countermeasures in security training courses. Moreover, social engineering is also considered in security standards. The ISO 27001 [15] international standard contains a list of countermeasures that includes for example Control A.7.2.2. information security awareness, education and training, and A.5.1.1 policies for information security. Similar controls can be found in national guidelines, e.g., the German Grundschutzhandbuch IT [16].

Attack Graph Analysis. An attack graph is a technique for security modelling and analysis of a system which specifies states, related to the privileges the attacker may have, and transitions between them. There are two types of attack graphs in the literature. The first type denotes every state as a set of all privileges the attacker possesses at a certain stage of an attack [17]. The graph in this case is acyclic if we assume that an attacker cannot loose her privileges. The advantage of this model is that such type of analysis as Markov Decision Process (MDP) may be applied to it [18,19]. The main drawback of this model is the state-explosion problem. The second type is proposed by Noel and Jajodia [20] who represented nodes as atomic privileges. An attacker possesses several privileges at some stage of an attack, "owns" several nodes, i.e., the privileges she has are the union of the privileges assigned to these vertices. A transition requires "owning" certain vertices and leads to new privileges. This model is free from the state explosion problem, but has cycles and cannot be used for analysis with MDP.

3 Social Engineering Exploits

We summarise several common social engineering exploits identified by Krombholz et al. [6]. We choose all possible exploits for which we could define pre and post conditions. For example, we excluded the waterholing exploit in which an attacker compromises a website that is likely to be of interest for an employee. Formulating the "likely interest" is rather difficult and requires a deeper understand of psychological sciences.

Phishing refers to masquerading as a trustworthy entity and using this trust to acquire information or manipulating somebody to execute an action.

Shoulder Surfing means to obtain information from a display by being physically close to it and reading the information on the screen.

Dumpster Diving is the act of analysing the documents and other things in a garbage bin of an organisation to reveal sensitive information.

Reverse Social Engineering is to create a problem for a victim and to offer help to the victim for solving the problem. This way the attacker establishes trust, which is used to exploit the victim for sensitive information.

Baiting is to leave a storage medium (e.g., a USB stick) inside a company location that contains a malicious software (e.g., a key logger). The malicious software is executed automatically when the stick is inserted in a computer.

Table 1. Social engineering exploits

ID	*Name*	Precondition	Postcondition
SEE-0001	**Phishing**	☒**Communication Channel** ☐**Physical Access**	☒**Credentials** ☐ **Access**
SEE-0002	**Shoulder Surfing**	☐**Communication Channel** ☒**Physical Access**	☒**Credentials** ☐ **Access**
SEE-0003	**Dumpster Diving**	☐**Communication Channel** ☒**Physical Access**	☒**Credentials** ☐ **Access**
SEE-0004	**Reverse Social Engineering**	☒**Communication Channel** ☒**Physical Access** ☒**Physical Access**	☒**Credentials** ☒**Access**
SEE-0005	**Baiting**	☐**Communication Channel** ☒**Physical Access**	☒**Credentials** ☒**Access**

We derived pre and post conditions for these exploits (see Table 1) and recognised that social engineering exploits are possible using different communication channels such as: Telephone, VoIP, In person, Email, Fax, Instant messaging, and Social networks. For simplicity's sake, we do not distinguish between them in this work. Note that in all pre conditions the attacker has to be in possession of domain knowledge about the target, e.g., the attacker has to know how to use a communication channel to contact employees. Moreover, an attacker often needs physical access to the target organisation, for example in order to distribute USB sticks in the baiting exploit. Note that in some exploits technical access might also be required, e.g., in the Reverse Social Engineering exploit to create a technical problem with a machine.

Social engineering exploits often result in an attacker (illegally) obtaining credentials of a user of an organisation, e.g., a username and password combination. This combination can be used by an attacker to authenticate herself as the user. Moreover, a user can be compromised to provide access to an operating system to an attacker, e.g., via being asked to run a malicious program on the attacker's behalf or to perform some operation for the attacker. In summary, we consider credentials and access as possible outcomes of social engineering exploits and state them as post conditions in Table 1.

4 Analysis of Social Engineering Threats with Attack Graphs

We propose a combined analysis of a system against social engineering attackers and attackers with technical skills. We show the possible interactions of these attackers in an attack graph. The graph aims at finding sequences of low-level actions that an attacker needs to execute in order to achieve her high-level goals. Figure 2 shows our security analysis method. Further in this section we describe the steps of our approach in details.

