ISBN 978-82-471-1577-0 (printed ver.)
ISBN 978-82-471-1578-7 (electronic ver.)
ISSN 1503-8181

NTNU
Norwegian University of
Science and Technology

Marie Elisabeth Gaup Moe

Doctoral Thesis

Doctoral theses at NTNU, 2009:98

Marie Elisabeth Gaup Moe

Security, Privacy and Trust in Dynamic Networks

NTNU
Norwegian University of
Science and Technology

Thesis for the degree of
philosophiae doctor

Faculty of Information Technology, Mathematics and
Electrical Engineering
Department of Telematics

Doctoral theses at NTNU, 2009:98

NTNU

Marie Elisabeth Gaup Moe

Security, Privacy and Trust in Dynamic Networks

Thesis for the degree of philosophiae doctor

Trondheim, May 2009

Norwegian University of
Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

NTNU
Norwegian University of Science and Technology

Thesis for the degree of philosophiae doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

ISBN 978-82-471-1577-0 (printed ver.)
ISBN 978-82-471-1578-7 (electronic ver.)
ISSN 1503-8181

Doctoral Theses at NTNU, 2009:98

Printed by Tapir Uttrykk

SECURITY, PRIVACY AND TRUST IN DYNAMIC NETWORKS

Thesis for the degree of philosophiae doctor

MARIE ELISABETH GAUP MOE

Department of Telematics
Norwegian University of Science and Technology
January 2009

Abstract

Emergent networks like mobile ad hoc networks, sensor networks, opportunistic networks, peer-to-peer networks and social networks are introducing new and exciting opportunities of communication between people and devices. But these dynamic networks also introduce many security- and privacy-related challenges. When dealing with complex and dynamic environments, information about the current level of security or privacy, expressed in a quantified manner, could be of great value in a decision-making process. In order to derive such quantified measures there is a need for mathematical models for security, privacy and trust. The development, application and evaluation of such models are the topics of this thesis.

In order to obtain quantitative measures of security, a state modeling approach, which has traditionally been used to model dependable systems is used. The modeling is based on the view that the notions of security and dependability are integrated concepts, both describing aspects of trustworthy computer systems. The state modeling allows for a probabilistic evaluation of the security of the system, which can be used for security quantification, prediction, risk assessment, intrusion detection and intrusion prevention.

The first part of the thesis describes a real-time risk assessment method for computer networks using hidden Markov modeling. Hidden Markov models are well suited for the modeling of sensor trustworthiness in an intrusion prevention system, and as a result of this research, a new method for aggregation of intrusion detection alerts from multiple intrusion detection systems is proposed. New security metrics for computer networks, such as computer network risk, the mean time to next intrusion and the intrusion frequency, are derived from the Markov models. Hidden Markov models are also used for supporting the actions of agents in dynamic networking environments who are faced with significant degrees of uncertainty in making decisions. Assuming access to perfect information about the environment and the properties of the interacting partners is unrealistic, but if agents are able to establish appropriate trust in each other, the decisions-making process would be facilitated and the risk associated with the interactions could still be acceptable. Trust may also play a significant role for the efficient operation of more general multiagent systems. A novel trust model based on hidden Markov modeling and reinforcement learning has been developed, where the measuring of agent

trustworthiness is based on the predicted state probability distribution. Trust modeling is also used as a basis for a decentralized reputation system suitable for dynamic multiagent environments.

As infrastructures are gradually becoming more intelligent, trust may play an increasingly important role in the interactions between network components. A trust-based security extension to the mobile ad hoc network dynamic source routing protocol is given, where the state probability of a node, according to its corresponding hidden Markov model, is being used for deciding the node's trustworthiness. Nodes with different trustworthiness may be offered different service levels based on a trust policy. Since network services normally will be denied to untrusted nodes, an incentive for nodes not to misbehave is created.

Users in dynamic networking environments like mobile ad hoc networks would be particularly exposed to threats against their privacy since they have limited control over the trustworthiness of network nodes that handle the messages sent. Appropriate privacy enhancing cryptographic mechanisms, which can be trusted to work as intended, are required to handle this problem. A novel approach to quantifying the amount of privacy that is offered by anonymous ad hoc routing protocols using conditional entropy is given, which takes into account the proportion of adversarial nodes and includes the a priori knowledge of the attacker.

Preface

This dissertation is submitted in partial fulfilment of the requirements of the philosophiae doctor (Ph.d.) degree at the Norwegian University of Science and Technology (NTNU). The work was performed at the Centre for Quantifiable Quality of Service in Communication Systems (Q2S), Centre of Excellence (CoE), and has been supervised by Professor Svein J. Knapskog and Professor Bjarne E. Helvik at the Department of Telematics. Q2S is established and funded by the Research Council of Norway, NTNU, UNINETT, and Telenor. The document has been formatted in LaTeX using a modified version of the document class *kapproc.cls* provided by Kluwer Academic Publishers.

Acknowledgements

This thesis is the result of collaborative work and could not have been written without the discussions with and advice from my two supervisors Professor Svein J. Knapskog and Professor Bjarne E. Helvik, or all my other co-authors from the Q2S Security Group: André Årnes, Tønnes Brekne, Kjetil Haslum, Karin Sallhammar and Mozhgan Tavakolifard. The Q2S centre and the department of Telematics have provided me with excellent research facilities and offered me a very good environment for carrying out the work, thanks to the technical and administrative staff. I would like to thank all my colleagues at the centre and the department for creating such a nice atmosphere and research spirit. I would like to especially thank my office mates Kjetil Haslum and Mozhgan Tavakolifard for our fruitful discussions and cooperation on research topics.

During the spring semester of 2007 I visited the Information and Networked Systems Security Research (INSS) Group at Macquarie University in Sydney, Australia. I would like to thank Professor Vijay Varadharajan and all the other members of the research group for our collaboration and discussions that led my research into the field of secure ad hoc routing protocols.

Finally, but most importantly I would like to thank my family and friends for all your understanding, support and love.

Marie Elisabeth Gaup Moe
Trondheim, Norway
January 2009

Contents

Publications Included in the Thesis

These papers are included as Part II of this thesis. Note that some of the papers have been subject to minor editorial changes since their publication.

- PAPER A:
  André Årnes, Karin Sallhammar, Kjetil Haslum, Tønnes Brekne, Marie E. G. Moe and Svein J. Knapskog. *Real-time Risk Assessment with Network Sensors and Intrusion Detection Systems.* In Proceedings of the 2005 International Conference on Computational Intelligence and Security (CIS'05). Springer. Xian, China. December 15-19, 2005.

- PAPER B:
  Kjetil Haslum, Marie E. G. Moe, and Svein J. Knapskog. *Real-time Intrusion Prevention and Security Analysis of Networks using HMMs.* In Proceedings of the Fourth IEEE LCN Workshop on Network Security (WNS 2008). IEEE. Montreal, Canada. October 17, 2008.

- PAPER C:
  Marie E. G. Moe *Quantification of Anonymity for Mobile Ad Hoc Networks.* In Proceedings of the 4th International Workshop on Security and Trust Management (STM 08). Elsevier. Trondheim, Norway. June 16-17, 2008.

- PAPER D:
  Marie E. G. Moe, Bjarne E. Helvik and Svein J. Knapskog. *TSR: Trust-based Secure MANET Routing using HMMs.* In Proceedings of the 4th ACM International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2008). ACM. Vancouver, Canada. October 27-31, 2008.

- PAPER E:
  Marie E. G. Moe, Mozhgan Tavakolifard and Svein J. Knapskog. *Learning Trust in Dynamic Multiagent Environments using HMMs.* In Proceedings of The 13th Nordic Workshop on Secure IT Systems (NordSec 2008). Copenhagen, Denmark. October 9-10, 2008.

- PAPER F:
  Marie E. G. Moe, Bjarne E. Helvik and Svein J. Knapskog. *Comparison of the Beta and the Hidden Markov Models of Trust in Dynamic Environments.* To appear in the Proceedings of the third IFIP WG 11.11 International Conference on Trust Management (IFIPTM 2009). Springer. West Lafayette, USA. June 15-19, 2009.

Other Publications by the Author

- Lillian Kråkmo and Marie E. G. Moe. *An Attack on the Stream Cipher Whiteout*. In Proceedings of the 9th Nordic Workshop on Secure IT-systems (Nordsec 2004). Helsinki, Finland. November 4-5, 2004.
- Danilo Gligoroski and Marie E. G. Moe. *On Deviations of the AES S-boxes when Represented as Vector Valued Boolean Function*. International Journal of Computer Science and Network Security (IJCSNS). Vol. 7, No. 4, 2007.
- Venkat Balakrishnan, Vijay Varadharajan, Uday Tupakula and Marie E. G. Moe. *Mitigating Flooding Attacks in Mobile Ad-hoc Networks Supporting Anonymous Communications*. In Proceedings of the 2nd International Conference on Wireless Broadband and Ultra Wideband Communications (AusWireless 2007). IEEE. Sydney, Australia. August 27-30, 2007.

I

THESIS INTRODUCTION

Introduction

We are moving towards a future where dynamic networks becomes an increasingly important part of our daily life. The Internet is already an essential ingredient in the functioning of our modern society. Emergent networks like mobile ad hoc networks, sensor networks, opportunistic networks, peer-to-peer networks and social networks are introducing new and exciting opportunities of communication between people and devices. On the other side of the coin, these dynamic networks also introduce many security- and privacy-related challenges. Examples of situations where innocent persons can have their security and privacy compromised, when using such communication networks are:

- A bluetooth enabled mobile phone downloads what appears to the user to be a security update, but what is really installed is malware that sends a copy of all her call logs and text messages to a remote server.
- A person clicks on a link to a funny picture sent to him by his 'friend' that he is chatting with on a social network. The url that he is clicking on executes a malicious script in his web browser that steals session cookies. His 'friend' may now proceed with hijacking the browser's sessions, potentially impersonating him by gaining access to the web applications he was logged into.
- A person is surfing the web using her wireless home network and clicks on a link to an interesting website. This website loads a malicious script in the user's web browser that takes advantage of a security weakness related to the universal plug and play feature of her home router. The malicious script reconfigures the router so that it uses a DNS server controlled by attackers. The next time the user types in her bank's url, the router redirects her to a phishing website that collects her login details.

These examples illustrate that users might have had their security or privacy compromised without their knowledge at the time of the compromise, and potentially without ever being made aware of this. When dealing with complex and dynamic environments, information about the current level of security or privacy, expressed in a quantified manner, could be of great value in a decision-making process. An example is the addition of information about the trustworthiness of the source of an application, presented to the user before she makes a decision about downloading or not. In order to derive such quantified security-, privacy- and trust-related measures, we need to have mathematical models for security, privacy and trust. The development, application and evaluation of such models is the topic of this thesis.

The main part of this thesis, Part II is a collection of six papers, Paper A-F. Part I gives an introduction to the areas of research covered in the papers. The different topics and application areas of the papers is illustrated in Figure 1

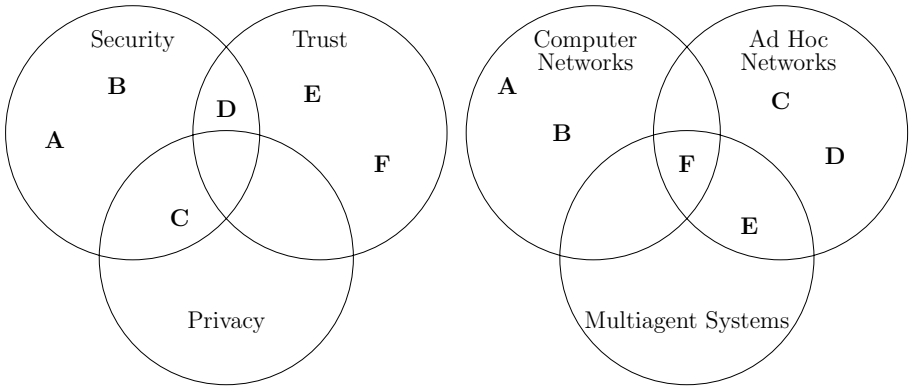


Figure 1. The topics and application areas of the Papers A-F

The introduction is organized as follows. First, a general background for the work is presented in Section 1. Then, the mathematical modeling approaches used in this thesis are presented in Section 2. An overview of related work is given in Section 3. The research methodology is discussed in Section 4. Section 5 provides short summaries of the papers and identifies the author's contributions. Finally, a summary and conclusions are provided in Section 6.

1. Background

This section gives a short background for the work presented in this thesis. First, some general approaches for increasing computer network security are presented in Section 1.1. Then, mechanisms for securing ad hoc routing are presented in Section 1.2. Section 1.3 gives a short overview of trust in multi-agent systems. Finally, a short overview of challenges related to privacy in ad hoc networks is given in Section 1.4.

1.1 Computer Network Security

Examples of potential threats against the security of computer networks and the services they provide, are:

- Eavesdropping of wired or wireless communication, leading to potential disclosure of sensitive information and loss of confidentiality.
- Unauthorized access to a database, leading to potential loss of integrity and confidentiality of data.
- Packet flooding of a web server, which may lead to denial-of-service to customers and loss of reputation for a company.

Traditional security mechanisms, like authentication, encryption and access control are implemented to prevent attackers from gaining unauthorized access

to computer networks. However, these counter-measures may be circumvented given enough time, effort and money. Since computer networks are vulnerable to attackers, both coming from inside and outside the computer network, there is a need for monitoring the network with the purpose of detecting and possibly preventing security compromise. Normally, an attack is detected because it has been successful, and we observe the consequences. To prevent an attack from being successful, it must be anticipated or detected in an early phase, before any consequences are observable. In the long run, attacks may indeed be prevented by convincing potential attackers that your defense is so strong that any intrusion attempts are futile.

Firewalls and intrusion detection systems (IDSs) are traditionally used for stopping suspicious traffic arriving to the network and for detection of suspicious activity inside the network. Firewalls make use of static filtering rules. With a proper configuration the most obvious malicious traffic can be detected and denied to enter a network, but the more clever attackers can find ways to circumvent these rules. Attacks at end users and organization internal services launched from inside an organization can also not be hindered by a firewall.

IDSs may discover attacks based on anomaly detection or signature matching, but have no protection mechanisms against these attacks. Intrusion prevention systems (IPSs) are proactive defense mechanisms designed to detect malicious packets embedded in normal network traffic and stop intrusions, blocking the offending traffic automatically before it does any damage. Detection mechanisms are often based on blacklists, whitelists and thresholds. Examples of prevention mechanisms are response techniques that stop the attack itself by terminating a network connection or a user session or by blocking access to a specific target. The IPS can also change the security environment by reconfiguration of a firewall, router or switch to prevent access from specific attacker IP addresses.

One challenge related to intrusion detection and prevention is the amount of information that needs to be handled manually by a network administrator, in the form of IDS alerts. The problem of *false positives*, i.e. false IDS alerts, often makes manual monitoring an overwhelming task. The objective of an IPS is to minimize this manual handling of alerts without increasing the amount of *false negatives*, i.e. not suppressing the real alerts.

1.2 Secure Routing in Ad Hoc Networks

Mobile ad hoc networks (MANETs) consist of a set of nodes with a dynamic behavior. They can join and leave the network at will, and quite often continuously change their physical location. MANETs are self-organizing and can be formed independently of fixed infrastructure. Some examples of MANETs are:

- A collection of laptops in the classroom communicating with each other using standard Wi-Fi.

