Representing Humans in System Security Models: An Actor-Network Approach

Wolter Pieters
University of Twente
Enschede, The Netherlands
w.pieters@utwente.nl

Abstract

System models to assess the vulnerability of information systems to security threats typically represent a physical infrastructure (buildings) and a digital infrastructure (computers and networks), in combination with an attacker traversing the system while acquiring credentials. Other humans are generally not included, as their behaviour is considered more difficult to express. We propose a graph-based reference model for reasoning about access in system models including human actions, inspired by the sociological actor-network theory, treating humans and non-humans symmetrically. This means that humans can employ things to gain access (an attacker gains access to a room by means of a key), but things can also employ humans to gain access (a USB stick gains access to a computer by means of an employee), leading to a simple but expressive model. The model has the additional advantage that it is not based on containment, an increasingly problematic notion in the age of disappearing boundaries between systems. Based on the reference model, we discuss algorithms for finding attacks, as well as examples. The reference model can serve as a starting point for discussing representations of human behaviour in system models, and for including human behaviour in other than graph-based approaches.

Keywords: actor-network theory, containment, hypergraphs, security modelling, socio-technical systems, vulnerability analysis

1 Introduction

The prevalence of the insider threat problem teaches us an important lesson: security is not so much a property of technical systems, but a property of so-called socio-technical or cyber-physical-social systems [1]. Human behaviour needs to be included in assessments of insider threat, referring both to the abilities of the attacker as well as the weaknesses of others in enforcing security policies (e.g. in the case of social engineering). Two questions can be asked in this context: (1) do indications for insider activity exist, and (2) if there were an insider, what could s/he do? The former corresponds to the notion of threat in the formula *risk* = *threat* $\times$ *vulnerability* $\times$ *impact*; the latter corresponds to vulnerability. This paper focuses on the second question, and addresses models for evaluating the potential damage done by insider threat, by assessing the vulnerability of socio-technical systems. For this reason, we do not discuss the motivations or intentions of the attacker, as this is part of the first question.

It has been argued that including human properties in vulnerability models is an important direction in the prevention of insider threat [2]. This paper provides foundational insights for such a direction, by putting forward a model based on the simplest possible assumptions on what constitutes human behaviour in a security context. Inspired by the actor-network theory of Bruno Latour, we develop a graph-based system model for vulnerability analysis in which humans and non-humans are treated fully symmetrically. Physical topology is no longer the basis, but rather a topology defined in terms of access relations. The main message is that it is not necessarily required to distinguish between actors, objects and credentials *a priori*. Indeed, as we will see later on, humans can take the role of credential. This basic model can serve as a starting point for discussing to what extent this simple representation of human behaviour

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, volume: 2, number: 1, pp. 75-92

is sufficient, and which extensions would be necessary. It can also serve as a reference for including human behaviour in more advanced vulnerability models complementing graph-based representations, such as those based on process algebras [3]. The aim is not to replace existing models, but rather to provide inspiration for extending these with human action, to address among other things insider threats.

The contributions of this paper are mainly theoretical. These are ideas that emerged during our work on integrating physical, digital and social aspects in security modelling and testing. Although we discuss algorithms and examples, we do not present a full-blown system, but rather considerations and directions for future development of security modelling. Readers interested in the more practical aspects are referred to [4] for security modelling and [5] for security testing.

In Section 2, we give a more detailed motivation of the research approach and discuss related work. In Section 3, we introduce the sociological framework of actor-network theory and investigate how this theory can be applied to include human actions in vulnerability analysis. In Section4, we present a formalisation of the model for describing security properties, called Actor-NetworK Hypergraphs (ANKH). Section 5 presents an example, and Section 6 discusses algorithms for vulnerability analysis.

2 Motivation

2.1 Human Action in System Models

To make our contribution more precise, let us first specify the security modelling background. A distinction can be made in this context between protocol models and system models. Protocol models aim at verifying protocols against the properties they aim to achieve, in the presence of an attacker (see e.g. [6, 7]). The focus is on whether the specified interaction can be maliciously changed in order to invalidate the supposedly guaranteed properties. Conversely, system models aim at checking whether a system can be compromised by incrementally increasing the objects that the attacker has access to. This typically involves the acquisition of credentials in an IT and/or physical infrastructure. The focus here is on maliciously acquiring access rather than maliciously changing interactions. The integration of these approaches remains an important research challenge. In the meantime, the present research is embedded in the system model paradigm, and therefore addresses attacks that involve incremental access acquisition in specified infrastructures.

Traditional system models for vulnerability analysis focus on a network of interconnected computer systems, which an attacker can traverse. The model of the infrastructure is typically a static tree of containment relations between objects, if necessary augmented with additional (network) connections. The attacker then traverses the infrastructure by obtaining credentials and thereby gaining additional access to objects, which can again contain credentials, *et cetera*. An attack is successful when the attacker in the end has access to some designated asset.

Such models are not sufficient when (1) other things than the attacker exhibit mobility properties, and (2) organisational and social aspects (interaction) can occur [8]. Recently, vulnerability models have been proposed that include the mobility aspect [4, 3]. These models allow extension of vulnerability analysis from digital to physical contexts, as the moving around of actors and items in buildings can now be expressed. In principle, such models can also be used to express interaction, but they do not provide explicit arguments for why human actors have been included in a specific way. This means that either distinctions are proposed between actors (moving humans) and non-actors (credentials) [3], or a construct of delegation is added, representing the possibility that other agents could be fooled into doing something for the attacker [4].

To assess the value of such approaches, research needs to address explicitly the question to what extent these extensions are necessary and sufficient to include human actions in vulnerability models. We therefore go back to the basics, and ask which assumptions are necessary to include human behaviour. In

particular, how much of a distinction do we need to make between the behaviour of human agents and the behaviour of what can be called non-human agents, *from the perspective of acquisition of access*? This does not mean that existing models are inadequate in terms of effectiveness in finding attacks, but rather that we want to put the question how to include human behaviour on the agenda explicitly. In particular, if a simpler way of doing so is possible, it would be recommendable for reasons of minimality.

Different approaches to modelling human behaviour are possible. One could let models of humans make rational decisions optimising value, or let them aim for achieving certain goals with their actions. However, in vulnerability analysis, we are mostly interested in whether it is *possible* for humans to acquire *access* to certain assets. How likely it is that they will indeed aim at acquiring this access is based on their goals, and these can be said to be external to the system model. Only when the system being modelled can be said to influence the goals of the modelled humans, then this aspect needs to be included (cf. [9, 10]). In that case, changes in the system can change the goals, thereby making the actors behave differently. Here, we focus only on assessing the possibilities of access, given that someone will indeed try to acquire it.