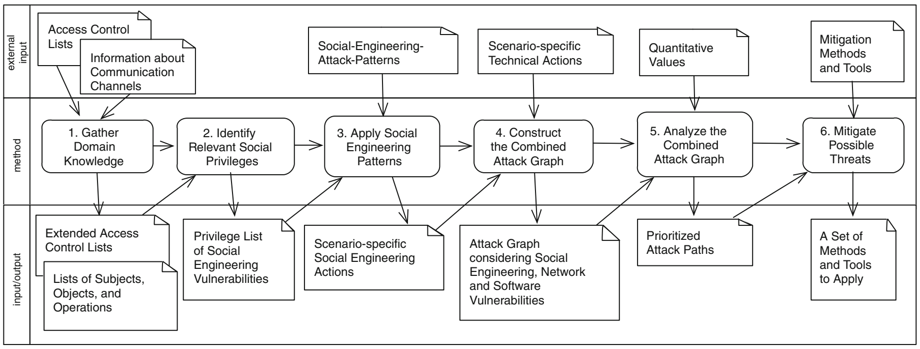
external input
method
input/output
Access Control Lists
Information about Communication Channels
Social-Engineering-Attack-Patterns
Scenario-specific Technical Actions
Quantitative Values
Mitigation Methods and Tools
1. Gather Domain Knowledge
2. Identify Relevant Social Privileges
3. Apply Social Engineering Patterns
4. Construct the Combined Attack Graph
5. Analyze the Combined Attack Graph
6. Mitigate Possible Threats
Extended Access Control Lists
Lists of Subjects, Objects, and Operations
Privilege List of Social Engineering Vulnerabilities
Scenario-specific Social Engineering Actions
Attack Graph considering Social Engineering, Network and Software Vulnerabilities
Prioritized Attack Paths
A Set of Methods and Tools to Apply

Fig. 2. An overview of our method for integrating social engineering threats in attack graphs

4.1 Step 1. Gather Domain Knowledge

Input: access control lists and information about communication channels.

Output: extended access control lists, lists of subjects, objects, allowed operations.

The first step in our methodology is gathering knowledge about the system. In particular, we will need access control information for different domains, existing communication channels (e.g., e-mail service, phone, skype, etc.), information about physical access and segregation of networks. Our goal is to create a model of the system.

Let S be a set of all subjects in the system plus an attacker att, O be a set of all objects, and OP be a set of all possible operations over objects. This information can be retrieved from existing access control lists used in the considered organisation. In the sequel, we consider only three types of operations: (R)ead, (W)rite, and (M)odify, but this list can be extended if needed. The real meaning of these operations depends on the type of the object. In this paper, we use "R" when a user may only read information (e.g., displayed on the computer), "W" when the user may interact with the object (e.g., execute the installed programs), and "M" when the user may modify the behaviour of the object (e.g., install programs on the computer).

Next, we need all access control lists themselves for different domains used in the considered organisation. By an access control list ACL we mean a set of access control rules expressed as triples $acl = (s, o, op)$, $s \in S$, $o \in O$, $op \in OP$, and $acl \in ACL$. Every access control list specifies access to a specific domain, i.e., the guarded set of objects. Therefore, we consider a set of such lists $\mathbb{ACL}$. We assume that $\mathbb{ACL}$ could be derived even if such systems as RBAC or ABAC are used in the system.

We sometimes need specific classes of subject and object. In other words, a class is a set of subjects or objects. Let CL be a set of classes $cl \in CL$ defined for the system, where domain of cl is $S \times O$. For example, we may have a class of administrators, class of workstations or class of network channels. The meaning of the required classes is specified by the social engineering exploits to be considered. We also assume that system analysts are able to specify which subjects and objects are contained in these classes using supporting information (e.g., business model, network model, etc.).

We need information about existing communication and physical channels. A channel is any means of human interaction used in the system. Let CH be a set of all possible channels in the system. We consider channels as objects, from which the users may receive messages ("R"), write messages to ("W"), and modify the channel ("M").

We also extend the notion of channels with channels for different networks. Social engineering attacks often lead to acquisitions of knowledge required for accessing some part of the system (e.g., login and password pair). On the other hand, next to the knowledge the attacker needs the possibility to use the knowledge (e.g., access to the internal network). Only having the knowledge *and* the possibility the attacker is able to execute her attack. Since, such channels are established between parts of the system (e.g., some objects) then we extend the S set with such objects.

Example 1. In our running example we consider two channels: e-mail ch^m and physical access ch^{phy}. Physical access channel refers to physical access to different parts of the system. In our example, we have only one unguarded building, i.e., there is only one physical channel in the system. Next to these social channels we have two network channels: internal LAN access ch^{lan}, and the Internet access ch^{int}. The internal access channel refers to the access to LAN between the workstation, laptop and server, when external access channel is available between the workstation, laptop and the Internet.