- A bluetooth-enabled network between PDAs with the purpose of sharing files at a meeting.
- A radio communication network established at an accident scene in a remote area to facilitate communication between the different emergency response units.
- A vehicular network of cars communicating with each other and roadside equipment which may be connected to the Internet.

The network can also be dynamic in terms of varying trustworthiness of the interacting parties, and no centralized trusted third party can be assumed to be present at all times. The nodes may also have limited computing capacity. All these features of ad hoc networks lead to security challenges for the routing mechanisms.

Security threats against MANETs could be routing disruption attacks, e.g. replay attacks, black holes or wormholes, denial of service attacks or location disclosure attacks. Routing disruption attacks can be defended against by using cryptographic methods in order to provide integrity and authenticity of the routing messages. Some suggested protocols, like ARAN [SDL+02] uses certificates and digital signatures, while others rely on symmetric keys, message authentication codes and one way hash-chains, like SRP [PH02], SEAD [HJP02] and Ariadne [HPJ02].

Even though integrity and authenticity of routing messages can be offered to some extent with existing protocols, the problem of nodes acting selfishly by selectively dropping packets and not sharing their bandwidth with the rest of the network, remains to be solved. The performance of ad hoc routing protocols without any security extensions is severely degraded in the presence of such misbehaving nodes.

If we accept the fact that we may have malicious nodes in our system, the challenge is to detect them and find a way to monitor their behavior and possibly influence their actions, in order to prevent them from causing any harm. Trust management serves this purpose by evaluating the trustworthiness of nodes and offering different service levels to them based on a trust policy. An incentive for nodes not to misbehave is created if network services are denied to untrusted nodes.

1.3 Trust and Multiagent Systems

A *multiagent system* is a collection of mobile agents acting on the behalf of humans or assisting human users in their decision-making. An agent is an autonomous computer program capable of sensing its environment and responding in a timely fashion to environment changes and also taking initiatives in order to realize its objectives. A dynamic multiagent environment could for instance be ad hoc networks, opportunistic networks, peer-to-peer networks, social networks or networks for electronic commerce. Application of

autonomous agents in large-scale open distributed systems presents a number of new challenges such as:

- Agents with different characteristics can enter the system and interact with one another.
- Each agent tries to maximize its individual utility because it represents a specific stakeholder with various objectives.
- Agents may change their identities on re-entering the system to avoid punishment for any wrong doing in the past.
- Agents should decide how, when, and with whom to interact without any guarantees that the interaction will actually achieve the desired benefits.

Agents are faced with significant degrees of uncertainty in making decisions since it is impossible to obtain perfect information about the environment and the properties of the interaction partners. If agents are able to establish appropriate *trust* in each other, the decisions-making process would be facilitated and the risk associated with the interactions could be assumed to decrease.

Computational trust and reputation models seek to quantify trust as a value derived from previous direct experiences and/or second-hand information, such as recommendations, and suggest mathematical and logical expressions for how to combine several opinions about trustworthiness into reputation values. Such models are clearly needed in the virtual world where non-human agents are making trust-based decisions. Also when a human end-user is making the decisions, such calculated trust values can be very useful as decision support.

1.4 Privacy in Ad Hoc Networks

Privacy has become an increasing concern for users of communication services. As communication networks are becoming more diverse and complex, users might lose control over their private information more easily. The widespread use of mobile devices allowing for the continuous tracking of a person's location adds new aspects to the concerns related to privacy in communication networks. Users in dynamic networking environments like mobile ad hoc networks would be particularly exposed to threats against their privacy since they have limited control over the trustworthiness of network nodes that handle the messages sent. Appropriate privacy enhancing cryptographic mechanisms, which can be trusted to work as intended, are required to handle this problem.

Anonymous routing protocols for mobile ad hoc networks aims at hiding the identity of the nodes participating in the network and preventing the location of these nodes from being revealed, including the identity of the users connected to the nodes. Some examples of such protocols are the ones suggested by Zhang et al. [ZLL05], Boukerche et al. [BEKXK04], Kong and Hong [KH03]

and Seys and Preneel [SP06]. Most proposed protocols use a variant of *onion routing*, where messages are wrapped in layers of encryption with the keys of all intermediate nodes on the route to the destination. At each node a layer of encryption is peeled off before the node forwards the messages in random order. The privacy of the sender and the receiver of a message relies on the fact that there should be no correspondence between incoming and outgoing messages from a node. In practice an external passive global adversary could just track the flow of messages through the network. To prevent this, an addition of dummy traffic and different mixing strategies are applied as extra measures beside the routing protocol.

There are many challenges when constructing privacy-preserving protocols for MANETs. There is always a trade-off between the cryptographic strength of a scheme and its efficiency. MANETs may have constrained computational resources and the routing function may deteriorate under delays due to the heavy computations required for the security functionality of the protocols. On the other hand, reasonably strong privacy is desired. This means that there is a need for measuring the amount of privacy that is offered by the scheme.

2. Modeling Approaches

This section describes the mathematical modeling approaches used in this thesis, and the different measures of security, privacy and trust that can be derived from these models. First, the modeling of security in computer networks is discussed in Section 2.1. Then the modeling of trust in dynamic environments is discussed in Section 2.2. Finally the modeling of privacy in ad hoc networks is discussed in Section 2.3.

2.1 Modeling of Security in Computer Networks

In order to obtain quantitative measures of security, we use a *state* modeling approach which has traditionally been used to model dependable systems. We base our modeling on the view that the notions of security and dependability are integrated concepts, both describing aspects of trustworthy computer systems. Such an integrated modeling of security and dependability has now been commonly accepted in the research community. Some examples of suggested models can be found in [Mea95, MVT02, ALRL04, NST04, Jon06, Sal07].

The *dependability* of a system can be defined as its *ability to deliver service that can justifiably be trusted* [ALRL04]. Following the taxonomy proposed in [ALRL04] dependability can be decomposed into the aspects: *availability, reliability, safety, integrity and maintainability.* Security is decomposed into the aspects: *availability, confidentiality and integrity.* Dependability is an integrating concept that focuses on the correctness of services and the absence of *faults*, which can be either *natural*, i.e. hardware faults caused by natural phenomena, or human-made but *non-malicious*, i.e. performed without

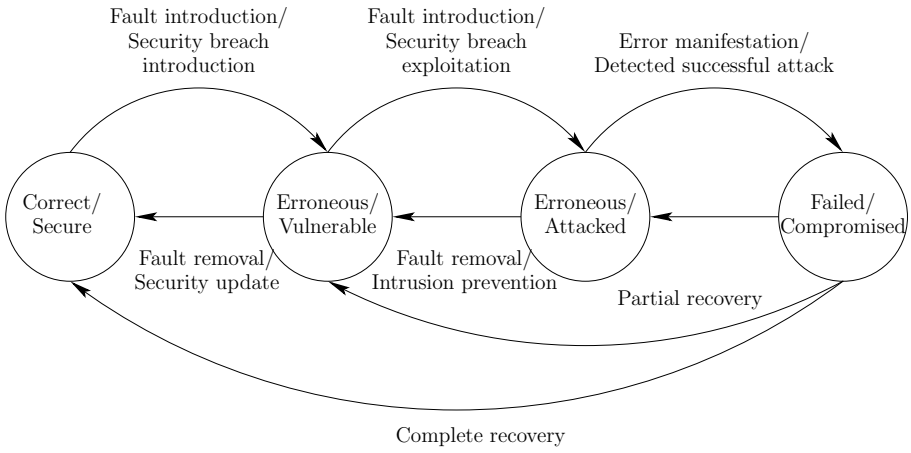


Figure 2. A state model for computer network security using the fault-error-failure terminology

malicious intentions. Security, on the other hand, focuses on correctness of services to *authorized* users and the absence of *vulnerabilities*. A vulnerability can be defined as an *internal fault that enables an external harm to a system*, it can be either natural or human-made, both malicious and non-malicious. An example of a natural vulnerability is non-tamperproof hardware that allows for side-channel attacks. A non-malicious human-made vulnerability could for instance be the presence of software flaws allowing for buffer overflow, caused by a programmer's mistake or ignorance. An example of a malicious human-made vulnerability could be the installation of a backdoor trojan on a computer, performed by an intruder to allow stealthy remote access to the computer for future attacks.

In the state modeling of security in computer networks we make use of the *fault-error-failure* terminology known from the dependability modeling literature, as described in [ALRL04, Jon98, JSL00, Jon06]. This is illustrated in Figure 2. The *secure* or *correct* state is defined as the state when the system is in adherence to the security policy, and there is an absence of vulnerabilities. When a computer network starts operating it is most likely already in the *vulnerable* state, since the complexity of such a system makes it very hard to avoid vulnerabilities, e.g. implementation or configuration faults. A vulnerable state could also be denoted as an *erroneous* state in the dependability terminology, since there are faults present which makes parts of the system deviate from the correct state. These faults can be *dormant*, i.e. they have not yet been activated causing any errors, or they could have resulted in *latent* errors that have not yet been detected. Errors in the system may lead to a *failure* when they cause the service delivered to the environment, i.e. users or other computer systems, to deviate from the correct service in an unac-

ceptable way. Such an error manifestation puts the system in the *failed* state. Failures vary in severity and duration. In order to transfer the system into a correct state a complete recovery has to be performed. This can be done by different means depending on the nature of the error that caused the failure, e.g. rollback of databases, reconfiguration, repair or replacement of physical components. In most cases such a recovery requires manual interference, also a *diagnosis* might be needed to identify the cause of errors. In many cases not all internal faults leading to the errors which manifested themselves during the failure are removed in such a recovery process. This means that we only have a partial recovery where the system is up and running again, but still in an erroneous state.

In the context of security, we can refer to faults as *security breaches*, since we are concerned about the vulnerabilities that can be exploited by attackers, i.e. unauthorized users or computer programs, in such a way that the service delivered by the system deviates from the security policy. Security breaches can be introduced into the system in the development phase or during operation by internal developmental or operational faults, or they can be introduced by external attackers. A security breach can be compared to a dormant fault or a latent error. It is something that makes it possible for an attacker to launch an attack on the system. Security breaches might be removed from the system before they cause any harm by *security updates*, for instance in the form of security patching, security upgrades, reconfiguration or replacement of components.

The moment a security breach is starting to be exploited by an attacker, the system moves into the *attacked* state. Such an intrusion attempt might be stopped by *intrusion prevention* mechanisms in the system before any harm is done, or the attacker may simply decide not to proceed with the attack. This leads us back to the vulnerable state. When a security breach is being exploited and the system lacks means to withstand the attack, this may lead to a *security failure*. This puts the system in a state where we have a *detected* unacceptable deviation from the security policy, e.g. the confidentiality and integrity of data in a database has been compromised. We denote this state the *compromised* state. An attack might also be successful but *undetected*. This means that the system state is deviating from the security policy but this has not yet been signaled to the environment by deviation of the delivered service. In this case the system stays in the attacked state until the attack is detected. Depending on the nature of the attack and its consequences the system state may then go to the compromised state.

The state model shown in Figure 2 is very general and could be divided into more states for a more refined model. In Paper B we split the attacked state into two different states, called *intrusion attempt* and *intrusion in progress*. This is done because we consider intrusion attempts like port scans and automated scripts for password guessing as very common events that should be distinguished from the real intrusions. A real intrusion could for instance be

when a person logs on to the system using one of the guessed passwords from an automated script and starts taking control of larger parts of the network. In both Paper A and Paper B we merge the *secure* and the *vulnerable* states into one single state, since we consider the probability of the computer network being in a completely secure state as very low.

The state modeling allows for a probabilistic evaluation of the security of the system, which can be used for security quantification, prediction, risk assessment, intrusion detection and intrusion prevention. In Paper A the security state of a computer network is modeled using a first order Markov model (Λ, π). The Markov model consists of a set of states $S = \{s_1, s_2, \ldots, s_N\}$, an initial distribution $\pi = (\pi_i)$, describing the state of the system when monitoring starts, and a transition rate matrix $\Lambda = (\lambda_{ij})$, describing the dynamics of the system. A method for quantitative real-time risk assessment is proposed based on observations from *sensors*, such as intrusion detection systems. In this paper the idea of using a *hidden Markov model* (HMM) [Rab90] for the modeling of the security state of each individual monitored network component is introduced. Each network component may be monitored by several sensors, and the output of the sensors could vary in accuracy. The *trustworthiness* of a sensor is its ability to give correct observations about the security state of the monitored network component. Since the real security state is not directly observable, the true state is considered as *hidden* and the HMM approach is used for estimating the probability distribution $\gamma_k = (\gamma_k(i))$ over the states, where $\gamma_k(i)$ is the probability that the monitored component is in security state s_i at time-step k.

In Paper B the security state of the individual network components are not modeled separately. Instead a common system model is used for the computer network as a whole, while each sensor is modeled by a separate HMM. This means that in a computer network with L sensors, each sensor $\psi \in \{1, \ldots, L\}$ is modeled by a separate HMM (Λ, π, Q^ψ) that includes the common system model (Λ, π), but uses an individual observation probability matrix Q^ψ in order to model specific properties for each sensor. The matrix Q^ψ describes the trustworthiness of sensor ψ. This approach facilitates the aggregation and filtering of alerts from the different sensors, thus minimizing the amount of alerts that needs to be handled manually by a network administrator. Alerts are filtered according to alert severity, and used as observations in the HMM. One common state distribution γ_k is updated for each new observation received from one of the sensors.