An easy way to add human action to system models would be to increase the number of attackers. As we are not so much interested in whether actions are due to malicious intent or merely lack of awareness, we could then assume collusion of the different actors in trying to gain access to the target asset, by means of acquiring credentials and sharing them. An attacker may then be able to obtain a credential previously fetched by another employee, but other actions beneficial to the attacker, such as leaving a door open or plugging a found USB stick into a computer, cannot be expressed.

Our proposal is to define action in system models purely in terms of acquisition of access, *but not only access to credentials*. Whether an attacker acquires access to a credential, an employee picks up a USB stick, or a USB stick acquires access to a computer, these can all be expressed as changes in access relations. It is then not needed to distinguish attackers, other humans, objects and credentials. One could imagine that this might easily fit into existing tree-based infrastructure models, with the only difference that nodes in the tree can now move, not just an attacker external to the infrastructure model.

2.2 Limitations of Containment-Based Models

However, the containment approach is itself subject to challenges, due to development in connectivity. Traditionally, protection of information has been directed at securing an organisation at the perimeter of its network, typically in the form of a firewall. Due to changes in technologies, business processes and their legal environments this approach has become problematic. Many organisations are outsourcing part of their IT processes, and employees demand that they can work from home. Mobile devices can access data from anywhere, smart buildings are equipped with small microchips, and with cloud computing, organisations can rent virtual PCs by the hour. Such systems cross the security perimeters that parties have put in place for themselves. Following the Jericho Forum [11], we call this process *de-perimeterisation*. Human behaviour also plays a role here, as employees move into and out of the organisations boundaries all the time.

Interestingly, techniques for modelling information security in organisations often focus precisely on the physical metaphor of containment as the basic principle (see e.g. [12, 13, 14, 15, 16]). This may be explained by their derivation from the traditional (physical) security paradigm, with physical security measures such as buildings and safes. However, even for physical access containment may not be sufficient. Assume that a building consists of a hallway and two rooms, where both rooms are connected to the hallway and to each other by doors (Fig. 1). Following the containment-based approach to security modelling, the rooms are clearly contained within the building. A suitable tree structure would therefore be a top node that represents the building and two children that represent the rooms. Entering the building would then be the same as entering the hall, and from there one would be able to enter one of

the rooms. But how do we express the door in between the two rooms? There is no connection between these rooms in the tree. We may wish to say that any two rooms within any building are connected, but that is generally not the case.

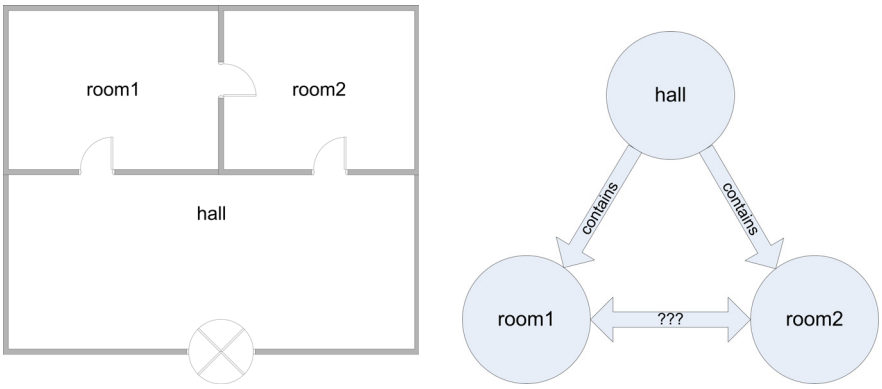

Figure 1: The traditional floor plan and its tree representation.

These problems quickly multiply when virtual assets have connections around the globe. Similar to the floor plan example, we can also think of two networks separated by a firewall. The choice to represent one network as “inside” and the other as “outside”, as in [15], will depend on the location of the assets, but cannot be meaningfully deduced from the structure of the world only. If the asset were on the other side of the firewall, the containment would be reversed. The representation of the structure of the world is then dependent on the value assigned to the entities, which is not only confusing, but may also have consequences for the attacks that can be found. We may also wish to combine digital and physical assets, creating new containment issues. How, for example, do we draw a containment tree when a computer is both *in* a room and *in* a network? In that case the computer is contained in multiple other entities as well. When combining the digital, physical and social world, such connections emerge everywhere. We therefore conclude that containment is not the best concept for graph-based system models, and we will develop an alternative later in this paper.

Existing approaches often combine a graph structure with an underlying process model to represent a world. Based on our previous work [4], it appears that the importance of finding an appropriate graph structure outweighs the importance of choosing the underlying process model when focusing on physical mobility and human action. Indeed, many features of process representations could be removed when focusing on acquisition of access. This motivates our choice to focus purely on graph structure and graph transformations in this paper. Moreover, graphs might well be more intuitive to business users than process descriptions. A third reason is that there is support for model checking of graph-based systems, notably in the form of the Groove tool [17]. Notwithstanding this choice, it would be very well feasible to extend nodes with processes if required for modelling purposes, but this is not discussed further here.

2.3 Our Approach

In this paper, we aim at addressing the question how to include human action in system models. The traditional approach was the movement of an attacker through a containment-based hierarchy. One can then add more actors to include other persons. However, when adding the issues of containment-based models to the argument, it seems a better idea to focus on a different kind of topology instead, not based on physical analogies. As the possession of information is primarily a social phenomenon, contributing to the way in which social beings will act, we develop an approach that starts from a *social* topology of

access instead.

In summary, we aim at addressing:

- the problem of combining digital, physical and social aspects of information security in a single system model;
- the limitations of the notion of containment as the basis of system models;
- the limitations of physical topologies for system models.

2.4 Related Work

Many different system models have been proposed. In the ambient calculus [12], the containing enclosure is termed "ambient". Ambients can contain other ambients, ambients within the same ambient can interact and an ambient can specify conditions on the entering and leaving of other ambients. Franqueira et al. [15] use ambient calculus to perform vulnerability analysis, but they add extra connections to allow more than just containment. Scott [16] also developed a tree-based model for representing physical location and mobility of entities, limiting the possibility to express multiple paths between entities. Moreover, in Scott's model it is possible to "teleport" an entity from one location to another, ignoring layers of protection that may reside in between [8]. Dragovic and Crowcroft [13, 14] address the protection given by the containment and the associated exposure of the data, but still rely on a containment tree to model paths to the assets. Probst and Hansen [3] propose a more flexible system model, but still employ a strict separation between locations and actors. The same holds for the bigraph system model [18, 19]. A quite different approach is taken by Mob_{adtl} [20]. Here, the world is divided into one-level *neighbourhoods*, with each neighbourhood having a guardian that moderates access from and to the outside world. The protection given by containment is transferred to the guardians. The Mob_{adtl} solution further has the advantage that the resulting structure is flat, but again membership of entities is confined to a single neighbourhood. Since it is required to model multiple levels of access control via different paths, this is again too limiting.