For specifying $\mathbb{ACL}$ we need a set of possible users $S = \{user, adm, att, ws, lap, serv, db\}$ and a set of objects is $O = \{ws, lap, serv, db, ch^m, ch^{phy}, ch^{int}, ch^{lan}\}$. Please, note, that the same entity can be both in the set of users and objects. This happens because the entity can either be accessed by another object or access another object itself depending on a scenario. A set of operations are: $OP = \{R, W, M\}$. Moreover, we need a class of computers $cl^{comp} = \{ws, lap, serv\}$, a class of network channels $cl^{net} = \{ch^{int}, ch^{lan}\}$, a class of separate physical areas: $cl^b = \{ch^{phy}\}$ and a class of e-mail channels $cl^m = \{ch^m\}$. In general, $CL = \{cl^{comp}, cl^{net}, cl^b, cl^m\}$.

Table 2 shows the access control lists that exist in the system. In fact, we have 4 access control lists for accessing the workstation, laptop, database and server. We also have access control lists for mail and physical channels and two lists for network access channels. We combine some access control rules by writing operations separated by "/".

There are a couple of simplifications in the model. First, the access to the e-mail channel can also be done through a client software installed on the user's computer. Access to this client is not guarded (as the login and password are stored by the client) and, thus, the only guard in this case is the access control mechanism of the workstation. Also the database is considered as having access

Table 2. Access control lists for our running example.

ACL^{ws} :	ACL^{lap} :	ACL^{db} :	ACL^{serv} :
$(user, ws, R/W/M)$	$(adm, lap, R/W/M)$	$(user, bd, R/W)$	$(adm, serv, R/W/M)$
$(adm, ws, R/W/M)$	$(adm, ch^{m}, R/W)$	$(adm, bd, R/W/M)$	$(adm, ch^{lan}, R/W)$
$(user, ch^{m}, R/W)$	$(adm, ch^{lan}, R/W)$	$(adm, ch^{lan}, R/W)$	
$(user, ch^{lan}, R/W)$	$(adm, ch^{int}, R/W)$	$(user, ch^{lan}, R/W)$	
$(user, ch^{int}, R/W)$			
$(adm, ch^{lan}, R/W)$			
$(adm, ch^{int}, R/W)$			
ACL^{lan} :	ACL^{int} :	ACL^{m} :	ACL^{phy} :
$(ws, ch^{lan}, R/W)$	$(ws, ch^{int}, R/W)$	$(adm, ch^{m}, R/W)$	$(user, ch^{phy}, R/W)$
$(lap, ch^{lan}, R/W)$	$(lap, ch^{int}, R/W)$	$(user, ch^{m}, R/W)$	$(adm, ch^{phy}, R/W)$
$(serv, ch^{lan}, R/W)$			
$(bd, ch^{lan}, R/W)$			

to the internal network. In fact, it accesses the network only though the server, but we skip these details for the sake of simplicity. We are going to work on a more strict model in the future.

4.2 Step 2. Identify Relevant Social Privileges

Input: extended access control lists, lists of subjects, objects, allowed operations.

Output: privilege list for social engineering vulnerabilities.

We define all privileges in a similar way as access control rules. Although, the way of defining access control rules and privileges are similar, their semantics is slightly different. In contrast to network attacks, for modelling social engineering attacks it is not enough for the attacker simply to receive credentials for accessing an object. The attacker needs also to have a channel to reach an object before she is able to use it. Thus, we would like to underline that *a privilege is a combination of credentials and an existing channel.* For example, assume, that an attacker was able to get credentials for a computer of an employee by deceiving him. An attacker needs a channel connecting her computer and the computer of the user to be able to access it (or have physical access).

Let P be a set of all possible privileges in a system and $\mathbb{P}(P)$ be a powerset of this set. In this work, we will split the whole set of privileges considered for the system into two sets: $P^{N} \cup P^{SE} = P$, where P^{N} is a set of privileges used for usual network attacks and P^{SE} are specific privileges for social engineering attacks. Although, similar ways of defining elements of these sets is not strictly required for consistency of the proposed approach, we assume that the privileges look similar for simplicity. Here, we focus on P^{SE} and only assume that some intersection between the sets can be established.

All possible privileges can be seen as a triple $\mathbb{P}(P^{SE}) = \{att\} \times O \times OP$ and every privilege $p \in P^{SE} \subseteq \mathbb{P}(P^{SE})$ can be seen as $p = (att, o, op)$. We assume, that an attacker, in theory, is able to get any privilege existing in the system to define P^{SE}:

$$P^{SE} = \{(att, o, op) | \exists s \in S, (s, o, op) \in ACL, ACL \in \mathbb{ACL}\} \tag{1}$$

Next to the possible privileges of an attacker (i.e., P^{SE}) we also need privileges of the system (P^{SYS}). We assume, that all access control rules have corresponding channels for realising the access for the users. This is a valid assumption, because otherwise the system is not configured correctly. P^{SYS} are needed only for the specification of applicable social engineering attacks and do not intersect with P. Therefore, we will ignore them when all social engineering exploits for a concrete system are identified.