Several useful measures can be derived from the Markov models. A measure of *risk* is proposed in Paper A, where a cost value $\mathcal{C}(i)$ is associated with state s_i and the total risk $\mathcal{R}_k$ for a monitored object at time-step k is given as

$$\mathcal{R}_k = \sum_{i=1}^{N} \mathcal{R}_k(i) = \sum_{i=1}^{N} \gamma_k(i)\mathcal{C}(i).$$

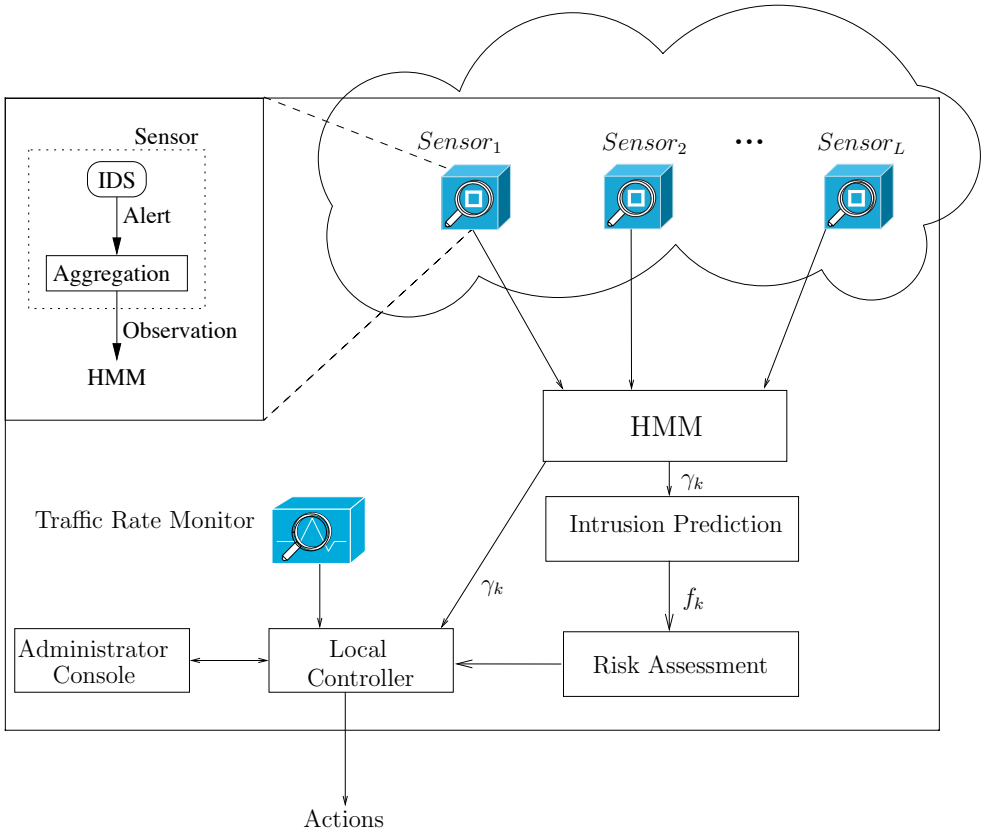


Figure 3. Architecture of an intrusion prevention system (from Paper B)

In paper B the mean time to next security failure (MTNSF) is derived from the Markov model. This is done by partitioning the statespace S into *good* S_G and *failed* states S_F, making the good states transient and the failed states absorbing. The time to absorption can then be derived analytically by estimation of the average number of times each of the transient states S_G is visited before the Markov chain reaches an absorbing state, denoted w_i, together with the estimated mean state occupation times h_i. MTNSF is a dynamic measure since the state probability distribution γ_k is used as initial distribution when the time to absorption is calculated. A measure of the *intrusion frequency* f is found by inverting MTNSF,

$$f = (MTNSF)^{-1} = \frac{1}{\sum_{i \in S_G} w_i h_i}.$$

This measure can then be used as input to a risk assessment component, integrated in an intrusion prevention system, as shown in Figure 3.

2.2 Modeling of Trust in Dynamic Environments

When we think about trust relationships in the real-life world, we base our interaction choices on our *opinions* about the intentions of other persons. This opinion is influenced by the outcome of previous direct interactions or second-hand opinions in the form of recommendations or warnings. In cases where we lack information from previous direct interactions it may also be influenced by prejudice, meaning that we base our opinion on feelings and expectations for a group that a person is perceived as belonging to.

In the virtual world we can also have opinions about other users, much in the same way as we would have opinions of a person that we know in real life. Normally the interaction partners that we are dealing with in this environment are in the form of a computer program, a website, a network service provider, etc. Also, the decision maker in the virtual world might not be a person, but a computer program acting on the user's behalf. Since the entities involved in the trust negotiation are not necessarily persons, we refer to them as *agents*.

The probabilistic view of trust that is commonly agreed upon, and that is most suited to the scope of our trust modeling, is based on the social scientist Gambetta's [Gam88] definition of trust:

> Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action.

This is a *predicitive* notion of trust upon which we also base our proposed trust calculations. As the definition above states, trust is a subjective measure, where different agents (trustors) might predict different probabilities concerning the same agent (trustee). *Reputation* is often described as the objective measure of trust. It is the general agreed upon opinion about an agent's trustworthiness in a community, based on ratings or recommendations from its members.

The model presented in Paper E consists of trust estimation and trust learning modules. The former and latter are constructed from *hidden Markov modeling* and *reinforcement learning (RL)*, as illustrated in Figure 4. The model parameters of the HMM are re-estimated after having learnt about its environment from the reinforcement learning module. The proposed method enables the improvement of the model reliability when dealing with a dynamic environment that changes over time.

We model the agent interaction as a stochastic process. The *state* of an agent can be characterized by whether or not it is behaving in a malicious manner in its interactions with other agents. When agents are interacting, an agent makes its opinion about the trustworthiness of the other agent based on the outcome of the interaction. After a random time interval these two agents meet again and based on their belief about the other agent's trustworthiness, they may decide whether or not to make an interaction. Since an agent's

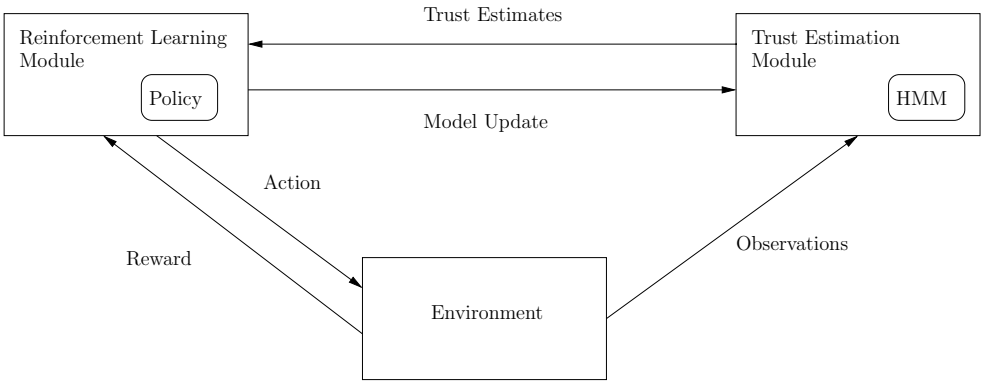


Figure 4. Architecture of the trust model (from Paper E)

behavior can be changing with time it is not necessarily the case that an agent is in the same state as it was at the last encounter. An agent can only do its best guessing about the trustworthiness state of an other agent based on its own previous direct experiences with said agent and *recommendations* from other agents in the system. This means that the system state is hidden, and hence we use the HMM approach, similarly to the modeling approach in Paper A and B. The system we consider now is a multiagent system and we want to use the model to estimate the behavior of each single agent. An agent in the system rates the performance of all of the other agents with respect to their behavior after an interaction, and uses an HMM per agent to decide and predict whether or not another agent is malicious. The trust estimate is given as the probability that an agent is in the *trusted* state, according to the HMM. The HMM is updated from observations, in the form of ratings after direct experiences, or recommendations requested from other agents.

The parameters of an HMM are usually set by offline training of the model with a large data set. Since we want the model to reflect the dynamic behavior of the multiagent system and also optimize the agents' trust-related behavior, an online learning of the HMM parameters with *reinforcement learning* is proposed in Paper E. Reinforcement learning (RL) [SB98] is a machine learning technique for solving decision problems of mapping actions to states based on interactions with the environment. The actions of an agent in the multiagent system could for instance be a result of a decision of whether or not it should interact with another agent, based on its belief about the state of the other agent derived from the HMM. Such a mapping from state to action is called a *policy*. In RL the agents learn policies based on feedback from the environment that is calculated based on a *reward function*. The RL framework also includes a *value function* $Q(s, a)$ which estimates the reward obtained if action a is performed in state s. *Q-learning* [WD92] is a well-known RL algorithm that updates the value function in each step so that the agent policy converges

to the optimal one. Q-learning works even though the state transition probabilities are unknown to the agent. In our approach we use the output of the Q-learning to improve the HMM by updating the state transition rate matrix according to the learned optimal policy.

The learning proceeds as follows: When an agent encounters another agent it derives the current state probability distribution γ_k belonging to this particular agent from its corresponding HMM and execute the action with the largest $Q(\gamma_k, a)$. After the action is performed, the agent receives the reward, the next step state probability distribution γ_{k+1} is derived from the HMM, and the Q-learning updating rule is applied. The process is repeated at the next encounter between the agents. As the agent learns about the behavior of the other agent through direct experience and recommendations, the HMM parameters should be updated in order to improve the predictiveness of the model. In Paper E we suggest that the state transition rates Λ and the observation probabilities of the HMM are updated after a predetermined number of Q-learning steps by the Baum-Welch algorithm, which finds the maximum likelihood parameter estimate.

An application of the trust-estimation approach using HMMs is given in Paper D. In this paper a trust-based security extension to the mobile ad hoc network dynamic source routing protocol (DSR) is proposed, called TSR (Trust-based Secure Routing). The state probability of a node, according to its corresponding HMM, is being used for deciding the node's trustworthiness, and the routing protocol is extended with a black-listing of nodes that are not deemed as trustworthy. This approach prevents the selective packet dropping behavior of selfish nodes that could not be detected with previous suggested solutions. Packet-dropping nodes are excluded from the network, thus enforcing cooperation among nodes. Each node in the MANET monitors all of its neighbors and uses an HMM to decide and predict whether or not each neighboring node is selfish. This monitoring framework is based on a similar theoretical foundation as the one described in Paper B. The TSR architecture is illustrated in Figure 5.

A node in the network can be in a *benign* or *selfish state* at any given time. The packet-sending behavior of the node can be modeled as a stochastic process as it is probabilistic and described in terms of state transitions that are triggered by events which happen randomly according to a probability distribution. A node is selfish if it is dropping packets, not participating in the forwarding of other node's packets because it wants to keep all its resources and bandwidth for itself. It can also refuse to participate in the route discovery part of the routing protocol, making sure that it is not on the path of any route originating from another node than itself. This can also be referred to as a *passive* DoS attack.

It is assumed that when nodes join the network there is a small probability that this node is in a selfish state, but most likely it is in the benign state. After joining the network the node has a probability of becoming selfish or

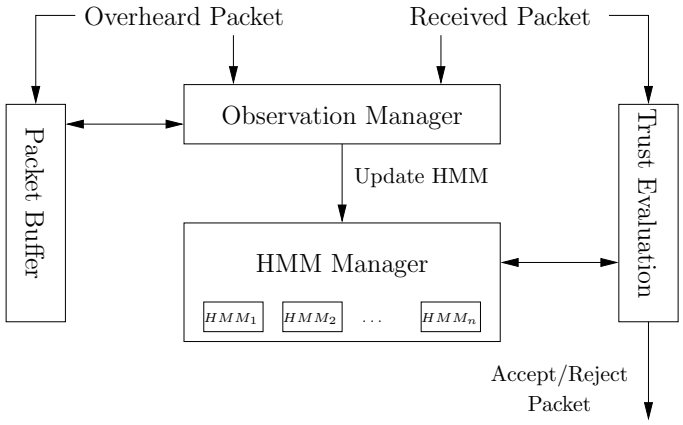


Figure 5. The TSR architecture (from Paper D)

malicious, e.g. it could be hi-jacked by an intruder or the intentions of the person controlling the node may change. One example could be that a node is acting benignly, forwarding packets for other nodes, but turns selfish and starts dropping packets as soon as its available energy falls below a certain threshold. When a node is dropping packets, this could mean that the node is acting selfishly, but it could also be that the node is accidentally dropping packets due to contention or congestion problems or broken links. Since a node could be legitimately dropping packets without malicious intents, an observer to the system cannot be certain if a node is selfish or benign, so the system state is hidden to the observer.

As seen in Figure 5, every node in the network has its own local trustworthiness rating of all other nodes in the network, derived from the HMMs. Observations that are used as input to the HMMs, can either be *directly* observed, by overhearing the transmissions of nodes in the environment of the monitoring node, or *implicitly* derived from the source routes on received packets, and from received or overheard route error packets. Whenever a node sends a packet, it will monitor the next hop node to see if the packet is forwarded. If it overhears a transmission of the packet from its neighbor, this is interpreted as an observation in the corresponding HMM in the list of HMMs that this node keeps updated for all of the network nodes. When a node receives a data packet it will check the trustworthiness of the source, destination, previous hop and next hop node, and only accept and forward the packet if all these nodes are deemed to be trusted nodes, i.e. their trustworthiness is above a certain threshold value. We refer the reader to Paper D for further details of the routing protocol.

In Paper F we take the computational trust component of Paper D and E and put it into a more general *reputation system*, as illustrated in Figure 6. This is a decentralized reputation system where each agent keeps and updates

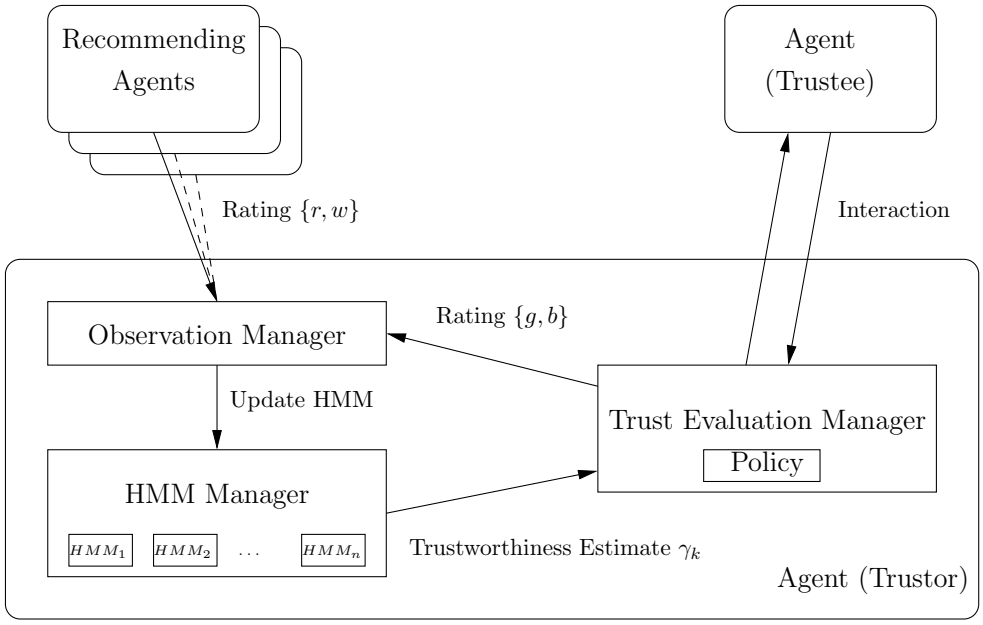


Figure 6. The architecture of a reputation system using HMMs (from Paper F)

HMMs, which are modeling the trust state of the other agents. Before an agent (trustor) initiates an interaction with another agent (trustee), it looks up the trustworthiness estimate derived from the HMM belonging to the trustee. The *trustworthiness estimate* is given as the probability that the agent is in the trusted state $\gamma_k(s_1)$. The trustor then decides according to a policy whether or not to interact with the trustee. After an interaction, the HMM belonging to the trustee is updated with a rating, good g, or bad b, based on the outcome of the interaction. The HMMs are also updated with observations in the form of ratings from other agents in the system, so called second-hand opinions, which are either in the form of recommendations r, or warnings w.

Trust and reputation is an active field of ongoing research, it has been proposed numerous different models for quantification and evaluation of trust. The modeling complexity varies, ranging from very simple eBay-like models, where a reputation value is calculated as the sum of ratings, to more sophisticated models based on probability theory, e.g. the Bayesian trust and reputation models [MMH02, JI02, BLB04, NKS07, JH07]. Bayesian trust models are based on the assumption that the behavior of an agent can be described according to a probability distribution. The trust value is a function of the expected value of the probability distribution, which is updated with every new rating received according to Bayes' Theorem. Binomial Bayesian reputation systems, where ratings can be expressed by two values, *good* or *bad*, are modeled with the Beta probability density function. Multinomial Bayesian

reputation systems, that allow for ratings with graded levels, are modeled with the Dirichlet probability density function. The objective of Paper F is to evaluate our proposed hidden Markov trust model, by comparing it to one of these trust models, namely the Bayesian binomial model of the *Beta reputation system* [JI02]. The reason for choosing this model is that it is based on a probabilistic view of trust, similarly to our model. It is also one of the trust models in the literature that seems more adaptable to dynamic networking environments where the behavior of agent's may change over time, as it includes a *forgetting factor* that discounts old ratings. The comparison study is done with the help of simulations of different trust scenarios, in order to see which trust and reputation system performs best with regard to reliability of the calculated values under various hostile agent strategies. From the simulation results we see that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents, due to its larger richness in model features. This means that our trust model may be more realistic in dynamic environments. However, the increased model complexity also leads to bigger challenges in estimating parameter values for the model. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Beta reputation system.