The ontology for our reference model is based on hypergraphs, which have been used before in security. Morin et al. [21] model a network as a hypergraph for alert correlation in intrusion detection. Modelling quality of protection for outsourced business processes is discussed by Massacci and Yautsiukhin [22]. Baiardi et al. [23] use hypergraphs to model security dependencies in the context of risk analysis. We believe we are the first to model the world as a hypergraph for finding possible attacks.

Our earlier work in security modelling focused on integrating digital, physical and social aspects of security in a single model called Portunes [4], based on the Klaim family of languages [24]. Klaim is a process calculus for agent interaction and mobility, based on tuple spaces. There are several Klaim dialects, including μKlaim [25], OpenKlaim [26] and acKlaim [27], where the μKlaim and acKlaim languages focuses specifically on access control. In Portunes we already made the simplifying choice to represent mobility by moving nodes rather than evaluating processes. Here, we radicalise these ideas by proposing a completely new graph structure for the model in terms of a social topology.

3 An Actor-Network View on Security

In this section, we present our sociological point of departure for security modelling. In sociology, starting from the field of science and technology studies, a theory has been developed that focuses on low-level interactions rather than high-level concepts to explain social phenomena. This approach is usually called actor-network theory (ANT), and its most important advocate is Bruno Latour [28]. Latour includes technological artefacts in social analysis, by granting them the status of actors (or as he prefers:

actants). An action is never performed by a single actant, but always by a complex of associated entities, without whom the action would have been different. Shooting is not done by a person alone, nor by a gun alone, but by the composition of both. The person makes the gun shoot, but the gun also makes the person shoot (she would not have done so without the gun).

Many of the examples Latour discusses are related to influencing human behaviour. For example, he discusses the measures that a hotel manager can take to make sure that guests return their keys [29]. By including more actants, such as signs asking the guests to return keys, or – more effectively – bulky key rings attached to the keys, more guests will conform to the policy, despite their own capabilities. Like many of his other examples, this is essentially a security problem. In this sense, security problems can be understood in terms of which agents can be mobilised to perform a certain action. Whether the action is successful is then dependent on which agents cooperate. For example, a human and a key together can open a door, but neither a human nor a key can open a door by itself. Who *initiates* the action is irrelevant from an actor-network point of view, as it is always the combined agents that perform it: the key may invite the human to open the door or the other way around (or both), but the result is the same.

Characteristic of actor-network theory is the completely flat social topology. Instead of having small structures contained in larger ones, the focus is on the connecting elements. Actors connect networks and networks connect actors. Information security faces a similar paradigm change with de-perimeterisation and the problems in containment-based modelling. Where traditionally a door would be represented as a perimeter around a room, it may also be seen as a connecting entity between the room and the hallway. From this point of view, actor-network theory may be the basis for a theory of information security that does not focus on containment. The question is how to translate actor-network theory from a sociological theory to a theory in the field of information science.

Actor-network theory has been used for various purposes in the analysis of information systems, notably in describing stakeholder interactions in the design process [30]. Here, we use it as the basis for a formal model of interactions *within* information systems, including the physical and social environment of devices. Similar to interpreting actor-networks as a formal model of computation [31], we see actor-networks as a formal model of *access*. Connections between actors and networks are the central theme. In essence, the connections we are interested in from an information security point of view are connections between pieces of information. Rather than considering an extension of the physical space we are familiar with, this requires thinking in terms of a space filled with connected and disconnected information entities (i.e. pieces of information). In contrast to many approaches to security modelling, which take physical space as their starting point, we are interested in a different kind of space, which has sometimes been termed *infosphere* [32]. This is the space filled by information entities instead of physical objects.

If we focus on the infosphere, the spatial arrangements in the physical world are only relevant as far as they enable or limit interaction between entities. For example, the situation where different entities are in the same room is only relevant for the consequence that this allows these entities to interact. Therefore, we can abstract from the spatial arrangement and define the ontology in terms of interaction capabilities instead. This is precisely how actor-network theory and the domain of information fit together: we use an actor-network view on interaction in the information domain. The most important characteristic of our model is that it is *flat*. Similar to the transformation Bruno Latour proposes in sociology, containment is discarded as an important notion in the topology. The general structure of the model is then represented by which entities can access each other, interact and collaborate.

In summary, an actor-network view on security involves:

- symmetry between human and non-human entities, and ascription of actions to a set of entities rather than a single initiator;
- a flat rather than hierarchical structure;

- a topology in terms of action-enabling connections.

For the example of the building, this will also yield a quite different representation (Fig.2). The doors, implicit in the original model, now show up as key entities. We say that the doors can collaborate with entities of both rooms that they connect, instead of saying that the door separates inside from outside. A door or a firewall is a *connecting* entity rather than a containing perimeter, explicating its role in access-acquiring actions. To simplify the connections between entities, we use *groups* of entities to represent the capability of these entities to interact. Thus, entities that are in the same room will belong to the same group, and can therefore interact. This prevents having to add separate connections between all of those entities. The door will belong both to the group of entities in the room and the group of entities in the hall. This represents the crucial role of the door in the ability of other entities to move from one group to the other.

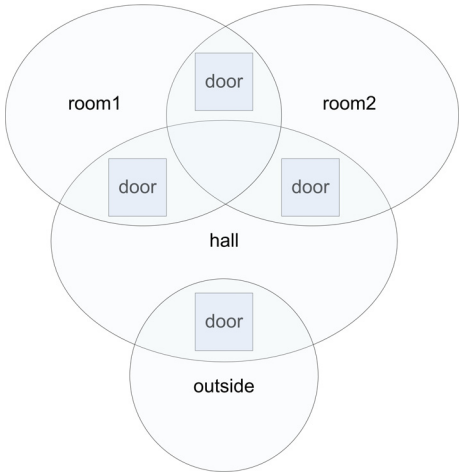

Figure 2: The floor plan from an actor-network point of view. The circles denote groups and their contained entities; they have no spatial meaning.

4 Actor-NetworK Hypergraphs (ANKH)

Based on the inspiration from sociology as well as our own work in security modelling [4], we aim at developing a general reference framework for including human action in system models, based on interaction capabilities of information entities. We model security in terms of interaction or collaboration based on group membership. In the model, *entities* belonging to the same *group* can interact. A group may consist of all the entities in a room, all the entities in a digital network, or all the entities that a person is carrying. Entities may be members of more than one group. The group structure is flat: groups cannot be members of groups. For convenience, groups may be named, but this is not necessary for the analysis.

4.1 Hypergraph Transformations

The notions of entity and group give rise to a formal representation in terms of hypergraphs, where entities and groups are mapped to nodes and hyperedges, respectively. We therefore label the model ANKH, for Actor-NetworK Hypergraphs. In a hypergraph, a hyperedge is an edge connecting any number of vertices, i.e. a hyperedge is a non-empty subset of the set of nodes. Nodes represent entities and hyperedges represent groups.