The privileges for the system can be received by aggregating all entries of $\mathbb{ACL}$:

$$P^{SYS} = \{(s, o, op) | \exists ACL, (s, o, op) \in ACL, ACL \in \mathbb{ACL}\} \tag{2}$$

Example 2. In our running example the P^{SE} is:

$$\begin{aligned} P^{SE} = \{ & (att, ws, R/W/M), (att, lap, R/W/M), (att, serv, R/W/M), \\ & (att, sb, R/W/M), (att, ch^{int}, R/W), (att, ch^{lan}, R/W), \\ & (att, ch^{m}, R/W), (att, ch^{phy}, R/W)\} \end{aligned} \tag{3}$$

4.3 Step 3. Apply Social Engineering Patterns

Input: social engineering patterns (techniques), access control lists, sets of privileges.

Output: a set of scenario specific social engineering actions.

Now we need to find the set of the exploits available for an attacker. In order to execute an exploit an attacker has to have some initial privileges (for example, access to a computer via LAN). Successful execution of a vulnerability provides the attacker with some additional privileges, e.g., root privileges on the targeted node. The sets of initial and ending privileges are determined on the basis of system configuration and identified vulnerabilities.

Let every action of an attacker $a \in Act$ be a single exploit of a vulnerability where Act is a set of possible actions for this system. Then, we may see actions as transitions from one set of privileges to a wider set:

$$Act \subseteq \mathbb{P}(P) \times \mathbb{P}(P) \ \wedge \ \forall\ a = (P^b, P^e) \in Act\ .\ P^b \subset P^e \tag{4}$$

where P^b is a minimal set of privileges required to perform the action and P^e is the resulting set of privileges. We also use two special functions: fst and snd, which return the first and the second element of a Cartesian product, i.e., $fst\ a = P^b$ and $snd\ a = P^e$ for $a = (P^b, P^e)$. Please, note that we make the usual assumption for attack graphs that privileges once gained remain until the end of an attack [17].

The set of network exploits is usually defined using different network scanning tools (e.g., Nessus, OpenVAS, etc.). The result of a scan is a set of vulnerabilities. These vulnerabilities are supported by pre- and postconditions, which help to identify the list of network actions for an attacker Act^N. We do not consider Act^N in this article, since this has already been done before (e.g., [21]) and focus on social engineering exploits Act^{SE}, $Act = Act^N \cup Act^{SE}$.

We propose to define social engineering patterns and apply them to the considered system in order to find the set of social engineering exploits. The social engineering patterns can be defined in the following way:

Pattern name	$< Name >$
Free Variables	$< List\ of\ variables >$
Pre-conditions	$P^b \cup P^s\ .\ P^b \subseteq P^{SE}\ \wedge\ \cup P^s \in Privs^{SYS}\ \wedge$
	$< Constraints\ B >$
Post-conditions	$P^e = P^b \cup P'\ .\ P' \in P^{SE} \bigwedge$
	$< Constraints\ E >$

The *constraints* B and E are the boolean logical expression, which can be written with any logic suitable for expressing constraints. In this work we use the first order logic. In the logical constraint we use only the finite sets defined earlier S, O, OP, $\mathbb{ACL}$, CL and free variables specified in the beginning. Free variables are the members of these sets and are unique for the whole pattern. A usual example for social engineering patterns is the subject the attacker decides to deceive. The free variables will be instantiated when a pattern is applied to a concrete system.

The *constraint* B restricts free variables and defines the initial set P^b of privileges the attacker must have to start the attack and the required set P^s of privileges the system must have. The *constraint* E does the same for a set P' of privileges gained by the attacker after successful execution of the attack.

When a pattern is applied to a system it is required to find all such combinations of free variables which satisfy the specified initial constraints. For construction of attack graph we will need only the privileges of attacker (since, the attack graph describes the evolution of attacker's privileges). Thus, privileges of the system are needed only to check whether the pattern is applicable. Every single set of privileges found applicable for the system uniquely defines an action (and its post conditions) in a set Act^{SE}.

Example 3. First, lets consider the baiting attack type as an example pattern. In this pattern an attacker needs to leave a flash key near the working place of the user and wait when the user will try to read it and a special program will install a key logger granting access privileges to the attacker. Thus, the initial conditions are (1) a physical access to the building, where the user is working, (2) the ability of the user to run a program on a computer, and (3) existence of a network channel between the attacker and the targeted computer of the user. We use $\bigwedge$ to highlight the separation of the requirements. The result is the access to all privileges of the user within the compromised domain.