2.3 Modeling of Privacy in Ad Hoc Networks

The model for anonymous ad hoc routing introduced in Paper C is a probabilistic information theoretical model based on the models for anonymity in mix-networks proposed by Diaz et al. [DSCP02] as well as Serjantov and Danezis [SD02]. The novelty of our approach is the application of the conditional entropy measure of anonymity in ad hoc networks, that the a priori knowledge of the attacker is taken into account and the inclusion of the quantification of the amount of additional information the attacker will gain by taking over more nodes in the network.

In order to properly define secure anonymous routing it is useful to have a security model that represents the system and to have some sort of measure that can quantify the amount of anonymity offered by the protocol. The definitions by Pfitzmann and Hansen [PH05] are generally accepted:

- *Anonymity* is the state of being not identifiable within a set of subjects, the *anonymity set*.

- *Unlinkability* of two or more items within a defined system means that these items are no more and no less related than they are related concerning the a-priori knowledge.

For the ad hoc routing setting an anonymous routing protocol should ideally offer sender and recipient anonymity, meaning that the sender of a message or recipient of a message remains unidentifiable under the assumed attacker model. We also want to achieve *relationship anonymity* between the sender

and the recipient of a message, so that an observer cannot determine which nodes are taking part in a specific communication flow. In other words, sender and recipient are unlinkable. A sender or receiver of a message in a MANET could be identified in terms of the *node identifier*, e.g the MAC address of a MANET device, or the *user identity*, e.g. the name of the person controlling the device. The sender or receiver could also be identified in terms of *node location*. The method of analysis proposed in Paper C focuses on quantifying anonymity with regard to *location privacy* of the nodes in the network.

The classic way of quantifying the degree of anonymity for a user, within a set of users that could be the potential senders/receivers of a message, is done by simply measuring the size of that set, the *anonymity set* [Cha81]. The size of the anonymity set is intuitively an indication of the degree of anonymity, as more members of the set of potential senders/receivers, reduces the probability that a randomly chosen member of the set was the actual sender/receiver. However, we should also take into account that anonymity is stronger the more evenly distributed the sending and receiving of messages by the subjects within that set is.

Yet another way the degree of anonymity could be quantified is in terms of the information theoretical *entropy* of the probability distribution that the attacker assigns to each possible sender as being the originator of a message, after observing the system. In a system with N users, let p_i be the probability for user i to be the sender/recipient of a message, assigned by the attacker, and let X be the discrete random variable taking the possible values $x_1, x_2, \ldots, x_N$ with probabilities $p_1, p_2, \ldots, p_N$ respectively, the entropy $H(X)$ of the probability distribution can be calculated by

$$H(X) = -\sum_{i=1}^{N} p_i \log_2(p_i).$$

The entropy can be interpreted as the number of bits of additional information that the attacker needs in order to definitely identify a user, or as the effective decrease in uncertainty. This information-theoretic measure of anonymity was proposed independently by Diaz et al. [DSCP02] and Serjantov and Danezis [SD02]. For quantification of the degree of anonymity Diaz et al. compared the information obtained by the attacker after observing the system against the optimal situation where all users are equally likely to have sent/received the message. The degree of anonymity is denoted d and defined as

$$d = 1 - \frac{H_{\max} - H(X)}{H_{\max}} = \frac{H(X)}{H_{\max}},$$

where $H_{\max} = \log_2(N)$ is the maximum entropy for the system.

Entropy may be used as a measure of how evenly the probabilities are distributed within each distribution, but two distributions with the same entropy could still have very different qualitative anonymity. Another problem of the

entropy measure is that it does not take into account the a priori knowledge of the attacker. In an ad hoc routing setting, if we consider an attacker that has both a global and local perspective on a network, we could imagine that the attacker has some a priori knowledge of the communication patterns of network nodes, derived from traffic analysis or from application-layer contexts. The global attacker could for instance know the frequency of route request transmissions from all nodes, which give rise to a probability distribution over the potential senders of a particular message. This a priori knowledge could then be combined with the information the attacker obtains by local observations, as suggested by Clauß and Schiffner [CS06]. As noted by Diaz et al. [DTD07], the problem of how to combine the entropy measures from two different sources has not yet been adequately addressed or fully solved. It is not necessarily true that the entropy decreases when an attacker gets access to more information in a given attack scenario. However, if we take the weighted average of all possible entropies that the attacker can obtain after observing the system, given the a priori knowledge, this entropy, defined by Shannon as the *conditional entropy*, will always be equal to or less than the entropy of the a priori probability distribution.

We are interested in knowing how many nodes in the network can be overtaken by an attacker before the anonymity offered by the routing protocol becomes unacceptably low. In order to achieve this we need to have a quantification of the anonymity offered by the protocol in relation to the number of compromised nodes as well as the total number of nodes in the network. We propose to use the term Passive-c/n for an attacker that is an external passive local or global attacker for the whole network, which is consisting of N nodes, of which this attacker can eavesdrop on the communication of a subset of n nodes, and that has compromised or owns c nodes inside the network. In other words, the local or global external attacker cooperates with a local internal passive attacker that controls c nodes. As we are focusing on the anonymity aspects of the routing protocol in our model, we do not take into account an *active* attacker that could inject, drop or modify packets in order to disturb the routing mechanisms or to launch a denial of service attack. We assume that the attacker carries out a probabilistic attack. This means that the attacker obtains a probability distribution over which of the sender or recipient nodes in the network that could have sent or is the recipient of a particular message.

When evaluating the anonymity offered by a routing protocol, we are interested in knowing how resistant the protocol is against possibly colluding malicious nodes. To achieve this we measure the anonymity in terms of entropy based on the external global view of the attacker before any nodes have been compromised, and then quantify the average gain in information of the attacker as it controls an increasing number of nodes in the network, using the conditional entropy measure.

Let X be a discrete random variable with probability mass function $p_i = P(X = x_i)$, x_i corresponds to a node N_i in the network and p_i is the probability that N_i will be sending a message m, as viewed by the attacker before any internal nodes have been captured. Let $\mathcal{P}_0$ be the discrete a priori probability distribution with values p_i for $1 \leq i \leq N$ in a network with N nodes. This a priori probability distribution could for instance be based on traffic analysis performed by the global external attacker. The anonymity of the nodes with respect to this external attacker could be measured in terms of the entropy of $\mathcal{P}_0$. In the case where an ad hoc routing protocol is resistant to this kind of analysis by means of extensive use of dummy traffic we could imagine that this a priori distribution is a uniform distribution with entropy $H_{\max} = \log_2(N)$.

Assume that a node N_j is taken over by the attacker and that this node receives the message m. With the internal information of this node the attacker could then possibly gain some new information about which node wherefrom m originated, so that the probability distribution $\mathcal{P}_0$ can be updated to $\mathcal{P}_1$. This new information can for instance be about how many hops away the message originated. If the node internal processing of the message m reveals the number of hops it has travelled or how many hops away to the destination it has yet to travel, the attacker can in the worst case locate the position of the sending or receiving node of this message, e.g. the message originated one hop away.

In our measurement model we want to combine the probabilities assigned to each node in this anonymity set with the a priori knowledge of the attacker, to form the new probability distribution $\mathcal{P}_1$.

Let Y denote the discrete random variable with probability mass function $q_k = P(Y = y_k)$, where q_k is the probability that a message m, received by the attacker node N_j, originated at a node k hops away, according to the knowledge the attacker can derive from the internal information of node N_j. Assume that there is a maximum path length λ in the ad hoc network, measured in number of hops. If we assume the local node attacker to have no a priori knowledge of the probability of other network nodes as being the originator of the received message m, the probability that m originated at a node k hops away is given by:

$$q_k = \frac{c_k}{\sum_{i=1}^{\lambda} c_i},$$

where c_k denotes the size of the anonymity set for a message originated k hops away. The entropy $H(Y) = -\sum_{k=1}^{\lambda} q_k \log_2(q_k)$ will express the attacker's uncertainty of at which node the message m originated, viewed *locally* from node N_j. We will combine this entropy measure with the measure of the a priori *global* view of the attacker using the Shannon *conditional entropy* $H(X|Y)$. The conditional entropy is not a measure of the uncertainty of the attacker in a specific attack scenario, but rather a measure of the attacker's

average uncertainty given all possible local observations:

$$\begin{aligned} H(X|Y) &= -\sum_{i,k} P(x_i, y_k) \log_2 P(x_i|y_k) \\ &= -\sum_{k} q_k \sum_{i} P(x_i|y_k) \log_2 P(x_i|y_k). \end{aligned}$$

The conditional entropy measure is the average entropy of X, given Y, weighted according to the probability of a particular observation y_k. Let Z denote the discrete random variable describing the conditional probability that node N_i originated a message, given the observation y_k. This gives us the anonymity measure

$$H(X|Y) = \sum_{k} q_k H_k(Z),$$

where $H_k(Z)$ denotes the entropy of Z, given the observation y_k. In a specific attack scenario $P_k(z_i) = P(x_i|y_k)$ would be the probability that N_i was the sending node, derived by an attacker who has an a priori knowledge of $\mathcal{P}_0$, and when capturing the message m can see that it originated k hops away.

In the analysis of a specific anonymous routing protocol we need to decide what information would be revealed if a node participating in the routing protocol is captured by an attacker. For protocols using variants of the onion routing technique such information could typically be about the number of hops a message has travelled from its destination. The next step of the analysis would be to determine or estimate the size of the anonymity set, i.e. the set of potential sender/receiver nodes, and then weight the sending probability of each node in the anonymity set according to the proportion of adversarial nodes as well as the a priori sending probability. Paper C provides examples of the application of our proposed method of analyzing anonymity for the two anonymous ad hoc routing protocols ANODR [KH03] and ARM [SP06].

3. Related Work

This section presents previously published research results which are related to the topics of this thesis. Section 3.1 presents related work on security modeling using Markov models. Section 3.2 presents related work on using HMMs for trust and reputation models. Related work on trust-based security of ad hoc routing protocols is presented in Section 3.3. Finally, Section 3.4 presents related work on the quantification of anonymity in routing protocols.

3.1 Security Modeling using Markov Models

Littlewood et al. [LBF+93] introduced the idea of quantifying security with a stochastic modeling approach. Since then several papers on using Markov models for security evaluation of computer systems have been published. Jon-

sson and Olovsson [JO97] conducted an intrusion experiment where students attacked a real system under controlled conditions. Their experiments indicate that an attack can be divided into different phases where time between security breaches is exponentially distributed. The research project SITAR [GPWW+01] presents a generic state transition model to describe the dynamics of intrusion tolerant systems. Madan et al. [MVT02, MGPVT04] derive several measures, like steady state availability and mean time to security failure, using this generic state transition model. Singh et al. [SCS03] use *stochastic activity networks* for quantitative evaluation of intrusion-tolerant systems via simulations. The SITAR project also include simulations, using *stochastic reward nets* [WMT03]. The modeling approach of SITAR differs from our modeling in Papers A and B as they use a static Markov model of the system while we use a dynamic hidden Markov model. Their approach is developed for performing security analysis in the development phase of an intrusion tolerant system, while we focus on the security of an operational system. Since a computer system does not remain static under its operation, a dynamic modeling approach seems preferable.

Markov models have traditionally been used to model and evaluate computer system dependability, Nicol et al. [NST04] argue that some of the modeling techniques used in the dependability community can be applied for security evaluation. Specifically, they suggest that Markov reward models may be suitable to model security aspects of software systems. Sallhammar et al. [SHK07, Sal07] describe a framework for combined security and dependability evaluation of computer networks based on the model in Paper A, but also including stochastic game theory for the modeling of an attacker's behavior.

Hidden Markov models have been used in IDS architectures to detect multistage attacks [OMSH03] and as a tool to detect misuse based on operating system calls [WFP99]. Khanna et al. [KL06] propose to build an IDS based on an HMM with multivariate Gaussian distributed observations, and dynamic re-estimation of parameters. In [KL07], Khanna et al. use distributed HMM processing in combination with a proportional integral differential (PID) control engine to design a distributed IDS for ad hoc networks. Their approach is different from ours as they are modeling only one sensor at a time and are relying on continuous observations of network parameters.

3.2 Hidden Markov Modeling of Trust and Reputation

In [HCD05], a trust model using a Markov model is proposed by Hussain et al. In this work, the Markov chain is defined as the chain of aggregated reputation values corresponding to a sequence of consecutive time slots. The Markov matrix of a given agent denotes the probability of the agent transiting from one trustworthiness level to another based on its past behavior captured using the Markov chain. In order to determine these probabilities, they use the ratio of the number of times that the agent has transited from trustworthiness

level A to trustworthiness level B to that of the total number of times that the agent has transited from trustworthiness level A to any other trustworthiness level. The future state vector of the agent is determined by multiplying the current state vector with the Markov matrix. The same authors also proposed a method for determining the effectiveness of their Markov model for predicting the future trustworthiness value of a given agent by utilizing simulation methods [HCD06]. Their modeling approach differs from our since they use a deterministic state vector, while we derive a probability distribution over the states. This results in a more realistic model, since we include the uncertainties about which state an agent really is in.

Sassone et al. [SKN06] compare the effectiveness of different probabilistic computational trust systems. They conclude that most existing probabilistic trust models are unrealistic, as the models allow for no dynamic behavior, and outline the idea of a trust model based on a hidden Markov model to cope with this problem.

Hidden Markov modeling as a statistical tool has been applied to several trust-related applications. Song et al. [SPX04] have developed a hidden Markov model based approach to measuring an agent's reputation as a recommender in a recommendation network. This approach does not consider how the trust values are calculated. It focuses on the recommendation process and models chained recommendation events as an HMM. The measuring requires information about the topology of the recommendation network as well as all the recommendation events. We have not included recommendation chains in our trust modeling and it would be interesting to extend the model with recommendation chains. However, the approach in [SPX04] is not readily applicable, due to the decentralized nature of our model.

3.3 Trust-based Security in Ad Hoc Routing Protocols

SAR [YNK01] introduces a trust-based approach to MANET routing. It provides an extension to on-demand routing protocols like AODV (Ad hoc On-demand Distance Vector Routing) or DSR (Dynamic Source Routing), that includes a trust metric and allows for a negotiation of trust levels in the routing mechanism. Nodes in the network are assigned different trust levels depending on their protection mechanisms against routing attacks. The trust levels can be defined by distributing one shared secret key among nodes at the same trust level. In the route discovery using SAR, a field indicating the trust level of the route is added to route reply packets. Routes with an assigned trust level are guaranteed to only include nodes with the required trust level. The SAR approach assumes static trust levels.