Definition 1: A *world* is a hypergraph $G = (V, E)$, where V is a set of nodes and E is a set of hyperedges.

To model vulnerability in the form of possible step-wise attacks, we need to define how the world can change over time. The basic assumption is that the world changes by modifications in group membership, i.e. transformations of the hyperedges of the graph. To make the complexity manageable, the set of nodes is static; i.e. no new nodes are created and no nodes disappear.[1] The main question to be answered is then under which conditions hyperedges can change, in particular when a node can obtain or lose membership of a hyperedge.

A special role is reserved here for nodes that are contained in more than one hyperedge. For example, a door connecting two rooms will be contained in the hyperedges representing both rooms. We call such multi-edged nodes *guardians*.[2] Guardians serve as connecting entities between different groups (hyperedges), and they control the possible transformations of hyperedges. Members of one group can become a member of another group only through interaction with a guardian.

Definition 2: A *guardian* is a marked node that is a member of more than one hyperedge. We then say that it is a guardian *between* these hyperedges.

One can choose to define guardians statically or dynamically, depending on the purposes of the model. If defined statically, guardians are only those nodes that belong to multiple hyperedges in the initial configuration. If defined dynamically, guardians can emerge during evolution of the model, when entities acquire membership of additional hyperedges. For example, a computer could become a guardian between a room and a network when it acquires a network connection. Somebody having access to the room can now acquire access to the network through this new guardian. For reasons of clarity of exposition, we assume statically defined guardians from now on.

An event (change of state) can occur if a guardian g interacts with another entity e, with $H, I \in E$; $g, e \in H$ and $g \in I$. This may result in one of the following actions:

- move(g, e, H, I): g moves e from group H to I;
- copy(g, e, H, I): g copies e from group H to I;
- remove(g, e, H) g removes e from group H.

The effect of actions are changes in hyperedge (group) membership. The following events are examples of what may happen:

- if g is a human and e a bag, and g moves e from the room (H) to her possessions (I) or vice versa: the bag is picked up or put down;
- if g is a computer and e is a piece of data, and g copies e from the hard-disk (H) to memory (I): the data is duplicated;
- if g is a USB stick and e a piece of data, and g removes e from the group representing its contents (H): the data is erased.

In these actions, we call the guardian the *active* entity and the object being affected the *passive* entity. In many situations, these terms may be somewhat counter-intuitive compared to a physical ontology. For example, when a person enters a room, the door will be the active entity, and the person will be the

[1] As long as we are interested in confidentiality of existing data, this is justified; this assumption may need to be changed when modelling for example integrity of data. In the latter case, modifying a copy is not the same as changing the original.

[2] This terminology is also used in Mob$_{adtl}$ [20]. However, the guardians in that approach are pre-assigned and they monitor pre-defined neighbourhoods. In the present paper, the only restriction is that a guardian is contained in more than one group.

passive one. This is due to the fact that we focus *only* on interaction possibilities, and here, *the door gives the person access to the room*. As mentioned before, which entity initiates the action is outside the scope of the model, since it is irrelevant from the point of view of what is possible security-wise. Note that *move* is identical to *copy* followed by *remove*; however, particularly in the physical domain, transfer of entities is often restricted to precisely the *move* action. We therefore include it as a separate action, allowing for differentiation of capabilities of digital and physical objects. Note also that once entities lose all their group memberships, they can never obtain one again, as there is no guardian they can interact with. This represents deletion or destruction.

Example 1: Alice is a member of the group of entities in the room that she is in, but also a member of the group of entities that she is carrying around. Therefore, she is a guardian. An object in the room may interact with Alice in order to become a member of the group of objects that she is carrying around (Alice can pick up an object). The object will lose membership of the group of entities in the room, and become a member of the group of entities that Alice is carrying. If Alice then asks the door (also a guardian) to become a member of the hall, she loses her membership of the room (as the door will only allow *move* actions, not *copy*). She is still a member of the group of objects that she is carrying (in the physical world one would say she is taking the objects with her). We do not need to explicitly model the concept of carrying, as this is already implicit in the possible hypergraph transformations: Alice will keep the possibility of interacting with the entities that she is carrying. The corresponding actions are: move(Alice,object,room,Alice's possessions) and move(door,Alice,room,hall).

4.2 Capabilities

We have already seen that there are constraints on the possible modifications in group membership. To be able to use these constraints for finding attacks, they need to be included explicitly in the model of the world. The constraints are centred around the guardians, as these are the only entities that can effect changes in group memberships. In a positive formulation, we use the notion of capabilities to describe possible events. *Capabilities* describe under which conditions an entity will – as a guardian – allow other entities to join or leave groups that it is a member of.

In the previous example, Alice can pick up an object only if this action conforms to the capabilities of Alice with respect to this object (for example, if Alice can lift the object and thereby make it a member of the group of entities she is carrying). Such capabilities may correspond to "natural" properties of the two entities, but also to "social" properties, such as whether a person will be inclined to leave her key. Other restrictions are specified in terms of which other entities need to be present to enable the action. We call those *credentials*. For example, to enable the event that a door moves a human into a room, the key must also be present. Depending on the guardian's capabilities, the passive entity may or may not be required to give up membership of other groups. In the previous example, when Alice tries to exit the room, the door won't let her stay in both places at the same time. We see that capabilities can be either *implicit* properties of the world (a person can move into a room, but a room cannot move into a person) or *explicit* security measures (a person can only move into this room if she has the right key).

A capability can thus be expressed in terms of (1) the type of event (move, copy or remove); (2) the guardian; (3) the entity being affected (the entity that changes group membership); and (4) the other entities that are required to be accessible (credentials). Using this structure of capabilities, we can now define states.

Definition 3: A *state* is a tuple (G,C) consisting of a world (a hypergraph $G = (V,E)$) and a set of capabilities C of type $A \times V \times V \times 2^V$, where $A = (move, copy, remove)$ is the set of actions and 2^V is the powerset of V.

The capability relation contains credentials: the associated action is only possible if the entity can access the credentials directly (they are in one of its groups). $(a,g,e,\{c\}) \in C$ means that entity g will interact with entity e in order to perform event a, provided g, e and c are in the same group. For example, a door (g) will let a person (e) into a room if the door, the person and the right key (c) are in the hall.[3] For now, we assume that the capability function is static: capabilities are defined for the initial state, and entities do not change their capabilities.