Pattern name	$Baiting: \ SEE-05$
Free variables	$\boldsymbol{s_1} \in S, \boldsymbol{o_1} \in cl^{comp}, \boldsymbol{o_2} \in cl^{b}, \boldsymbol{o_3} \in cl^{net}$
Pre-conditions	$P^b \cup P^s \ . \ P^b \subseteq P^{SE} \ \wedge \ P^s \subseteq P^{SYS} \ \bigwedge$
	$P^b = \{(att, \boldsymbol{o_2}, op_1), (att, \boldsymbol{o_3}, op_3)\} \ \bigwedge$ $P^s = \{(\boldsymbol{s_1}, \boldsymbol{o_2}, op_2), (\boldsymbol{s_1}, \boldsymbol{o_1}, op_4), (\boldsymbol{o_1}, \boldsymbol{o_3}, op_5)\} \ \bigwedge$ 1) $op_1 = W \ \wedge \ op_2 = W \ \bigwedge$ 2) $op_4 = M \ \bigwedge$ 3) $op_3 = W \ \wedge \ op_5 = R$
Post-conditions	$P^e = P^b \cup P' \ . \ P' \in P^{SE} \ \bigwedge$
	$P' = \{(att, o', op') \mid \exists ACL' \in \mathbb{ACL} \ \wedge \ (\boldsymbol{s_1}, \boldsymbol{o_1}, op) \in ACL' \ \wedge$ $op = M \ \wedge \ (\boldsymbol{s_1}, o', op') \in ACL' \ \}$

Note, that the privileges received by the attacker are limited by the privileges of the user s in the access control list ACL'. This is because once the attacker (by)passed the access control check she is able to access any object, which the deceived user can access within this access control domain.

Example 4. For instantiation of the baiting pattern $Baiting(s, o_1, o_2, o_3)$ we need to find possible free variables, satisfying the *constraint* B specified in earlier examples. For subject s_1=`user`, o_1 could be only `ws`, since `user` has execution access right only on `ws`. In the system we have only one channel o_2 of type building - ch^{phy}. There are two possible networks (o_3) through which the attacker may access the workstation `ws`: ch^{lan} (through a previously compromised `lap` or `serv`) and ch^{int} (directly from the Internet). Then, the baiting pattern for the `user` can be instantiated as the following possible actions of the attacker:

Pre-conditions	$\{(att, ch^{phy}, W), (att, ch^{int}, W),$ ~~$(user, ws, M), (user, ws, M), (ws, ch^{int}, M)$~~$\}$
Post-conditions	$\{(att, ch^{phy}, W), (att, ch^{int}, W), (att, ws, M), (att, ch^{m}, R/W), (att, ch^{lan}, R/W)\}$

Pre-conditions	$\{(att, ch^{phy}, W), (att, ch^{lan}, W),$ ~~$(user, ws, M), (user, ws, M), (ws, ch^{lan}, M)$~~$\}$
Post-conditions	$\{att, ch^{phy}, W), (att, ch^{lan}, W), (att, ws, M), (att, ch^{m}, R/W), (att, ch^{int}, R/W)\}$

The administrator in our example has modify access rights on all 3 computers in the system. Thus, we get 5 additional instances of this pattern for the administrator `adm` (i.e., 3 instances for ch^{lan} and 2 for ch^{int}), which are formed using the same strategy.

4.4 Step 4. Construct the Combined Attack Graph

Input: set of technical actions, set of social engineering actions.

Output: an attack graph considering social engineering and technical attackers.

In this section we briefly recall how an attack graph could be formally defined and constructed. More details could be found in our previous work [8].

For the construction we need a set of privileges P and a set of actions Act. Both these sets consist of network and social engineering privileges or actions, but the construction of the graph does not depend on origin of privileges and actions.

Definition 1. *Let P be a set of all possible privileges and Act be a set of all possible attacker actions relevant for the system. Then, the attack graph $\mathcal{G} \subseteq \mathbb{P}(\mathbb{P}(P) \times Act \times \mathbb{P}(P))$ associated to P and Act is defined as follows:*

$$\begin{aligned} G := \{(P^b, a, P^e) \in \mathbb{P}(P) \times Act \times \mathbb{P}(P) |\\ 1)\ fst\ a \subseteq P^b;\ 2)\ P^e = P^b \cup snd\ a;\ 3)\ snd\ a \setminus fst\ a \not\subseteq P^b\} \end{aligned} \tag{5}$$

In words, the attack graph is defined as a set of edges, which relate to actions and allow an attacker to move from one set of privileges to a wider set. A vertex in the attack graph is defined by a set of privileges. The attack graph defined in Definition 1 is a directed acyclic graph (DAG).

The graph specifies all possible attacks on the system. In most cases, the administrators and security staff are interested in protection against a specific type of attacker (e.g., an outsider) which would like to achieve some goal (e.g., get access to the database). Formally, this means that we should consider only a part of the graph which is formed by all paths from a vertex with initial privileges of the attacker to the vertices containing goal privileges.