Watchdog/Pathrater [MGLB00] is a trust-based extension to DSR that aims at detecting misbehaving nodes. The Watchdog method assumes that nodes operate in promiscuous mode. Every node is monitoring its neighbors and measuring the frequency of packet dropping or misrouting and updating a trustworthiness rating of all known nodes. The Pathrater mechanism ensures

that routing paths are chosen based on the nodes' trustworthiness rating, excluding routes with nodes that are identified as misbehaving according to the Watchdog mechanism. As noted by the authors of [BB02], the Watchdog and Pathrater increase the throughput in MANETs whenever packet dropping nodes are present, but fails to enforce node cooperation since nodes that are misbehaving do not receive any form of punishment. When misbehaving nodes are excluded from routing paths they are actually relieved from forwarding packets for other nodes.

CONFIDANT [BB02] is a reputation-based extension to DSR that includes a monitor, a reputation system, a path manager and a trust manager. It aims to detect and isolate misbehaving nodes by combining monitored and experienced information of a node's behavior with warnings reported from other nodes. Cooperation is encouraged by imposing isolation of nodes that are detected as misbehaving, since nodes in the network refuse to forward packets originating from a black-listed node.

Another approach for stimulating cooperation by the introduction of virtual currency was introduced in the Nuglets [BH01] and Sprite [ZCY03] schemes. In these credit based systems nodes are credited or charged for the packets they send, thus providing incentives for cooperation. The downside of Nuglets is that its security relies on the use of tamper-proof hardware in every node. The Sprite approach avoids this but instead introduces a centralized authority thus violating the assumption of a decentralized structure of ad hoc routing protocols. The upside of both of these approaches however, is that they do not require the nodes to be in promiscuous monitoring mode.

Cooperation enforcement with a reputation-based scheme like CONFIDANT or CORE [MM02], where black-listed nodes are denied network services, provides an incentive for node cooperation, but at the cost of increased overhead due to the transmission of recommendations and warnings.

The TEAM [BVTL07] model avoids the communication of extra packets or additional headers for recommendations. Since a node will only forward a packet if its previous hop, next hop, source and destination are trustworthy, the recommendations can be derived implicitly from the routes contained in the packets. This approach removes the problem of recommender's bias and the overhead caused by the extra communication involved in other reputation-based solutions.

The issue of setting the thresholds involved when deciding whether a node should be black-listed, in order to defend against selectively packet-dropping nodes, is not considered in any of the above mentioned reputation-based approaches. If a node is detected to misbehave a certain number of times, this triggers a negative recommendation. A selfish node may be aware of the thresholds involved in the applied scheme and adjust its behavior accordingly, selectively dropping packets, but staying below the threshold not to affect its

trustworthiness metric. In Paper D we deal with this problem by using an HMM for each node for detection and prediction of node misbehavior.

3.4 Quantification of Anonymity in MANET Routing

Anonymous routing protocols for MANETs, e.g. SDAR [BEKXK04], MASK [ZLL05] and ASR [ZWK+04], have been analyzed in terms of quantification of the efficiency of the routing mechanisms, but typically the privacy offered by the protocol has not been quantified, only analyzed from a qualitative point of view. The notions of *weak* and *strong* location privacy are used in the analysis of ASR. A protocol is said to offer weak location privacy if no one knows the exact location of the source and destination of a message except the source and destination nodes themselves. If the protocol in addition hides the number of hops to source and destination for all the intermediate nodes on the route, it is said to offer strong location privacy. Kong and Hong include the concept of *traceable ratio* in the analysis of the protocol ANODR [KH03], which offers a quantification of privacy in terms of the ratio of a route that can be traced due to the linking of node pseudonyms, in relation to the number of malicious nodes on a route.

The analysis of the position-based routing protocol proposed by Wu and Bertino [WB05], applies the *k-anonymity* metric to MANETs, where k-anonymity [Swe02] means that the anonymity set is always guaranteed to be at least of size k. The protocol is analyzed with regard to the probability of keeping the required k-anonymity under different node densities and radii of the anonymity sets, with respect to communication duration time. This approach is based on the same kind of reasoning about anonymity as introduced in the *Crowds* system [RR98]. This system is designed to provide Internet users with a mechanism for anonymous web browsing by letting users hide amongst a crowd of other users. A problem with this approach when applied to the ad hoc routing setting is that the 'crowd' of nodes might be sparse in some applications, e.g. due to node mobility. The analysis of the protocol in [WB05] also does not take into account the possible *a priori* knowledge of an attacker.

Another approach to quantifying anonymity in MANETs based on *Dempster-Shafer theory* [Sha76] is proposed by Huang [Hua06]. In this approach evidence of communication is collected by measuring the number of packets sent between sets of nodes within a given time period. Probabilities are then assigned to all possible routes for these packets and the anonymity offered is quantified using the Dempster-Shafer theory, which offers methods for reasoning about the uncertainties involved in the evidence of communication. The analysis method in [Hua06] is interesting as it could be used to monitor the anonymity performance of a MANET, if packets are being collected and the measure calculated at regular time intervals in a monitoring framework. However, the approach requires full knowledge of network topology at all times, including all possible routing paths between nodes. In

a dynamic network with mobile nodes, such a centralized monitoring system seems impractical.

4. Research Methodology

The research is focused on the development of new theoretical models for analyzing security, privacy and trust in dynamic networks. The development of the models presented in this thesis was done by the thesis author in cooperation with other members of the Q2S security research group. The research is based on literature studies, group discussions and seminars, analytical modeling, implementations and simulations and evaluation of simulation results. The models have been published and presented at international conferences.

Some of the modeling ideas were implemented and tested by simulations, to confirm that they behaved as expected in various scenarios, but full-scale implementations and testing of all parameter variations for model evaluation have not been done due to the time limitation of this work.

5. Summary of Papers

This section provides a short summary of each of the papers that constitute Part II of this thesis and identifies the main contributions of each paper. A statement of the specific contributions of the thesis author is also given.

5.1 Paper A

Real-time Risk Assessment with Network Sensors and Intrusion Detection Systems.

This paper considers a real-time risk assessment method for information systems and networks based on observations from networks sensors such as intrusion detection systems. The system risk is dynamically evaluated using hidden Markov models, providing a mechanism for handling data from sensors with different trustworthiness in terms of false positives and negatives. The method provides a high level of abstraction for monitoring network security, suitable for risk management and intrusion response applications.

Statement of contribution: This paper was the result of a collaboration between all members of the security group at the Q2S centre, and contributions of the author were mainly in the discussions leading to the development of the model used in the paper.

5.2 Paper B

Real-time Intrusion Prevention and Security Analysis of Networks using HMMs

In this paper we propose to use a hidden Markov model to model sensors for an intrusion prevention system (IPS). Observations from different sensors are

aggregated in the HMM and an intrusion frequency security metric is estimated. We use a Markov model that captures the interaction between the attacker and the network to model and predict the next step of an attacker. A new HMM is created and used for updating the estimated system state for each observation, based on the sensor trustworthiness and the time since last observation processed. Our objective is to calculate and maintain a state probability distribution that can be used for intrusion prediction and prevention. We show how our sensor model can be applied to an IPS architecture based on intrusion detection system sensors, real-time traffic surveillance and online risk assessment. Our approach is illustrated by a small case study.

Statement of contribution: This paper was the result of a joint work between the author, Kjetil Haslum and our supervisor Svein J. Knapskog. The development of the mathematical model used in this paper is a result of discussions between all the authors and can be seen as a further development of previous work on intrusion prevention systems by Kjetil Haslum. The author was responsible for writing most of the paper, except sections 3.2, 3.3 and 3.4.

5.3 Paper C

Quantification of Anonymity for Mobile Ad Hoc Networks

We propose a probabilistic system model for anonymous ad hoc routing protocols that takes into account the a priori knowledge of the adversary and illustrate how the information theoretical entropy can be used for quantification of the anonymity offered by a routing protocol as the adversary captures an increasing number of nodes in the network. The proposed measurement schema is applied to ANODR and ARM routing protocols.

Statement of contribution: This paper was written by the author, without any co-authors.

5.4 Paper D

TSR: Trust-based Secure MANET Routing using HMMs

In this paper we propose a trust-based security extension to the mobile ad hoc network dynamic source routing protocol (DSR), where the state probability of a node, according to its corresponding hidden Markov model (HMM), is being used for deciding the node's trustworthiness. Our approach detects the selective packet dropping behavior of selfish nodes that could not be detected or defended against with previous suggested solutions. Packet-dropping nodes that are acting selfishly by selectively dropping packets and not sharing their bandwidth with the rest of the network, are excluded from the network. This policy is enforcing cooperation among nodes and reduces the incentives for selfish node behavior.

Statement of contribution: This paper was written by the author, with advice and comments by Svein J. Knapskog and Bjarne E. Helvik.

5.5 Paper E

Learning Trust in Dynamic Multiagent Environments using HMMs

In this paper, we propose a trust model for autonomous agents in mulitagent environments based on hidden Markov models and reinforcement learning. By this combination, the reliability of the hidden Markov model will be improved since its parameters are re-estimated after training of the model with the reinforcement learning module.

Statement of contribution: Most parts of this paper, except some parts of the introduction and related work were written by the author. The idea of combining reinforcement learning with the hidden Markov modeling was the result of discussions between the author and Mozhgan Tavakolifard.

5.6 Paper F

Comparison of the Beta and the Hidden Markov Models of Trust in Dynamic Environments

In this paper we present a comparison of our proposed hidden Markov trust model to the Beta reputation system. The hidden Markov trust model takes the time between observations into account. It also distinguishes between system states and uses methods previously applied to intrusion detection for the prediction of which state an agent is in. We show that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents. This means that our trust model may be more realistic in dynamic environments. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Beta reputation system.

Statement of contribution: This paper was written by the author, with advice and comments by Svein J. Knapskog and Bjarne E. Helvik.

6. Summary and Concluding Remarks

We have given an introduction to the work that are presented in the papers in the second part of this thesis. To summarize, the main contributions of this thesis are:

- Mathematical modeling of security in computer networks using hidden Markov models for the modeling of sensor trustworthiness in an intrusion prevention system.
- New security metrics for computer networks derived from the Markov models: computer network risk, the mean time to next intrusion and the intrusion frequency.
- The development of a new method for aggregation of intrusion detection alerts from multiple intrusion detection systems in a computer network.

- A trust model for multiagent systems based on hidden Markov modeling and reinforcement learning.
- The measuring of agent trustworthiness based on the predicted state probability distribution.
- A decentralized reputation system based on hidden Markov modeling suitable for dynamic environments.
- A trust-based security extension to the dynamic source routing protocol, with the purpose of detecting nodes that are dropping packets selectively.
- Modeling of anonymous ad hoc routing and a new method for measuring the amount of anonymity offered by the routing protocol using conditional entropy, including the a priori knowledge of the attacker.

The mathematical models describe the behavior of complex and dynamic systems, where human and non-human entities are interacting. The analytical modeling approaches give an abstract view of the system, where simplifications of the involved complex processes are needed in order to create a tractable model. The statistical modeling approach using Markov models brings along the assumption that the history of the system is contained in its state, i.e. the behavior of the system is only depending on its current state. Also we assume that the state occupancy time can be described by a probability distribution. For the hidden Markov modeling we also have to include the assumption of independence of observations. All these assumptions are simplifications of the system behavior, which means that the models might not be returning fully realistic results valid for a given real-life setting.

To improve the confidence of the modeling approaches, the models could be further evaluated with simulations and tested with real-life data input. In [ÅVVK06], the hidden Markov modeling of risk given in Paper A is evaluated using two data sets based on real network traffic, with promising results. This testing indicates that the risk-level estimated from the hidden Markov model is indeed reflecting the true network risk, as long as the individual sensors, i.e. intrusion detection systems, are giving reasonably reliable output. If the sensors suffer from high probabilities of false positives and false negatives, this will naturally affect the reliability of the calculated risk values. In Paper B we model the trustworthiness of each sensor in the system and propose to use a learning algorithm for estimating the parameters, so that the reliability of each individual sensor is reflected in the model. This solution may improve the confidence in the results of the risk assessment under circumstances where sensors have variable probabilities of false positives and negatives, but this remains to be tested on real-life data sets.

Another issue with intrusion detection and intrusion prevention systems that we did not consider in this work is the security of the implementation of the system itself. The intrusion prevention system is an attractive target

for attackers, it needs to be properly secured from attacks coming from both outside and inside the network. An intrusion prevention system implementing too strict security policies resulting in general low performance could easily become a target of denial of service (DoS) attacks and could even suffer from problems with self denial of service.

The hidden Markov trust model could also be further evaluated by a full-scale implementation and simulations using a testbed for trust and reputation models, e.g. the ART testbed [FKM+05]. Both the hidden Markov trust model and the network security model using HMMs could be made more refined by adding more states and observation symbols. However, this would lead to a large number of parameters that would need to be managed. If people without expert knowledge in security and statistical modeling are to use these models, the amount of parameters should be kept to a minimum. For specific applications of the hidden Markov modeling, the minimum number of states and observation symbols required could be found dynamically by using clustering techniques [KR90] on training data sets. One application of such dynamic clustering for the purpose of detecting credit card fraud is given in [SKS08]. However, such data sets may be hard to obtain in many cases since organizations usually are reluctant to release this type of potentially sensitive statistical material.

The observations used in the hidden Markov models which we describe in Papers A, B and D are derived from monitoring of the network. The monitoring activity might conflict with the privacy of the users associated with the network nodes, e.g. in the ad hoc network routing protocol we assume that nodes operate in promiscuous mode. Adequate privacy-enhancing techniques are required to ensure the privacy of users in such applications, e.g by using pseudonymization of node identities. Anonymous routing protocols aim at hiding the location and identity of network nodes. The application of trust-based mechanisms like the one proposed in Paper D, seems difficult in combination with anonymous routing as described in Paper C. In [BVTM07] one possible solution for the detection of flooding attacks under anonymous communication is proposed, but the problem of detecting anonymous packet-dropping nodes in a general ad hoc network scenario remains to be solved.

II

INCLUDED PAPERS

Paper A

Real-time Risk Assessment with Network Sensors and Intrusion Detection Systems

André Årnes, Karin Sallhammar, Kjetil Haslum, Tønnes Brekne, Marie E. G. Moe and Svein J. Knapskog.

In Proceedings of the 2005 International Conference on Computational Intelligence and Security (CIS'05)

Springer. Xian, China. December 15-19, 2005

REAL-TIME RISK ASSESSMENT WITH NETWORK SENSORS AND INTRUSION DETECTION SYSTEMS

André Årnes, Karin Sallhammar, Kjetil Haslum, Tønnes Brekne,
Marie Elisabeth Gaup Moe, and Svein Johan Knapskog
Centre for Quantifiable Quality of Service in Communication Systems
Norwegian University of Science and Technology
O.S. Bragstads plass 2E, N-7491 Trondheim, Norway
{andrearn,sallhamm,haslum,tonnes,marieeli,knapskog}@q2s.ntnu.no

Abstract This paper considers a real-time risk assessment method for information systems and networks based on observations from networks sensors such as intrusion detection systems. The system risk is dynamically evaluated using hidden Markov models, providing a mechanism for handling data from sensors with different trustworthiness in terms of false positives and negatives. The method provides a higher level of abstraction for monitoring network security, suitable for risk management and intrusion response applications.