We can now formally define the hypergraph transformation rules. Given a world (G,C), where $g,e \in V$; $H,I \in E$ and $R \subseteq E$, the following rules define the possible transformations (with preconditions above the line, postconditions below):

$$\frac{g,e \in H \qquad g \in I \qquad (move,g,e,R) \in C \qquad R \subseteq H}{g \in H \qquad g,e \in I \qquad e \notin H \qquad (move,g,e,R) \in C \qquad R \subseteq H}\ \text{move}$$

$$\frac{g,e \in H \qquad g \in I \qquad (copy,g,e,R) \in C \qquad R \subseteq H}{g,e \in H \qquad g,e \in I \qquad (copy,g,e,R) \in C \qquad R \subseteq H}\ \text{copy}$$

$$\frac{g,e \in H \qquad (remove,g,e,R) \in C \qquad R \subseteq H}{g,e \in H \qquad e \notin H \qquad (remove,g,e,R) \in C \qquad R \subseteq H}\ \text{remove}$$

In case of social interactions, people may accept others into their group, which means that they can perform actions for them. For example, a student may ask a janitor to open a room for her. If the janitor accepts her in his group, the student can go wherever the janitor can go (he will open the rooms for her). Again, this represents interaction possibilities rather than physical containment, as the janitor will not literally pick up the student. In this way, our model is more flexible than a containment-based one. As system models deal with access, human interaction is specified in terms of who gives whom access to what, based on what a person already possesses.

4.3 Goals

A goal is a set of undesirable target states. Goals thus represent which entities should or should not be in the same group. Goals can be checked by analysing the reachability of undesirable states, starting from the initial state of the world. The initial state is specified by the physical arrangement of the building, the layout of the network, the locations of employees, and the capabilities of all entities. In particular, the group memberships of the attacker and the asset are vital. Target states are specified by the property that the attacker is in the same group as the asset.

5 An Example

As an example, we consider the road apple attack. The term road apple refers to an apple that is found on a road, tempting the finder to take it. In the IT world, the apple is usually an infected generic dongle, with the logo of the organisation, left by the adversary in a public place in the organisation's premises, such as a canteen. When an employee finds the dongle, she may be tempted to plug the dongle into her laptop [33]. If she does, the dongle will install a rootkit on the hard disk drive (hdd) without the employee's knowledge. Entities and groups are depicted in Fig. 3. The goal is to have the rootkit in the hdd contents. Note that for reasons of simplicity, we have still chosen a world that can be represented as a tree. To make use of the full power of our approach, we could add hyperedges representing a

[3] This means the key is directly accessible from the hall; otherwise, it would have been in a separate group representing for example a cabinet in the hall.

wireless LAN, for example, containing both the target computer and a device controlled by the attacker, giving additional opportunities for attack. This would, however, make the example too complicated for explanation purposes.

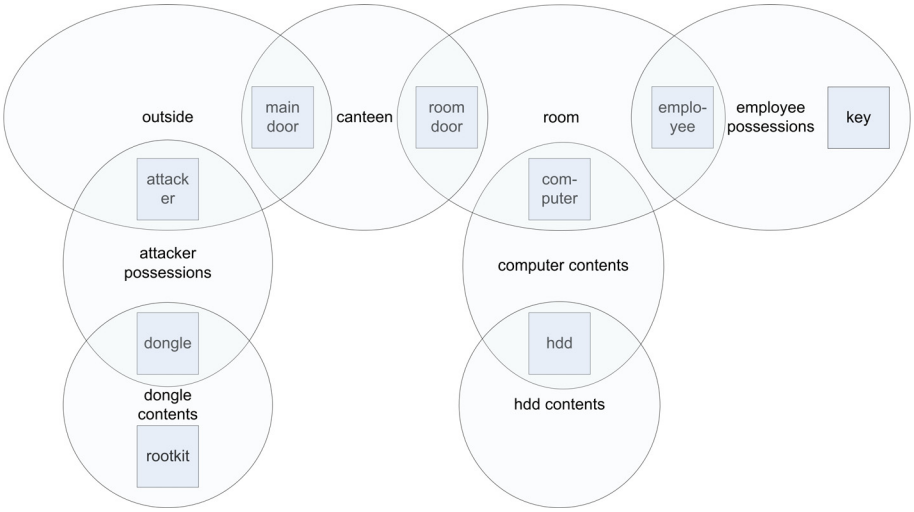

Figure 3: The initial state of the road apple attack from an actor-network point of view.

The following capabilities are present:

- (move,room door,employee,{key})
- (move,room door,attacker,{key})
- (move,main door,employee,∅)
- (move,main door,attacker,∅)
- (move,employee,dongle,∅)
- (move,attacker,dongle,∅)
- (move,computer,dongle,{employee})
- (move,computer,dongle,{attacker})
- (copy,dongle,rootkit,∅)
- (copy,hdd,rootkit,∅)

Note that for reasons of brevity, abbreviations could be introduced by means of entity types. A capability could then refer to a type rather than a specific entity. A wildcard (i.e. any entity) is then represented by a universal type. An additional restriction (not shown here) is that physical entities can only be added to physical groups, and similarly for digital entities. This prevents, for example, the rootkit from being added to employee possessions. A possible attack (indeed, the road apple) would use the following actions:

1. move(main door,attacker,outside,canteen): attacker is moved from outside to canteen
2. move(attacker,dongle,attacker possessions,canteen): dongle is moved from attacker possessions to canteen
3. move(room door,employee,room,canteen): employee is moved from room to canteen

4. move(employee,dongle,canteen,employee possessions): dongle is moved from canteen to employee possessions
5. move(room door,employee,canteen,room): employee is moved from canteen to room (requires key)
6. move(employee,dongle,employee possessions,room): dongle is moved from employee possessions to room
7. move(computer,dongle,room,computer contents): dongle is moved from room to computer contents (needs employee)
8. copy(dongle,rootkit,dongle contents,computer contents): rootkit is copied from dongle contents to computer contents
9. copy(hdd,rootkit,computer contents,hdd contents): rootkit is copied from computer contents to hdd contents

There is a seemingly special kind of credential that is vital in this example. The dongle needs a credential to go into a computer: it can enter a computer because an employee is there. The employee is thus a credential in this case. This may seem counter-intuitive, but since the employee neither changes group membership nor acts as a guardian, this is the natural way to express this. This is precisely what the actor-network theory tells us: there is symmetry between humans and non-humans in what they can do. For security this is projected onto access relations and changes in those by access acquisition. For our reference model, this means that entities, human or non-human, can take the role of active entity, passive entity, or credential, depending on the situation.

6 Analysis

In this section, we present techniques for generating attack scenarios based on a goal and an initial state. There is no need to invent completely new analysis procedures, as existing techniques are well-suited for the ANKH framework. They do however need to be translated to fit the modelling constructs. Our analysis is based on the one presented in [34]. In their terminology, we can define the *attributes* of our model, i.e. the changeable properties that describe a state. Under the assumption that capabilities are static, these are simply the group memberships of entities (hyperedge memberships of nodes). Attributes, again in the terminology of [34], can be changed by *exploits*, which are transformations that change the attributes. In ANKH, exploits are changes in group membership enabled by capabilities.