Example 5. The analyst in our example would like to consider how an outsider can get access to the data of the database. This attacker initially has access to the Internet and physical access to the premises of the organisation. Thus, the attacker starts with the set of privileges $P^0 = \{(att, ch^{int}, W), (att, ch^{phy}, W)\}$ and would like to get the privilege: $P^f = \{(att, db, W)\}$.

In order to construct the attack graph we need a set of combinations of privileges (states) and a set of available actions (transitions). An attack graph usually is a huge, interconnected structure. Here *we concentrate only on a part of this graph* to exemplify the effect of using social engineering exploits on the combined attack graph. In other words, we consider only a subgraph formed by some paths from the initial set of privileges to the sets which contain the desired privilege.

For the construction of the subgraph we need the combinations of privileges shown in Table 3 and a set of actions listed in Table 4. The resulting subgraph is shown in Fig. 3.

Table 3. Privileges corresponding to the vertices of the considered subgraph.

Node	Relevant combinations of privileges
v_0	$\{(att, ch^{int}, W/R), (att, ch^{phy}, W/R)\}$
v_1	$v_0 \cup \{(att, ws, M/W/R), (att, ch^{lan}, W/R), (att, ch^{m}, W/R)\}$
v_2	$v_1 \cup \{(att, ch^{m}, W/R), (att, serv, M/W/R)\}$
v_3	$v_1 \cup \{(att, db, W/R)\}$
v_4	$v_2 \cup \{(att, db, M/W/R)\}$

Table 4. Actions available for the attacker.

CVE code	Target	Initial privileges	Resulting privileges
CVE-2011-3108	Chrome	$\{(att, ch^{int}, W/R)\}$	$\{(att, ws, M/W/R), (att, ch^{m}, W/R), (att, ch^{lan}, W/R), (att, ch^{int}, W/R)\}$
CVE-2011-4862	FreeBSD	$\{(att, ws, M/W/R), (att, ch^{lan}, W/R)\}$	$\{(att, ws, M/W/R), (att, ch^{lan}, W/R), (att, serv, M/W/R)\}$
CVE-2012-0114	MySQL	$\{(att, serv, M/W/R)\}$	$\{(att, serv, M/W/R), \{(att, db, M/W/R)\}$
SEE-04	Employee	$\{(att, ws, M/W/R), (att, ch^{m}, W/R)\}$	$\{(att, ws, M/W/R), (att, ch^{m}, W/R), (att, db, W/R)\}$
SEE-05	Employee	$\{(att, ch^{int}, W/R), (att, ch^{phy}, W/R)\}$	$\{(att, ch^{int}, W/R), (att, ws, M/W/R), (att, ch^{lan}, W/R), (att, ch^{m}, W/R)\}$

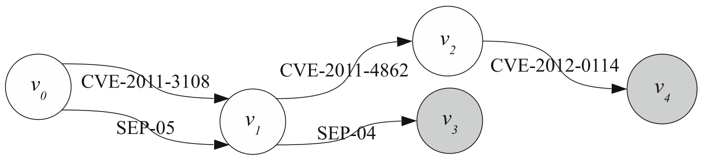

Fig. 3. A subgraph with social engineering and network exploits.

In Fig. 3 there are four possible paths from initial v_0 to one of the combinations of privileges which contain the desired privilege v_3 or v_4.

$$\begin{aligned}\pi_1 =& \langle \text{CVE-2011-3108}, \text{CVE-2011-4862}, \text{CVE-2011-0114} \rangle, \\ \pi_2 =& \langle \text{CVE-2011-3108}, \text{SEE-04} \rangle, \\ \pi_3 =& \langle \text{SEE-05}, \text{SEE-04} \rangle, \\ \pi_4 =& \langle \text{SEE-05}, \text{CVE-2011-4862}, \text{CVE-2011-0114} \rangle\end{aligned} \tag{6}$$

Path π_1 is purely technical. First, a vulnerability in Chrome browser `CVE-2011-3108` is exploited remotely, then the attacker exploits the vulnerability in the server via LAN (`CVE-2011-4862`) and, finally, compromises the database (`CVE-2012-0114`). Path π_3 consists only of social engineering attacks. The attacker uses baiting (`SEP-05`) throwing a USB stick in the building of the targeted company. The employee plugs in the stick to the workstation thus the attacker obtains access to the the workstation, LAN of the company and the e-mail of the employee. Then the attacker applies reverse social engineering (SEP-04) disabling the access to the database from the workstation and asking for the credentials of the user by e-mail on the behalf of the administrator "to solve the issue". There are also two hybrid paths (π_2 and π_4) for the attacker which contain mixed sets of used exploits.

4.5 Step 5. Analyse the Combined Attack Graph

Input: attack graph, quantitative input parameters.

Output: prioritised attack paths.