1. Introduction

Risk assessment is a central issue in management of large-scale networks. However, current risk assessment methodologies focus on manual risk analysis of networks during system design or through periodic reviews. Techniques for real-time risk assessment are scarce, and network monitoring systems and intrusion detection systems (IDS) are the typical approaches. In this paper, we present a real-time risk assessment method for large scale networks that build upon existing network monitoring and intrusion detection systems. An additional level of abstraction is added to the network monitoring process, focusing on risk rather than individual warnings and alerts. The method enables the assessment of risk both on a system-wide level, as well as for individual objects.

The main benefit of our approach is the ability to aggregate data from different sensors with different weighting according to the trustworthiness of the sensors. This focus on an aggregate risk level is deemed more suitable for network management and automated response than individual intrusion detection alerts. By using hidden Markov models (HMM), we can find the most likely state probability distribution of monitored objects, considering the

trustworthiness of the IDS. We do not make any assumptions on the types of sensors used in our monitoring architecture, other than that they are capable of providing standardized output as required by the model parameters presented in this paper.

1.1 Target Network Architecture

The target of the risk assessment described in this paper is a generic network consisting of computers, network components, services, users, etc. The network can be arbitrarily complex, with wireless ad-hoc devices as well as ubiquitous services. The network consists of entities that are either *subjects* or *objects*. Subjects are capable of performing actions on the objects. A subject can be either users or programs, whereas objects are the targets of the risk assessment. An asset may be considered an object. The unknown factors in such a network may represent vulnerabilities that can be exploited by a malicious attacker or computer program and result in unwanted incidents. The potential exploitation of a vulnerability is described as threats to assets. The *risk* of a system can be identified through the evaluation of the probability and consequence of unwanted incidents.

1.2 Monitoring and Assessment Architecture

We assume a multiagent system architecture consisting of agents that observe objects in a network using sensors. The architecture of a multiagent risk assessment system per se is not the focus of this paper, but a description is included as a context.

An *agent* is a computer program capable of a certain degree of autonomous actions. In a multiagent system, agents are capable of communicating and cooperating with other agents. In this paper, an agent is responsible for collecting and aggregating sensor data from a set of sensors that monitor a set of objects. The main task of the agent is to perform real-time risk assessment based on these data. A multiagent architecture has been chosen for its flexibility and scalability, and in order to support distributed automated response.

A *sensor* can be any information-gathering program or device, including network sniffers (using sampling or filtering), different types of intrusion detection systems (IDS), logging systems, virus detectors, honeypots, etc. The main task of the sensors is to gather information regarding the security state of objects. The assumed monitoring architecture is hybrid in the sense that it supports any type of sensor. However, it is assumed that the sensors are able to classify and send standardized observations according to the risk assessment model described in this paper.

1.3 Related Work

Risk assessment has traditionally been a manual analysis process based on a standardized framework, such as [Sta04]. A notable example of real-time risk assessment is presented in [GK04], which introduces a formal model for the real time characterization of risk faced by a host. *Distributed intrusion detection systems* have been demonstrated in several prototypes and research papers, such as [SCCC+96, SBD+91]. Multiagent systems for intrusion detection, as proposed in [BGFI+98] and demonstrated in e.g. [HWH+03] (an IDS prototype based on lightweight mobile agents) are of particular relevance for this paper. An important development in distributed intrusion detection is the recent IDMEF (Intrusion Detection Message Exchange Format) IETF Internet draft [DCF05]. *Hidden Markov models* have recently been used in IDS architectures to detect multi-stage attacks [OMSH03], and as a tool to detect misuse based on operating system calls [WFP99]. *Intrusion tolerance* is a recent research field in information security related to the field of fault tolerance in networks. The research project SITAR [GPWW+01] presents a generic state transition model, similar to the model used in this paper, to describe the dynamics of intrusion tolerant systems. Probabilistic validation of intrusion tolerant systems is presented in [SCS03].

2. Risk Assessment Model

In order to be able to perform dynamic risk assessment of a system, we formalize the distributed network sensor architecture described in the previous section. Let $O = \{o_1, o_2, \ldots\}$ be the set of objects that are monitored by an agent. This set of objects represents the part of the network that the agent is responsible for. To describe the security state of each object, we use discrete-time Markov chains. Assume that each object consisting of N states, denoted $S = \{s_1, s_2, \ldots, s_N\}$.

As the security state of an object changes over time, it will move between the states in S. The sequence of states that an object visits is denoted $X = x_1, x_2, \ldots, x_T$, where $x_t \in S$ is the state visited at time t. For the purpose of this paper, we assume that the state space can be represented by a general model consisting of three states: Good (G), Attacked (A) and Compromised (C), i.e. $S = \{G, A, C\}$. State G means that the object is up and running securely and that it is not subject to any kind of attack actions. In contrast to [GPWW+01], we assume that objects always are vulnerable to attacks, even in state G. As an attack against an object is initiated, it will move to security state A. An object in state A is subject to an ongoing attack, possibly affecting its behavior with regard to security. Finally, an object enters state C if it has been successfully compromised by an attacker. An object in state C is assumed to be completely at the mercy of an attacker and subject to any kind of confidentiality, integrity and/or availability breaches.

The security observations are provided by the sensors that monitor the objects. These *observation messages* are processed by agents, and it is assumed that the messages are received or collected at *discrete time intervals*. An observation message can consist of any of the symbols $V = \{v_1, v_2, \ldots, v_M\}$. These symbols may be used to represent different types of alarms, suspect traffic patterns, entries in log data files, input from network administrators, and so on. The *sequence* of observed messages that an agent receives is denoted $Y = y_1, y_2, \ldots, y_T$, where $y_t \in V$ is the observation message received at time t. Based on the sequence of observation messages, the agent performs dynamic risk assessment. The agent will often receive observation messages from more than one sensor, and these sensors may provide different types of data, or even inconsistent data. All sensors will not be able to register all kinds of attacks, so we cannot assume that an agent is able to resolve the correct state of the monitored objects at all times. The observation symbols are therefore probabilistic functions of the object's Markov chain, the object's true security state will be *hidden* from the agent. This is consistent with the basic idea of HMM [Rab90].

2.1 Modeling Objects as Hidden Markov Models

Each monitored object can be represented by a HMM, defined by $\lambda = \{\mathbf{P}, \mathbf{Q}, \pi\}$.

$\mathbf{P} = \{p_{ij}\}$ is the state transition probability distribution matrix for object o, where $p_{ij} = P(x_{t+1} = s_j | x_t = s_i), 1 \leq i, j \leq N$. Hence, p_{ij} represents the probability that object o will transfer into state s_j next, given that its current state is s_i. To be able to estimate $\mathbf{P}$ for real-life objects, one may use either statistical attack data from production or experimental systems or the subjective opinion of experts. Learning algorithms may be employed in order to provide a better estimate of $\mathbf{P}$ over time.

$\mathbf{Q} = \{q_j(l)\}$ is the observation symbol probability distribution matrix for object o in state s_j, whose elements are $q_j(l) = P(y_t = v_l | x_t = s_j), 1 \leq j \leq N, 1 \leq l \leq M$. In our model, the element $q_j(l)$ in $\mathbf{Q}$ represents the probability that a sensor will send the observation symbol v_l at time t, given that the object is in state s_j at time t. $\mathbf{Q}$ therefore indicates the sensor's false-positive and false-negative effect on the agents risk assessments.

$\pi = \{\pi_i\}$ is the initial state distribution for the object. Hence, $\pi_i = P(x_1 = s_i)$ is the probability that s_i was the initial state of the object.

2.2 Quantitative Risk Assessment

Following the terminology in [Sta04], risk is measured in terms of *consequences* and *likelihood*. A consequence is the (qualitative or quantitative) outcome of an event and the likelihood is a description of the probability of the event. To perform dynamic risk assessment, we need a mapping: $\mathcal{C} : S \rightarrow \mathbb{R}$, describing the expected cost (due to loss of confidentiality, integrity and avail-

ability) for each object. The total risk $\mathcal{R}_t$ for an object at time t is

$$\mathcal{R}_t = \sum_{i=1}^{N} \mathcal{R}_t(i) = \sum_{i=1}^{N} \gamma_t(i)\mathcal{C}(i) \tag{1}$$

where $\gamma_t(i)$ is the probability that the object is in security state s_i at time t, and $\mathcal{C}(i)$ is the cost value associated with state s_i.

In order to perform real-time risk assessment for an object, an agent has to dynamically update the object's state probability $\gamma_t = \{\gamma_t(i)\}$. Given an observation y_t, and the HMM λ, the agent can update the state probability γ_t of an object using Algorithm 1. The complexity of the algorithm is $O(N^2)$. For further details, see the Appendix.

Algorithm 1 Update state probability distribution

Require: y_t, λ {the observation at time t, the hidden Markov model}
Ensure: γ_t {the security state probability at time t}
  if $t = 1$ **then**
    for $i = 1$ to N **do**
      $\alpha_1(i) \leftarrow q_i(y_1)\pi_i$
      $\gamma_1(i) \leftarrow \frac{q_i(y_1)\pi_i}{\sum_{j=1}^{N} q_j(y_1)\pi_j}$
    end for
  else
    for $i = 1$ to N **do**
      $\alpha_t(i) \leftarrow q_i(y_t) \sum_{j=1}^{N} \alpha_{t-1}(j)p_{ji}$
      $\gamma_t(i) \leftarrow \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)}$
    end for
  end if
  return γ_t

3. Case – Real-time Risk Assessment for a Home Office

To illustrate the theory, we perform real-time risk assessment of a typical home office network, consisting of an Internet router/WLAN access point, a stationary computer with disk and printer sharing, a laptop using WLAN, and a cell phone connected to the laptop using Bluetooth. Each of the objects (hosts) in the home office network has a sensor that processes log files and checks system integrity (a host IDS). In addition, the access point has a network monitoring sensor that is capable of monitoring traffic between the outside network and the internal hosts (a network IDS).

For all objects, we use the state set $S = \{G, A, C\}$. The sensors provide observations in a standardized message format, such as IDMEF, and they are

capable of classifying observations as indications of the object state. Each sensor is equipped with a database of signatures of potential attacks. For the purpose of this example, each signature is associated with a particular state in S. We define the observation symbols set as $V = \{g, a, c\}$, where the symbol g is an indication of state G and so forth. Note that we have to preserve the discrete-time property of the HMM by sampling sensor data periodically. If there are multiple observations during a period, we sample one at random. If there are no observations, we assume the observation symbol to be g. In order to use multiple sensors for a single object, a round-robin sampling is used to process only one observation for each period. This is demonstrated in example 3.

The home network is monitored by an agent that regularly receives observation symbols from the sensors. For each new symbol, the agent uses Algorithm 1 to update the objects' security state probability, and (1) to compute its corresponding risk value. Estimating the matrices $\mathbf{P}$ and $\mathbf{Q}$, as well as the cost $\mathcal{C}$ associated with the different states, for the objects in this network is a non-trivial task that is out of scope for this paper.

The parameter values in these examples are therefore chosen for illustration purposes only. Also, we only demonstrate how to perform dynamic risk assessment of the laptop.

3.1 Example 1: Laptop Risk Assessment by HIDS Observations

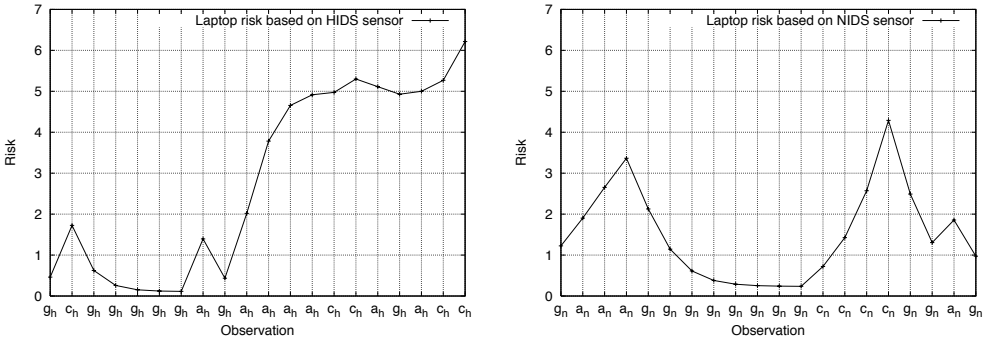
First, we assess the risk of the laptop, based on an observation sequence Y_{HIDS-L}, containing 20 samples collected from the laptop HIDS. We use the HMM $\lambda_L = \{\mathbf{P}_L, \mathbf{Q}_{HIDS-L}, \pi_L\}$, where

$$\mathbf{P}_L = \begin{pmatrix} p_{GG} & p_{GA} & p_{GC} \\ p_{AG} & p_{AA} & p_{AC} \\ p_{CG} & p_{CA} & p_{CC} \end{pmatrix} = \begin{pmatrix} 0.995 & 0.004 & 0.001 \\ 0.060 & 0.900 & 0.040 \\ 0.008 & 0.002 & 0.990 \end{pmatrix}, \tag{2}$$

$$\mathbf{Q}_{HIDS-L} = \begin{pmatrix} q_G(g) & q_G(a) & q_G(c) \\ q_A(g) & q_A(a) & q_A(c) \\ q_C(g) & q_C(a) & q_C(c) \end{pmatrix} = \begin{pmatrix} 0.70 & 0.15 & 0.15 \\ 0.15 & 0.70 & 0.15 \\ 0.20 & 0.20 & 0.60 \end{pmatrix}, \tag{3}$$

$$\pi_L = (\pi_G, \pi_A, \pi_C) = (0.8, 0.1, 0.1). \tag{4}$$

Since the HIDS is assumed to have low false-positive and false-negative rates, both $q_G(a), q_G(c), q_A(c) \ll 1$ and $q_A(g), q_C(g), q_C(a) \ll 1$ in $\mathbf{Q}_{HIDS-L}$. The dynamic risk in Figure 1(a) is computed based on the observation sequence Y (as shown on the x-axis of the figure) and a security state cost estimate measured as $\mathcal{C}_L = (0, 5, 10)$.



(a) HIDS sensor (b) NIDS sensor

Figure 1. Laptop risk assessment

3.2 Example 2: Laptop Risk Assessment by NIDS Observations

Now, we let the risk assessment process of the laptop be based on another observation sequence, Y_{NIDS-L}, collected from the NIDS. A new observation symbol probability distribution is created for the NIDS

$$\mathbf{Q}_{NIDS-L} = \begin{pmatrix} 0.5 & 0.3 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{pmatrix}. \tag{5}$$

One can see that the NIDS has higher false-positive and false-negative rates, compared to the HIDS. Figure 1(b) shows the laptop risk when using the HMM $\lambda_L = \{\mathbf{P}_L, \mathbf{Q}_{NIDS-L}, \pi_L\}$. Note that the observation sequence is not identical to the one in example 1, as the two sensors are not necessarily consistent.

3.3 Example 3: Aggregating HIDS and NIDS Observations

The agent now aggregates the observations from the HIDS and NIDS sensors by sampling from the observation sequences Y_{HIDS-L} and Y_{NIDS-L} in a round-robin fashion. To update the current state probability γ_t, the agent therefore chooses the observation symbol probability distribution corresponding to the sampled sensor, i.e the HMM will be

$$\lambda_L = \{\mathbf{P}_L, \mathbf{Q}^*, \pi_L\}, \text{where } \mathbf{Q}^* = \begin{cases} \mathbf{Q}_{HIDS-L} \text{ if } y_t \in Y_{HIDS} \\ \mathbf{Q}_{NIDS-L} \text{ if } y_t \in Y_{NIDS} \end{cases}. \tag{6}$$

The calculated risk is illustrated in Figure 2. The graph shows that some properties of the individual observation sequences are retained.