To allow systematic analysis of the possible transformations, we make the simplifying assumption that when an entity belongs to a group at one point in time, this means that the entity can always go back if it wants to. Entities are thus never moved or removed, but always copied. This is a monotonicity assumption [34]. In this way, we can create a table of all *possible* groups an entity can be in after a certain number of steps. That is, in each step, we investigate all possible events, and for each possible event, we add the moved entity to the possible entities of the group it moved to. It will also remain as a possible entity in its previous group (Algorithm 1).

Algorithm 1 Possibility analysis

```
Require: initial state
  repeat
    for each possible transformation do
      put moved entity in new group for this step
      set links to relevant conditions of previous step
    end for
  until table unchanged in this step
```

In the first phase of the analysis of the road apple example, a table is thus created (Table 1), which represents for each step and for each group the possibly included entities. For each entity in the table

	Outside	Attacker poss.	Dongle cont.	Canteen	Room	Employee poss.	Comp. cont.	Hdd cont.
0	a,md	a,d	d,r	md,rd	rd,e,c	e,k	c,h	h
1	a,md	a,d	d,r	md,rd,a,e	rd,e,c	e,k	c,h	h
2	a,md	a,d	d,r	md,rd,a,e,d	rd,e,c	e,k	c,h	h
3	a,md	a,d	d,r	md,rd,a,e,d	rd,e,c	e,d,k	c,h	h
4	a,md	a,d	d,r	md,rd,a,e,d	rd,e,c	e,d,k	c,h,d	h
5	a,md	a,d	d,r	md,rd,a,e,d	rd,e,c	e,d,k	c,h,d,r	h
6	a,md	a,d	d,r	md,rd,a,e,d	rd,e,c	e,d,k	c,h,d,r	h,r

Table 1: Analysis of the example: first stage. Rows indicate steps in the algorithm; columns indicate groups. Entities are denoted by their first letters.

that was moved or copied in an event, we also administer the relevant conditions that made the event possible. For example, if the computer moves the dongle from the room into the computer contents, we link the event to two conditions of the previous step, namely the dongle and the employee being in the room. The last row of the table gives the situation when no more new group memberships are possible.

In the second phase of the analysis, we supply a goal. As mentioned before, a goal expresses undesirable target states. We assume that the goal is *simple*, i.e. refers to the condition of an entity being in a group. Otherwise, we first decompose the goal into simple goals. In the last row of the table, we can find possible instantiations of the goal. If we find such an instantiation, we track this situation back through the table by means of the links to the relevant conditions. In this way, we can recreate the trace of events that led to the undesired situation. This trace thus constitutes an attack (Algorithm2). As shown in the example of a human moved into a room, it is possible that multiple conditions were necessary for an event to happen (a human AND the key need to be present). In this case, the trace of the attack branches as an AND. It is also possible that several different conditions made an event possible. In this case, the trace branches as an OR. In this way, we obtain an attack tree as defined in [35]. It might be possible that in several stages of the calculation of the trace, the same condition emerges. To obtain a better description of the attack, we may choose to represent this as multiple pointers to the same node in the attack tree, in which case it becomes a graph rather than a tree.

Algorithm 2 Attack generation (recursive)

Require: simple goal (entity in group)
  set simple goal as top node of attack tree T
  $P \leftarrow$ conditions linked from simple goal by Algorithm 1
  if P is disjunction **then**
    make T an OR-node
  else
    make T an AND-node
  end if
  for $p \in P$ **do**
    add as a child the result of the algorithm for p
  end for
  return T

For the second stage of the analysis of our example, we request an attack for the goal of the rootkit being in the hdd contents. In the last row of the table, we can see that this situation may indeed be the case. Now we follow backwards the conditions that made this state possible. For the last step, the movement of the rootkit into the hdd was made possible by the condition that the rootkit was in the computer. Backtracking one more step, this was possible because the dongle was in the computer. The dongle got

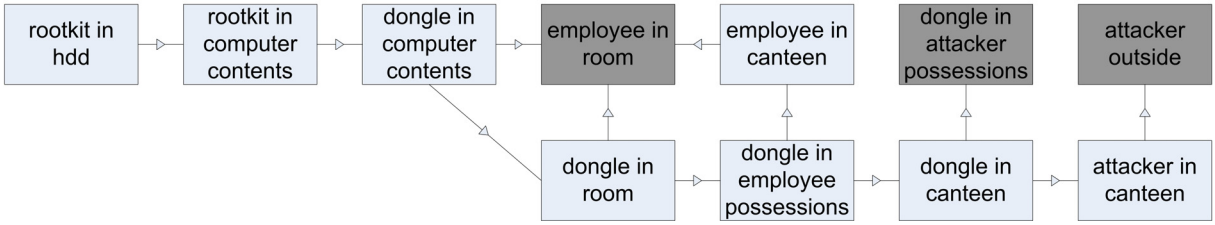

Figure 4: The attack graph constructed in the second stage of the analysis of the example. All splits are AND-splits. The dark grey boxes denote conditions that are already true in the initial state.

into the computer because it was in the same room and the employee was present. Here, the trace splits. We need to find out how the employee got into the room and how the dongle got into the room. The employee was already in the room at the start. The dongle got into the room because it was possessed by the employee. The employee got the dongle because both the employee and the dongle were in the canteen. Following this method, we can trace the attack back to the start (Fig.4).

Note that for this attack to work, the employee needs to be *both* in the canteen and in the room. The *order* of events is thus important for the attack to succeed. The monotonicity assumption leads to an overapproximation, and the resulting attack tree must be judged from this point of view to determine whether the attack is possible if the limitations of physical movement are put back in place.

Following [34], the theoretical complexity of Algorithm 1 can be shown to be $O(N^2E^2)$, where N is the number of nodes and E the number of hyperedges. Let N be the number of nodes and E the number of hyperedges. An *attribute* represents whether a particular node is in a particular hyperedge. Consequently, there are NE attributes (i.e. fields in the incidence matrix). Because of monotonicity, all possible attributes are satisfied after at most NE steps in the first algorithm. For each newly satisfied attribute, we can optimise the look-up of newly enabled actions by pre-computing a table with all actions per attribute. At most NE actions can be made possible, corresponding to moving an entity to a group. For these actions, we need to check whether new attributes are satisfied by their execution. This means that the total worst-case complexity is the number of attributes times the actions enabled per attribute, giving $O(N^2E^2)$. In a realistic scenario, the number of hyperedges (rooms, networks, possessions) will typically be in the order of the number of nodes. Given this assumption, the complexity can also be expressed as $O(N^4)$. Further optimisation techniques, such as those in [36], may be applicable.

For Algorithm 2, the number of steps in the algorithm can again be at most NE, being the maximum number of attributes to be satisfied. Since each step is a constant look-up, the total complexity is NE. Note that these complexities are not comparable to those of purely digital vulnerability modelling, as the latter typically does not address mobility, and can therefore keep the graph or tree static.