Social engineering attacks do not require much technical knowledge and may be executed by an inexperienced attacker. Moreover, many organisations pay much attention to hardening the technical security of their network underestimating the danger of social attacks [5]. On the other hand, social engineering attacks often require a direct contact with people working in the organisation and physical penetration into the premises of the organisation. Such actions rise the risk of detection and the risk to be caught.

Let π be a path in an attack graph G. We can see this path as a sequence of $(P_i^b, a_i, P_i^e) = x_i,\ i = \{1, 2, ..., n\}$, such that P_1^b is equal to the initial sets of privileges and P_n^e contains the goal privileges of the attacker. Let the probability of successful execution of an action a_i be $\mathbf{pr}_i^e$ and the probability to be caught be $\mathbf{pr}_i^c$. In general, it is far not every time the attacker is caught, when an attack fails, therefore $\mathbf{pr}_i^c \leq 1 - \mathbf{pr}_i^e$. Let also the benefit the attacker aims to get after successful execution of the attack be c_{succ}, when the loss in case of capture (e.g., cost of the lawyers or a fine) is c_{loss}. Then, we may identify the path, which is more profitable for the attacker:

$$Benefit = c_{succ} \times \prod_{i=1}^{n} \mathbf{pr}_i^e; Loss = c_{loss} \times (1 - \prod_{i=1}^{n}(1 - \mathbf{pr}_i^c)); \qquad (7)$$
$$Profit = Benefit - Loss$$

Example 6. The probabilities of the successful execution of actions and the probabilities for the attacker to be caught are in Table 5. Suppose also in case of successful attack attacker obtains $c_{succ} = 10$ (thousands of \$) and in case the attacker fails her loss is $c_{loss} = 20$ (thousands of \$). Profits of the attacker for paths are correspondingly $Profit(\pi_1) = -1.5$, $Profit(\pi_2) = 2.1$, $Profit(\pi_3) = -2.4$, and $Profit(\pi_4) = -4.56$. We see, that nevertheless, the pure social engineering attack (π_2) is relatively beneficial for the attacker (3.2) the potential probability to be caught is high and so are the potential loss (5.7). Thus, the most dangerous path according to the analysis is π_2 and the analyst should put high priority to mitigate this attack.

Table 5. Probabilities of success and possible capture for the considered actions.

a	$\mathbf{pr}^e$	$\mathbf{pr}^c$
CVE-2011-3108	0.70	0.08
CVE-2011-4862	0.60	0.05
CVE-2011-0114	0.50	0.06
SEE-05	0.40	0.20
SEE-04	0.80	0.10

4.6 Step 6. Mitigate Possible Threats

Input: prioritised attack paths, set of mitigation methods/tools.

Output: a set of methods/tools to apply.

We propose to establish the following countermeasures for social engineering attacks, as introduced by Mitnick [5]. The first action should be (i) to design clear and easily understandable security policies. Strict constraints on length and technical language have to be imposed on these policies to achieve this goal. The second action is (ii) data classification, e.g., in the four basic categories: confidential, private, internal and public. This distinction clearly defines data that employees should guard carefully and get suspicious if someone aims to get access them. Then the administrator should establish (iii) verification and authorisation procedures that include the use of caller ID, callback, shared secret, etc. to prevent unauthorised access to confidential, private, or internal data. (iv) Simple and repeated security awareness training with seminars and reminders, e.g., via messages on screensavers is also a useful countermeasure. The fifth action is (v) penetration testing for social engineering attacks and a reward system for employees that prevent the attacks. Finally, (vi) an alert system for social engineering attacks where employees can report possible attacks.

Example 7. The most important attack to be mitigated is the reverse social engineering (`SEE-05`), since it belongs to π_3. Security awareness training should provide the `user` with the ability to get suspicious and trigger a security check for the requesting person.

5 Conclusions

We contributed a structured threat analysis method for combining the analysis of social engineering attackers and technical attackers (e.g., network attackers) using attack graphs. In contrast to current research in attack graph analysis, which focuses solely on technical attackers. Our methodology relies on access control lists and communication diagrams of a company to identify relevant actors that can become victims of a social engineering attacker. We have shown how social engineering patterns could be defined and instantiated for a specific organisation. Once the model of a system (i.e., access control lists, sets of objects and subjects and required classes) is defined, the actions required for attack graph construction and the construction itself can be done (semi)automatically. Finally, the proposed quantitative analysis specifies not only the most beneficial path for the attacker, but also takes into account the possibility of the attacker to be caught.

One direction for the future work is a more rigorous specification of the system model. Currently, we rely a lot on an analyst that aggregates the existing information and models the system. We will also consider not only access control lists and communication channels, but also organisational charts that illustrate the hierarchies in companies, detailed network topologies that illustrate the flow of digital information, and system architecture diagrams that explain the relation of all information.