4. Managing Risk with Automated Response

In order to achieve effective incident response, it must be possible to effectively initiate defensive measures, for example by reconfiguring the security services and mechanisms in order to mitigate risk. Such measures may be manual or automatic. An information system or network can be automatically reconfigured in order to reduce an identified risk, or the system can act as a support system for system and network administrators by providing relevant information and recommending specific actions. To facilitate such an approach, it is necessary to provide a mechanism that relates a detected security incidence to an appropriate response, based on the underlying risk model. Such a mechanism should include a policy for what reactions should be taken in the case of a particular incident, as well as information on who has the authority to initiate or authorize the response. Examples of distributed intrusion detection and response systems have been published in [CHSP00, PN97].

The dynamic risk-assessment method described in this paper can provide a basis for automated response. If the risk reaches a certain level, an agent may initiate an automated response in order to control the risk level. Such a response may be performed both for individual objects (e.g. a compromised host) or on a network-wide level (if the network risk level is to high). Examples of a local response may be firewall reconfigurations for a host, changing logging granularity, or shutting down a system. Examples of a global response may be the revocation of a user certificate, the reconfiguration of central access control configurations, or firewall reconfigurations. Other examples include traffic rerouting or manipulation, and honeypot technologies. Note that such adaptive measures has to be supervised by human intelligence, as they necessarily introduce a risk in their own right. A firewall reconfiguration mechanism can, for example, be exploited as part of a denial-of-service attack.

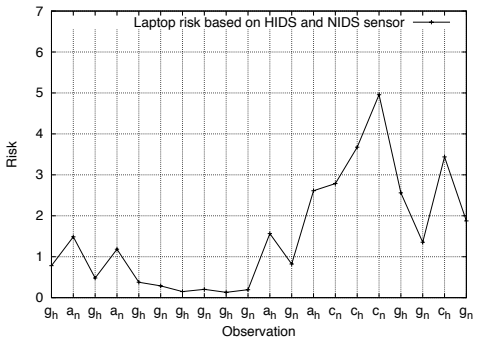


Figure 2. Laptop risk assessment based on two sensors (HIDS and NIDS)

5. Conclusion

We present a real-time risk-assessment method using HMM. The method provides a mechanism for aggregating data from multiple sensors, with different weightings according to sensor trustworthiness. The proposed discrete-time model relies on periodic messages from sensors, which implies the use of sampling of alert data. For the purpose of real-life applications, we propose further development using continuous-time models in order to be able to handle highly variable alert rates from multiple sensors. We also give an indication as to how this work can be extended into a multiagent system with automated response, where agents are responsible for assessing and responding to the risk for a number of objects.

Appendix: On Algorithm 1

Given the first observation y_1 and the hidden Markov model λ, the initial state distribution $\gamma_1(i)$ can be calculated as

$$\gamma_1(i) = P(x_1 = s_i|y_1, \lambda) = \frac{P(y_1, x_1 = s_i|\lambda)}{P(y_1|\lambda)} = \frac{P(y_1|x_1 = s_i, \lambda)P(x_1 = s_i|\lambda)}{P(y_1|\lambda)}. \tag{A.1}$$

To find the denominator, one can condition on the first visited state and sum over all possible states

$$P(y_1|\lambda) = \sum_{j=1}^{N} P(y_1|x_1 = s_j, \lambda)P(x_1 = s_j|\lambda) = \sum_{j=1}^{N} q_j(y_1)\pi_j. \tag{A.2}$$

Hence, by combining (A.1) and (A.2)

$$\gamma_1(i) \quad = \quad \frac{q_i(y_1)\pi_i}{\sum_{j=1}^{N} q_j(y_1)\pi_j}, \tag{A.3}$$

where $q_j(y_1)$ is the probability of observing symbol y_1 in state s_j, and π is the initial state probability. To simplify the calculation of the state distribution after t observations we use the *forward-variable* $\alpha_t(i) = P(y_1 y_2 \cdots y_t, x_t = s_i|\lambda)$, as defined in [Rab90]. By using recursion, this variable can be calculated in an efficient way as

$$\alpha_t(i) = q_i(y_t) \sum_{j=1}^{N} \alpha_{t-1}(j)p_{ji}, \qquad t > 1. \tag{A.4}$$

From (A.1) and (A.3) we find the initial forward variable

$$\alpha_1(i) = q_i(y_1)\pi_i, \qquad t = 1. \tag{A.5}$$

In the derivation of $\alpha_t(i)$ we assumed that y_t only depend on x_t and that the Markov property holds.

Now we can use the forward variable $\alpha_t(i)$ to update the state probability distribution by new observations. This is done by

$$\begin{aligned}\gamma_t(i) = P(x_t = s_i | y_1 y_2 \cdots y_t, \lambda) &= \frac{P(y_1 y_2 \cdots y_t, x_t = s_i|\lambda)}{P(y_1 y_2 \cdots y_t|\lambda)} \\ &= \frac{P(y_1 y_2 \cdots y_t, x_t = s_i|\lambda)}{\sum_{j=1}^{N} P(y_1 y_2 \cdots y_t, x_t = s_j|\lambda)} = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)}.\end{aligned} \tag{A.6}$$

Note that (A.6) is similar to Eq. 27 in [Rab90], with the exception that we do not account for observations that occur after t, as our main interest is to calculate the object's state distribution after a number of observations.

References

[BGFI+98] J. S. Balasubramaniyan, J. O. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni. An architecture for intrusion detection using autonomous agents. In *Proceedings of the 14th Annual Computer Security Applications Conference*, page 13. IEEE Computer Society, 1998.

[CHSP00] Curtis A. Carver Jr., John M.D. Hill, John R. Surdu, and Udo W. Pooch. A methodology for using intelligent agents to provide automated intrusion response. In *Proceedings of the IEEE Workshop on Information Assurance and Security*, 2000.

[DCF05] H. Debar, D. Curry, and B. Feinstein. Intrusion detection message exchange format (IDMEF) – Internet-Draft, 2005.

[GK04] Ashish Gehani and Gershon Kedem. Rheostat: Real-time risk management. In *Recent Advances in Intrusion Detection: 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004. Proceedings*, pages 296–314. Springer, 2004.

[GPWW+01] Katerina Goseva-Popstojanova, Feiyi Wang, Rong Wang, Fengmin Gong, K. Vaidyanathan, K. Trivedi, and B. Muthusamy. Characterizing intrusion tolerant systems using a state transition model. In *Proceedings of DARPA Information Survivability Conference and Exposition II, DISCEX '01.*, volume 2, pages 211–221, 2001.

[HWH+03] Guy Helmer, Johnny S. K. Wong, Vasant G. Honavar, Les Miller, and Yanxin Wang. Lightweight agents for intrusion detection. *J. Syst. Softw.*, 67(2):109–122, 2003.

[OMSH03] Dirk Ourston, Sara Matzner, William Stump, and Bryan Hopkins. Applications of hidden markov models to detecting multi-stage network attacks. In *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS)*, 2003.

[PN97] P. A. Porras and P. G. Neumann. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In *Proc. 20th NIST-NCSC National Information Systems Security Conference*, pages 353–365, 1997.

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.

[SBD+91] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur. DIDS (distributed intrusion detection system) - motivation, architecture, and an early prototype. In *Proceedings of the 14th National Computer Security Conference*, pages 167–176, Washington, DC, 1991.

[SCCC+96] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. GrIDS – A graph-based intrusion detection system for large networks. In *Proceedings of the 19th National Information Systems Security Conference*, 1996.

[SCS03] S. Singh, M. Cukier, and W.H. Sanders. Probabilistic validation of an intrusion-tolerant replication system. In de Bakker, J.W., de Roever, W.-P., and Rozenberg, G., editors, *International Conference on Dependable Systems and Networks (DSN'03)*, June 2003.

[Sta04] Standards Australia and Standards New Zealand. AS/NZS 4360: 2004 risk management, 2004.

[WFP99] C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: Alternative data models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, 1999.

Paper B

Real-time Intrusion Prevention and Security Analysis of Networks using HMMs

Kjetil Haslum, Marie E. G. Moe, and Svein Knapskog

In Proceedings of the Fourth IEEE LCN Workshop on Network Security (WNS 2008)

IEEE. Montreal, Canada. October 17, 2008

Is not included due to copyright

Paper C

Quantification of Anonymity for Mobile Ad Hoc Networks

Marie E. G. Moe

In Proceedings of the 4th International Workshop on Security and Trust Management (STM 08)

Elsevier. Trondheim, Norway. June 16-17, 2008

QUANTIFICATION OF ANONYMITY FOR MOBILE AD HOC NETWORKS

Marie Elisabeth Gaup Moe
Centre for Quantifiable Quality of Service in Communication Systems
Norwegian University of Science and Technology
O.S. Bragstads plass 2E, N-7491 Trondheim, Norway
marieeli@q2s.ntnu.no

Abstract We propose a probabilistic system model for anonymous ad hoc routing protocols that takes into account the a priori knowledge of the adversary, and illustrate how the information theoretical entropy can be used for quantification of the anonymity offered by a routing protocol as the adversary captures an increasing number of nodes in the network. The proposed measurement schema is applied to ANODR and ARM routing protocols.

1. Introduction

There is a need to provide secure cryptographic services in dynamic network environments with untrusted parties and a changing net topology. Cryptographic services are security services implemented by cryptographic mechanisms, examples of such services are confidentiality, authenticity, integrity, privacy, accountability, accessibility and nonrepudiation. Privacy is a service that is often difficult to realize at the same time as other cryptographic services, like authenticity, accountability and nonrepudiation. Parties in dynamic networking environments like mobile ad hoc networks, where each node is acting as a combined terminal and router, would be particularly exposed to threats against their privacy since they have no control over the trustworthiness of network nodes that handle the messages sent. Appropriate privacy enhancing cryptographic mechanisms, that can be trusted to work as intended, are required to handle this problem. Privacy has become an increasing concern for users of communication services. As communication networks are becoming more complex and diverse, the trustworthiness of network nodes, like routers, cannot always be guaranteed.

In order to properly define secure anonymous routing it is useful to have a security model that represents the system, and to have some sort of measure that can quantify the amount of anonymity offered by the protocol.

In this paper we propose a probabilistic system model for anonymous ad hoc routing protocols that takes into account the a priori knowledge of the

adversary and illustrate how the information theoretical entropy measure can be used for quantification of the anonymity of the system as the adversary captures an increasing number of nodes in the network.

2. Background: Anonymity Metrics

In this section we will give a short survey of the state of art on quantification of anonymity.

2.1 Defining Anonymity

Before we can start with measuring anonymity we need to have a clear understanding of what anonymity means. We adopt the definitions by Pfitzmann and Hansen [PH05]:

- *Anonymity* is the state of being not identifiable within a set of subjects, the *anonymity set*.
- *Unlinkability* of two or more items within a defined system means that these items are no more and no less related than they are related concerning the a-priori knowledge.

For the ad hoc routing setting an anonymous routing protocol should ideally offer sender and recipient anonymity, meaning that the sender of a message or recipient of a message remains unidentifiable under the assumed adversary model. We also want to achieve *realationship anonymity* between the sender and the recipient of a message, so that an observer cannot determine which nodes are taking part in a specific communication flow. In other words, sender and recipient are unlinkable. Note that the unlinkability property is weaker than the anonymity property, as anonymity of both sender and recipient implies the unlinkabilty between them. For the remaining of this paper we will focus on anonymity in the context of ad hoc routing, and in particular look at sender and recipient anonymity in the route discovery part of ad hoc routing protocols.

An *identity* is defined in [PH05] as any subset of attributes of an individual which identifies this individual within any set of individuals. So usually there is no such thing as the identity, but several of them. The nodes in an ad hoc network could be identified in terms of a node *identifier*. A node identifier could for instance be the node's mac address or ip address, or the identity of the user controlling the node at the time. We will assume that every node in a network of N nodes has a unique authenticated node identifier N_i, $1 \leq i \leq N$. A node could also be identified by its *location*, signal positioning could easily be used to determine an approximate location of a transmitting node. To obtain *location privacy* of a sender node a packet should not reveal the number of hops it has travelled. The packet should also not reveal how many hops it has left to traverse before arriving at the destination in order to obtain recipient location privacy.

2.2 Measuring Anonymity

The classic way of quantifying the degree of anonymity is done by simply measuring the size of the anonymity set [Cha81]. The size of the anonymity set is intuitively an indication of the degree of anonymity, as the more members of the set of potential senders/receivers, the less is the probability that a randomly chosen member of the set was the actual sender/receiver. But we should take into account that anonymity is stronger the more evenly distributed the sending and receiving of messages by the subjects within that set is.

Reiter and Rubin [RR98] give a qualitative scale for degrees of anonymity ranging from *absolute privacy* to *provable exposed*:

- *absolute privacy* means that sending a message is unobservable for the attacker
- *beyond suspicion* means that even though the attacker can see evidence of a sent message, the sender appears no more likely to be the originator than any other potential sender in the system
- *probable innocence* means that to the attacker, the sender appears no more likely to be the originator than to not be the originator
- *exposed* means that the attacker can identify the sender of a message
- *provable exposed* means that the attacker can also prove the identity of the sender to others

The degree of anonymity could also be quantified in terms of the information theoretical *entropy* of the probability distribution that the attacker assigns to each possible sender as being the originator of a message, after observing the system. In a system with N users, let p_i be the probability assigned by the attacker for user i to be the sender/recipient of a message, and let X be the discrete random variable taking the possible values $x_1, x_2, \ldots, x_N$ with probabilities $p_1, p_2, \ldots, p_N$ respectively, the entropy $H(X)$ of the probability distribution can be calculated by

$$H(X) = -\sum_{i=1}^{N} p_i \log_2(p_i). \tag{1}$$

The entropy can be interpreted as the number of bits of additional information that the attacker needs in order to definitely identify a user, or as the effective decrease in uncertainty. This information-theoretic measure of anonymity was proposed independently by Diaz et al [DSCP02] and Serjantov and Danezis [SD02]. For quantification of the degree of anonymity Diaz et al compared the information obtained by the attacker after observing the system against the optimal situation where all users are equally likely to have sent/received the

message. The degree of anonymity is denoted d and defined as

$$d = 1 - \frac{H_{\max} - H(X)}{H_{\max}} = \frac{H(X)}{H_{\max}},$$

where $H_{\max} = \log_2(N)$ is the maximum entropy for the system. This measure tells us how evenly distributed the probabilities within the anonymity set are.

Entropy may be used as a measure of how evenly the probabilities are distributed within each distribution, but two distributions with the same entropy could still have very different qualitative anonymity. In particular the *beyond suspicion* property could be broken even with high entropy, since a distribution of high entropy does not necessarily guarantee that a particular sender or recipient does not have a much higher probability to have sent or received a message than the rest of the potential senders/receivers. Some examples of such probability distributions are given by Tóth et al [THV04]. To capture this they suggest to use the worst case metric *minimum entropy* $H_{\min}$, which denotes the probability of the most likely sender/receiver within the anonymity set

$$H_{\min} = -\log_2(\max_{1 \leq i \leq N}(p_i)).$$

This measure was also used by Shmatikov and Wang [SW06] to calculate the relationship anonymity between sender and recipient in several simulations of mix networks, where they take into account the route selection mechanisms and the distribution of message destinations.