These algorithms can easily be adapted for probabilistic capabilities. The only difference is that for each link in the model, a probability will now be stored indicating its likelihood. For human behaviour, probabilities represent that people will not act in the same way all the time (characteristic of social science results), and can therefore be a valuable addition to the model.

For the practical system that led to the theoretical insights presented here [4], we have a running implementation [37] of the algorithms within the Groove tool [17]. We aim at adapting the implementation for the hypergraph model, and comparing the two approaches.

7 Conclusions

In this paper, we introduced the actor-network approach for including humans in system models for information security, based on interaction possibilities. Main source of inspiration is the sociology of

actor-network theory, which argues that analysis of connections should be based on a flat topology, and that actions are performed symmetrically by multiple cooperating entities. We translated the idea to the domain of information, where we defined *groups* and *guardians*. Guardians are able to move or copy other entities between their groups, depending on specified capabilities. We defined states, events and capabilities in the formal model of Actor-NetworK Hypergraphs (ANKH) and showed how an initial world can be analysed for reachability of undesirable states, and how an attack graph can be constructed from the analysis. The road apple attack was presented as the main example.

Compared to existing models, our reference model treats actions as symmetric with respect to the involved people and objects. The only relevant criteria are what is being moved (given new access options), and what needs to be there for this move to be possible. People are not the primary movers; their movement is physical and therefore visible, but from an information perspective this does not have priority. Instead, any guardian may initiate a move. From an information point of view, a door moves a human into a room, because it is the door that guards the group of the room. The door thus admits the human into the group of the room. The resulting model is flat, symmetric, and has a topology only based on access relations. This rethinking of security modelling is our main contribution.

The hypergraph-based formal model is very simple, expressive, and reasonably efficient, and provides a basis for understanding how human action contributes to (in)security, without a priori making a distinction between human and non-human actors. It may look simplistic to some, but the intention was indeed to go back to the basics, and the assumptions were chosen to be as minimal as possible. In future work, we aim at developing our theoretical insights into a full-blown vulnerability model with tool support, based on the implementation of our earlier system model. Practical considerations can be addressed by applying the model in realistic case studies, where the attacks generated can then be tested for effectiveness by means of penetrating testing experiments [5]. The case studies will also assist in identifying precisely which conceptual extensions are possible or necessary.

A useful extension could be the use of dynamic capabilities, where actions may not only affect the connections in the world, but also the capabilities of entities. This would correspond to for example changing the security settings in a system. Violations of, say, separation of duty could then be used in an attack either by acquiring the credentials of the other party or by changing the settings. Also, probabilistic capabilities could be considered.

Our primary aim here was to show a basic, simple reference model for including human action in system models, and that has been achieved. We encourage further research and critical remarks on which extensions are needed to fully grasp the intricacies of human behaviour and its influence on the vulnerability of socio-technical information systems. In this context, our reference model can be used as a basis for providing extensions for human behaviour in other types of system models. An important topic to be addressed by the community is the integration of such system models with protocol models, such that the resulting frameworks specify not only the capabilities, but also how these capabilities are achieved by interaction (protocols). Then, it can not only be specified who gives whom access to what based on which credentials, but also in which sequences of communicative acts this will actually be the result. Again, adapting the associated protocol models for including human behaviour is key. The long-term goal is then to enable the automatic identification of attacks that combine weaknesses in the system architecture with weaknesses in interaction protocols, not only between computer systems, but also with the people involved. Only then can we claim to really have addressed security in socio-technical systems.

Acknowledgements

This research is supported by the research program Sentinels (`www.sentinels.nl`). Sentinels is being financed by Technology Foundation STW, the Netherlands Organization for Scientific Research (NWO), and the Dutch Ministry of Economic Affairs. The author wishes to thank (in alphabetical order) André

van Cleeff, Trajce Dimkov, Pieter Hartel and Virginia Nunes Leal Franqueira for helpful comments.

References

[1] F.-Y. Wang, "The emergence of intelligent enterprises: From CPS to CPSS," *IEEE Intelligent Systems*, vol. 25, no. 4, pp. 85–88, 2010.

[2] M. Bishop, L. Coles-Kemp, D. Gollmann, J. Hunker, and C. Probst, "10341 Report – Insider threats: Strategies for prevention, mitigation, and response," in *Insider Threats: Strategies for Prevention, Mitigation, and Response, Dagstuhl Seminar Proceedings*, no. 10341. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2010.

[3] C. Probst and R. Hansen, "An extensible analysable system model," *Information security technical report*, vol. 13, no. 4, pp. 235–246, 2008.

[4] T. Dimkov, W. Pieters, and P. Hartel, "Portunes: representing attack scenarios spanning through the physical, digital and social domain," in *Proc. of the Joint Workshop on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security (ARSPA-WITS'10). Revised Selected Papers, Paphos, Cyprus, LNCS*, vol. 6186. Springer-Verlag, March 2010, pp. 112–129.

[5] T. Dimkov and W. Pieters and P.H. Hartel, "Two methodologies for physical penetration testing using social engineering," in *Proc. of the Annual Computer Security Applications Conference (ACSAC'10), Austin, Texas, USA*. ACM Press, December 2010, pp. 399–408.

[6] P. Ryan and S. Schneider, *The modelling and analysis of security protocols: the CSP approach*. Addison-Wesley Professional, 2001.

[7] C. Cremers, "The Scyther Tool: Verification, falsification, and analysis of security protocols," in *Proc. of the 20th International Conference on Computer Aided Verification (CAV'08), Princeton, USA, LNCS*, vol. 5123. Springer-Verlag, July 2008, pp. 414–418.

[8] T. Dimkov, Q. Tang, and P. Hartel, "On the inability of existing security models to cope with data mobility in dynamic organizations," Centre for Telematics and Information Technology, University of Twente, Tech. Rep. TR-CTIT-08-57, 2008.

[9] P. Hartel, M. Junger, and R. Wieringa, "Cyber-crime science = crime science + information security," http://eprints.eemcs.utwente.nl/18500/, Centre for Telematics and Information Technology, University of Twente, Enschede, Technical Report TR-CTIT-10-34, October 2010.

[10] P. Verbeek, *What things do: Philosophical Reflections on Technology, Agency, and Design*. Pennsylvania State University Press, 2005.

[11] Jericho Forum, "Jericho whitepaper," 2005. [Online]. Available: http://www.opengroup.org/projects/jericho/uploads/40/6809/vision_wp.pdf

[12] L. Cardelli and A. Gordon, "Mobile ambients," *Theoretical computer science*, vol. 240, pp. 177–213, 2000.

[13] B. Dragovic and J. Crowcroft, "Information exposure control through data manipulation for ubiquitous computing," in *Proc. of the 2004 Workshop on New Security Paradigms (NSPW'04), Nova Scotia, Canada*. ACM Press, September 2004, pp. 57–64.