Acknowledgements. This paper was partially supported by ARTEMIS Joint Undertaking SESAMO project, POR-CREO 2007-2013 Secure! regional project, PRIN Security Horizons project and the Ministry of Innovation, Science, Research and Technology of the German State of North Rhine-Westphalia and EFRE (Grant No. 300266902).

References

1. Peltier, T.R.: Social engineering: concepts and solutions. Inf. Syst. Secur. **15**(5), 13–21 (2006)
2. Algarni, A., Xu, Y., Chan, T., Tian, Y.C.: Social engineering in social networking sites: Affect-based model. In: Proceedings of the 8th International Conference on Internet Technology and Secured Transactions, pp. 508–515. IEEE (2013)
3. Dimkov, T., van Cleeff, A., Pieters, W., Hartel, P.: Two methodologies for physical penetration testing using social engineering. In: Proceedings of the 26th Annual Computer Security Applications Conference, pp. 399–408. ACM (2010)
4. Laribee, L., Barnes, D., Rowe, N., Martell, C.: Analysis and defensive tools for social-engineering attacks on computer systems. In: Proceedings of the Information Assurance Workshop, pp. 388–389. IEEE (2006)
5. Mitnick, K.D., Simon, W.L.: The Art of Deception. Wiley, Indianapolis, Indiana (2009)
6. Krombholz, K., Hobel, H., Huber, M., Weippl, E.: Social engineering attacks on the knowledge worker. In: Proceedings of the 6th International Conference on Security of Information and Networks, pp. 28–35. ACM (2013)
7. Kvedar, D., Nettis, M., Fulton, S.P.: The use of formal social engineering techniques to identify weaknesses during a computer vulnerability competition. J. Comput. Sci. Coll. **26**(2), 80–87 (2010)
8. Beckers, K., Heisel, M., Krautsevich, L., Maritnelli, F., Yautsiukhin, A.: Considering attacker motivation in attack graphs analysis in a smart grid scenario. In: Proceedings of the Second Open EIT ICT Labs Workshop on Smart Grid Security, Springer (2014, To appear)
9. Ahmadi, N., Jazayeri, M., Lelli, F., Nesic, S.: A survey of social software engineering. In: Workshop Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering, pp. 1–12. IEEE (2008)

10. Mills, D.: Analysis of a social engineering threat to information security exacerbated by vulnerabilities exposed through the inherent nature of social networking websites. In: Proceedings of the Information Security Curriculum Development Conference, pp. 139–141. ACM (2009)
11. Chitrey, A., Singh, D., Singh, V.: A comprehensive study of social engineering based attacks in india to develop a conceptual model. Int. J. Inf. Netw. Secur. **1**(2), 45–53 (2012)
12. Winkler, I.S., Dealy, B.: Information security technology?...don't rely on it: a case study in social engineering. In: Proceedings of the 5th Conference on USENIX UNIX Security Symposium, p. 1–1. USENIX Association (1995)
13. Gonzalez, J.J., Seasick, A.: A framework for human factors in information security. In: Proceedings of WSEAS International Conference on Information Security (2002)
14. Twitchell, D.P.: Social engineering in information assurance curricula. In: Proceedings of the 3rd Annual Conference on Information Security Curriculum Development, pp. 191–193. ACM (2006)
15. International Organization for Standardization (ISO), International Electrotechnical Commission (IEC): Information technology - Security techniques - Information security management systems - Requirements. ISO/IEC 27001 (2013)
16. BSI: Grundschutzhandbuch IT. Bundesamt für Sicherheit in der Informationstechnik (BSI) (2007). http://www.bsi.bund.de/gshb/index.htm
17. Jha, S., Sheyner, O., Wing, J.: Two formal analyses of attack graphs. In: Proceedings of the Computer Society Security Foundations Workshop. IEEE (2002)
18. Krautsevich, L., Martinelli, F., Yautsiukhin, A.: Towards modelling adaptive attacker's behaviour. In: Garcia-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N., Miri, A., Tawbi, N. (eds.) Foundations and Practice of Security. LNCS, vol. 7743, pp. 357–364. Springer, Heidelberg (2013)
19. LeMay, E., Ford, M.D., Keefe, K., Sanders, W.H., Muehrcke, C.: Model-based security metrics using adversary view security evaluation (advise). In: Proceedings of the 8th International Conference on Quantitative Evaluation of SysTems, pp. 191–200. IEEE (2011)
20. Noel, S., Jajodia, S.: Managing attack graph complexity through visual hierarchical aggregation. In: Proceedings of the Workshop on Visualization and Data Mining for Computer Security. ACM (2004)
21. Sheyner, O., Haines, J., Jha, S., Lippmann, R., Wing, J.M.: Automated generation and analysis of attack graphs. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy. IEEE (2002)