[14] B. Dragovic and J. Crowcroft, "Containment: from context awareness to contextual effects awareness," in *Proc. of the 2nd International Workshop on Software Aspects of Context (IWSAC'05), Santorini, Greece*, July 2005.

[15] V. Nunes Leal Franqueira, R. Lopes, and P. van Eck, "Multi-step attack modelling and simulation (MsAMS) framework based on mobile ambients," in *Proc. of the 24th Annual ACM Symposium on Applied Computing (SAC'09), Honolulu, Hawaii, USA*. ACM Press, March 2009, pp. 66–73.

[16] D. Scott, "Abstracting application-level security policy for ubiquitous computing," Ph.D. dissertation, University of Cambridge, 2004.

[17] A. Rensink, "The GROOVE simulator: A tool for state space generation," in *Proc. of the 2nd International Workshop on Applications of Graph Transformations with Industrial Relevance (AGTIVE'03),Charlottesville, VA, USA, LNCS*, vol. 3062. Springer-Verlag, September-October 2004, pp. 479–485.

[18] D. Grohmann, "Security, cryptography and directed bigraphs," in *Proc. of the 4th International Conference on Graph Transformation (ICGT'08), Leicester, UK, LNCS*, ser. LNCS, vol. 5214. Springer-Verlag, September 2008, pp. 487–489.

[19] R. Milner, "Bigraphical reactive systems," in *Proc. of the 12th International Conference on Concurrency Theory (CONCUR'01), Aalborg, Denmark, LNCS*, vol. 2154. Springer-Verlag, August 2001, pp. 16–35.

[20] G. Ferrari, C. Montangero, L. Semini, and S. Semprini, "Mobile agents coordination in Mobadtl," in *Proc. of the 4th International Conference on Coordination Languages and Models (COORDINATION'00), Limassol, Cyprus, LNCS*, vol. 1907. Springer-Verlag, September 2000, pp. 232–248.

[21] B. Morin, L. Mé, H. Debar, and M. Ducassé, "M2D2: A formal data model for IDS alert correlation," in *Proc. of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'02), Zurich, Switzerland, LNCS*, vol. 2516. Springer-Verlag, October 2002, pp. 115–137.

[22] F. Massacci and A. Yautsiukhin, "Modelling of quality of protection in outsourced business processes," in *Proc. of the 3rd International Symposium on Information Assurance and Security (IAS'07), Manchester, UK*. IEEE, August 2007, pp. 247–252.

[23] F. Baiardi, S. Suin, C. Telmon, and M. Pioli, "Assessing the risk of an information infrastructure through security dependencies," in *Proc. of the 1st International Workshop on Critical Information Infrastructures Security (CRITIS'06), Samos Island, Greece, LNCS*, vol. 4347. Springer-Verlag, August-September 2006, pp. 42–54.

[24] R. D. Nicola, G. L. Ferrari, and R. Pugliese, "Klaim: A kernel language for agents interaction and mobility," *IEEE Transactions on software engineering*, vol. 24, no. 5, pp. 315–330, May 1998.

[25] D. Gorla and R. Pugliese, "Resource access and mobility control with dynamic privileges acquisition," in *Proc. of the 14th International Colloquium on Automata, Languages and Programming (ICALP'03), Eindhoven, The Netherlands, LNCS*, vol. 2719. Springer-Verlag, June-July 2003, pp. 119–132.

[26] L. Bettini, M. Loreti, and R. Pugliese, "An infrastructure language for open nets," in *Proc. of the 2002 ACM Symposium on Applied Computing (SAC'02), Madrid, Spain*. ACM Press, March 2002, pp. 373–377.

[27] C. Probst, R. Hansen, and F. Nielson, "Where can an insider attack?" in *Proc. of The 4th International Workshop on Formal Aspects in Security and Trust (FAST'06), Hamilton, Ontario, Canada, LNCS*, vol. 4691. Springer-Verlag, August 2007, pp. 127–142.

[28] B. Latour, *Reassembling the Social: An Introduction to Actor-Network-Theory*. Oxford: Oxford University Press, 2005.

[29] M. Akrich and B. Latour, "A summary of a convenient vocabulary for the semiotics of human and nonhuman assemblies," in *Shaping Technology/Building Society: Studies in Sociotechnical Change*, W. Bijker and J. Law, Eds. Cambridge, MA: MIT Press, 1992, pp. 259–264.

[30] G. Walsham, "Actor-network theory and IS research: Current status and future prospects," in *Information Systems and Qualitative Research*, A. Lee, J. Liebenau, and J. DeGross, Eds. London: Chapman Hall, 1997, pp. 466–480.

[31] D. Pavlovic and C. Meadows, "Actor-network procedures: Modelling multi-factor authentication, device pairing, social interactions," Forthcoming.

[32] L. Floridi, "Information ethics: on the philosophical foundation of computer ethics," *Ethics and Information Technology*, vol. 1, no. 1, pp. 37–56, 1999.

[33] S. Stasiukonis, "Social engineering the USB way," 2006. [Online]. Available: http://www.darkreading.com/document.asp?doc_id=95556

[34] P. Ammann, D. Wijesekera, and S. Kaushik, "Scalable, graph-based network vulnerability analysis," in *Proc. of the 9th ACM Conference on Computer and Communications Security (ACM CCS'02), Washington, DC, USA*. ACM Press, November 2002, pp. 217–224.

[35] S. Mauw and M. Oostdijk, "Foundations of attack trees," in *Proc. of the 8th Annual International Conference on Information Security and Cryptology (ICISC'05), Seoul, Korea, LNCS*, vol. 3935. Springer-Verlag, December 2006, pp. 186–198.

[36] X. Ou, W. Boyer, and M. McQueen, "A scalable approach to attack graph generation," in *Proc. of the 13th ACM Conference on Computer and Communications Security (ACM CCS'06), Alexandria, USA*. ACM Press, October-November 2006, p. 345.

[37] A. Ghamarian, M. de Mol, A. Rensink, E. Zambon, and M. Zimakova, “Modelling and analysis using GROOVE,” http://eprints.eemcs.utwente.nl/17830/, Centre for Telematics and Information Technology University of Twente, Enschede, Technical Report TR-CTIT-10-18, April 2010.

Wolter Pieters (1978) is a postdoctoral researcher in information security at the University of Twente. He studied computer science and philosophy of science, technology and society at the same university, and wrote his interdisciplinary PhD “La volonté machinale: understanding the electronic voting controversy” at the Radboud University Nijmegen. Afterwards he advised the Dutch Ministry of the Interior on electronic voting and electronic travel documents. Since September 2008 he works in the VISPER project at the University of Twente, concentrating on disappearing boundaries in information security. He was program chair of the 2010 CPDP workshop on Security and Privacy in Cloud Computing, and will co-organise the 2011 Dagstuhl seminar on Secure Architectures in the Cloud. He published on electronic voting, cloud computing, verification of security properties, access control, and philosophy and ethics of information security.