Another problem of the entropy measure is that it does not take into account the a priori knowledge of the adversary. In an ad hoc routing setting, if we consider an adversary that has both a global and local perspective on a network, we could imagine that the adversary has some a priori knowledge of the communication patterns of network nodes, derived from traffic analysis or from application-layer contexts. The global adversary could for instance know about the frequency of route request transmissions from all nodes, which give rise to a probability distribution over the potential senders of a particular message. This a priori knowledge could then be combined with the information the adversary obtains by local observations, as suggested by Clauß and Schiffner [CS06]. As noted by Diaz et al [DTD07], the problem of how to combine the entropy measures from two different sources has not yet been fully addressed. It is not necessarily true that the entropy decreases when an adversary gets access to more information in a given attack scenario. However, if we take the weighted average of all possible entropies that the adversary can obtain after observing the system, given the a priori knowledge, this entropy, defined by Shannon as the *conditional entropy*, will always be equal or less to the entropy of the a priori probability distribution.

3. Model Description

The security model for anonymous ad hoc routing introduced in this paper is a probabilistic information theoretical model based on the models for anonymity in mix-networks proposed by Diaz et al [DSCP02] and Serjantov and Danezis [SD02]. The novelty of our approach is that we apply the conditional entropy measure of anonymity to ad hoc networks, that we take into account the a priori knowledge of the adversary and that we quantify the amount of additional information the adversary will gain by taking over more nodes in the network.

3.1 Adversary Model

The ad hoc network consists of a collection of nodes that can come and go into the network, the nodes simultaneously act as senders, recipients and routers. An adversary model usually distinguishes between external/internal, passive/active and global/local adversaries. An external adversary can only capture the communication between nodes while the internal adversary has access to all internal information of compromised nodes. A passive adversary can only eavesdrop on the communication or read the internal information of nodes, while an active adversary may insert, delete or modify messages or alter internal information in nodes. A global adversary has full information of the network while a local adversary only controls part of the network. The most common adversary model used when analyzing the anonymity offered by ad hoc routing protocols is an external passive global adversary (an eavesdropper on the wireless communication of all nodes in the network), that possibly cooperates with one or more internal passive or active local adversaries (malicious nodes inside the network). The proposals for anonymous routing protocols by Zhang et al [ZLL05], Boukerche et al [BEKXK04], Kong and Hong [KH03] and Seys and Preneel [SP06] all use variants of this adversary model.

Hu and Perrig [HP04] propose to characterize an adversary based on the number of nodes it owns in the network and the number of nodes it has compromised, they suggest to use the notation Active-n-m for an active adversary that has compromised n nodes and owns m nodes. We do not wish to separate between owned and compromised nodes, as we assume that a compromised node is fully controlled by the adversary. We are interested in knowing how many nodes in the network that can be overtaken by an adversary before the anonymity offered by the routing protocol gets unacceptably low. In order to achieve this we need to have a quantification of the anonymity offered by the protocol in relation to the number of compromised nodes as well as the total number of nodes in the network. We propose to use the term Passive-c/n for an adversary that is an external passive local or global adversary for the whole network, which is consisting of N nodes, of which this adversary can eavesdrop on the communication of a subset of n nodes, and that has compromised or owns c nodes inside the network, in other words the local or global external

adversary cooperates with a local internal passive adversary that controls c nodes. As we are focusing on the anonymity aspects of the routing protocol we do not in our model take into account an *active* adversary that could inject, drop or modify packets in order to disturb the routing mechanisms or to launch a denial of service attack.

We assume that the adversary carries out a probabilistic attack, this means that the adversary obtains a probability distribution over the potential sender or recipient nodes in the network that could have sent or is the recipient of a particular message. Depending on the number of nodes controlled by the adversary, this probability distribution will vary. The worst case scenario is a Passive-N/N adversary, which is a rather uninteresting case since the adversary controls all nodes in the network. For the case study used in this paper we assume that the external adversary has a global view of the network. This means that the weakest adversary in our model will be a Passive-$0/N$ adversary, which is a global external adversary without any knowledge of any internal node's information.

3.2 Network Topology Model

We choose to use an analytical probabilistic model of the ad hoc network topology, because we want the measurement model to be as general as possible to be able to compare different protocols not only for specific network topologies and specific attack scenarios. An alternative to our analytical approach could be to use simulations, where the anonymity measure is calculated over many different simulated network topologies and routes. When concerning the mobility of nodes this would indeed be a better solution and will be investigated in our further work. It should be noted that the proposed measurement model is resistant on the net topology, so our approach could still be applied to other network topology models.

The analytical network topology model requires some simplifying assumptions. Inspired by the topological model used in [Sey06] we assume that at any given time the network nodes are evenly distributed on a two-dimensional plane and that all nodes have an equal transmission range and communicate through a wireless symmetric channel. We also assume that routes follow shortest distance paths, so that a message transmitted from node N_1 to node N_2 could not have originated from a node closer to N_2 than to N_1. We refer to the *node density* ρ as the number of nodes that lie within each node's transmission range. Let c_1 be the number of nodes that are one hop away from any particular network node, c_2 denotes the number of nodes two hops away and so on. We define $c_0 = 1$, as the only node zero hops away from any node is the node itself. As the hop-count increases from $k-1$ hops to k hops, the number of nodes grows proportionally according to the number of nodes contained within the area difference of two concentric circles with radii k and $k-1$. The number of nodes k hops away from a particular node will be $c_k = (1/2)\rho(2k-1)$.

3.3 Measurement Model

When evaluating the anonymity offered by a routing protocol, we are interested in knowing how resistant the protocol is against possibly colluding malicious nodes. To achieve this we measure the anonymity in terms of entropy based on the external global view of the adversary before any nodes have been compromised, and then quantify the average gain in information of the adversary as it controls an increasing number of nodes in the network, using the conditional entropy measure and following some of the discussion about this measure by Diaz et al [DTD07]. In the following we will only discuss sender anonymity, with minor adjustments the same reasoning can also be applied to recipient anonymity.

Let X be a discrete random variable with probability mass function $p_i = P(X = x_i)$, x_i corresponds to a node N_i in the network and p_i is the probability that N_i will be sending a message m, as viewed by the adversary before any internal nodes have been captured. Let $\mathcal{P}_0$ be the discrete a priori probability distribution with values p_i for $1 \leq i \leq N$ in a network with N nodes. This a priori probability distribution could for instance be based on traffic analysis performed by the global external adversary. The anonymity of the nodes with respect to this external adversary could be measured in terms of the entropy of $\mathcal{P}_0$, as given by Equation 1. In the case where an ad hoc routing protocol is resistant to this kind of analysis by means of extensive use of dummy traffic we could imagine that this a priori distribution is a uniform distribution with entropy $H_{\max} = \log_2(N)$.

Assume that a node N_j is taken over by the adversary, and that this node receives the message m. With the internal information of this node the adversary could then possibly gain some new information about which node that originated m, so that the probability distribution $\mathcal{P}_0$ can be updated to $\mathcal{P}_1$. As will be illustrated by the examples later this new information can for instance be about how many hops away the message was originated. If the node internal processing of the message m reveals the number of hops it has travelled or how many hops away to the destination it has left to travel, the adversary can in the worst case locate the position of the sending or receiving node of this message, e.g. the message was originated one hop away. If the message reveals that it was generated k hops away, the size of the anonymity set for the sending node will be

$$c_k = (1/2)\rho(2k - 1).$$

In our measurement model we want to combine the probabilities assigned to each node in this anonymity set with the a priori knowledge of the adversary, to form the new probability distribution $\mathcal{P}_1$.

Let Y denote the discrete random variable with probability mass function $q_k = P(Y = y_k)$, where q_k is the probability that a message m, received by the adversary node N_j, was originated at node k hops away, according to

the knowledge the adversary can derive from the internal information of node N_j. Assume that there is a maximum path length λ in the ad hoc network, measured in number of hops. If we assume the local node adversary to have no a priori knowledge of the probability of other network nodes as being the originator of the received message m, the probability that m was originated at a node k hops away is given by:

$$q_k = \frac{c_k}{\sum_{i=1}^{\lambda} c_i} \tag{2}$$

The entropy $H(Y) = -\sum_{k=1}^{\lambda} q_k \log_2(q_k)$ will express the adversary's uncertainty on which node that originated the message m, viewed *locally* from node N_j, we will combine this entropy measure with the measure of the a priori *global* view of the adversary using the Shannon *conditional entropy* $H(X|Y)$. The conditional entropy is not a measure of the uncertainty of the adversary in a specific attack scenario, but rather a measure of the adversary's average uncertainty given all possible local observations:

$$\begin{aligned} H(X|Y) &= -\sum_{i,k} P(x_i, y_k) \log_2 P(x_i|y_k) \\ &= -\sum_{k} q_k \sum_{i} P(x_i|y_k) \log_2 P(x_i|y_k). \end{aligned}$$

The conditional entropy measure is the average entropy of X, given Y, weighted according to the probability of getting a particular observation y_k. Let Z denote the discrete random variable describing the conditional probability that node N_i originated a message, given the observation y_k. Thus we have that $P_k(z_i) = P(x_i|y_k)$ and

$$H(X|Y) = \sum_{k} q_k H_k(Z), \tag{3}$$

where $H_k(Z)$ denotes the entropy of Z, given the observation y_k. In a specific attack scenario $P_k(z_i)$ would be the probability that N_i was the sending node, derived by an adversary that has an a priori knowledge of $\mathcal{P}_0$, and that by the capturing of the message m can see that it was originated k hops away.

In the case where $\mathcal{P}_0$ is uniformly distributed, the adversary only controls one network node and the adversary can derive that the message m, received by the network node controlled by the adversary, was generated k hops away, this observation will limit the set of potential sending nodes to only the nodes that are located k hops away. In this case the entropy measure will be reduced from $H(X) = H_{\max} = \log_2(N)$ without any observations, to $H_k(Z) = \log_2(c_k)$ given this particular observation.

If an adversary controls more than one node in the network, the anonymity set of senders, given an observed message arriving at one of the adversarial

nodes, could be further reduced. If the adversary controls half of the nodes in the network, that is we have a Passive-$\frac{N}{2}/N$ adversary, we could assume that on average half of the nodes in the anonymity set would be adversarial. In that case we can derive $H_k(Z) = \log_2(\frac{c_k}{2})$, and insert this into the conditional entropy measure given in Equation 3. More generally, if the adversary controls c out of N network nodes we get the measure $H_k(Z) = \log_2((1 - \frac{c}{N})c_k)$.

If the adversary has some a priori knowledge of the node's communication patterns, $\mathcal{P}_0$ will not be uniformly distributed. In this case we will have to find the value of $P_k(z_i) = P(x_i|y_k)$, which can be rewritten using Bayes' rule as

$$P(x_i|y_k) = \frac{P(y_k|x_i)P(x_i)}{\sum_{i=1}^{N} P(y_k|x_i)P(x_i)}, \tag{4}$$

where $P(y_k|x_i)$ is the probability that a node observes that a message was originated k hops away, given that node N_i generated this message. As we will see in the examples in the following section, this probability can be derived from properties of the specific routing protocol being used.

4. Examples of Measuring Anonymity

In this section we will illustrate by two examples of anonymous ad hoc routing protocols how the entropy measure can be used for quantification of the anonymity of an ad hoc routing protocol with respect to the previously described adversary model. We will first introduce the concept of *onion routing*, which is a technique used in different variations in many proposed anonymous routing protocols.

4.1 Onion routing

Onion routing is a variant of Chaum's mix-networks [Cha81], where messages are wrapped in layers of encryption with the keys of all intermediate nodes on the route to the destination. At each node a layer of encryption is peeled off before the node forwards the messages in random order. If for example a message m is to be sent from the node N_1 to N_4 via the intermediate nodes N_2 and N_3, the message sent to N_2 from N_1 would be

$$\{N_3, \{N_4, \{m\}_{k_4}\}_{k_3}\}_{k_2},$$

where the k_i are secret keys shared between N_1 and all the other nodes on the route. This message is called an onion, Some padding also has to be added to the onion, so that it has a constant size, otherwise the size of the onion would reveal the distance in number of hops from the sender to the recipient. The privacy of the sender and the receiver of a message relies on the fact that there should be no correspondence between incoming and outgoing messages from a node. In practice an external passive global adversary could just track the flow of messages through the network. To prevent this, an addition of dummy

traffic and different mixing strategies are applied as extra measures beside the routing protocol.

Most proposed anonymous ad hoc routing protocols, e.g. ANODR [KH03] and ARM [SP06], are on demand routing protocols that use onions in some way or another. The main idea of these protocols is that the source node N_s that is to send a message to the recipient node N_r, broadcasts a route request message that contains some information that only the recipient node can recognize (typically some information encrypted with a shared key between N_s and N_r). When nodes that are on the route, but not the recipient receives this route request they either keep some state information of this route request, or they add some encrypted information to the route request, so that later when the recipient node broadcasts the route reply message they know how to process and forward this message. When the source node N_s receives the route reply from N_r it can start to send data messages along the established route.

4.2 The ANODR Protocol

The route discovery part of ANODR uses a variant of onion routing where the source node broadcasts a route request message containing the inner core of an onion, as the route reply is forwarded throughout the network each node on the route adds a layer to this onion so that when the request reaches the recipient node the onion is wrapped with layers of encryption of all the intermediate nodes on the route. When the route reply is sent from the recipient node it contains this onion, and as the route reply traverses the route back to the source every node on the route peels off one layer of encryption from the onion. The onion is padded with random bits so that its size does not reveal the number of hops from the source or recipient node, but as noted by the authors of [ZWK+04], this padding only protects against external adversaries. An internal adversary controlling one of the nodes on the route will see the size of the onion and can from this knowledge deduce the number of hops away the message was originated.

When measuring the anonymity offered by the ANODR protocol in terms of the conditional entropy, assuming one compromised network node, and given the adversary's a priori knowledge $\mathcal{P}_0$, the term $P(y_k|x_i)$ in Equation 4 is equal to 1 if the node N_i is k hops away from the adversarial node receiving the message m, and equal to 0 otherwise. This means that we are simply reducing the anonymity set to the nodes k hops away and scaling the probabilities according to the a priori probability distribution.

If more than one node is compromised we need to exclude a number of nodes from the anonymity set according to the network proportion of adversarial nodes. One way of doing this when the a priori distribution is not uniform is to weight the sending probability of each node in the anonymity set according to the proportion of adversarial nodes as well as the a priori sending probability. If the adversary controls c out of N network nodes we would then get the

conditional entropy measure:

$$H(X|Y) = -\sum_k q_k \sum_i (1 - \frac{c}{N}) P(x_i|y_k) \log_2((1 - \frac{c}{N}) P(x_i|y_k)). \quad (5)$$

4.3 The ARM Protocol

The ARM protocol uses a probabilistic padding of onions and a probabilistic time-to-live scheme in the route discovery part of the protocol.

The length of route request messages grows as they traverse the network, so in order to prevent the disclosure of the distance the message has travelled, the source node N_s randomly selects a padding length of the route request message according to a specific probability distribution. This means that a neighbor node of N_s can calculate the probability that N_s was the originator of this route request message. For the route reply and data messages every node on the route chooses a time-to-live value according to a specific probability distribution, similarly in this case a neighbor node can calculate the probability that this message originated from the broadcasting node.

Corresponding route request and route reply messages carry the same pseudonym identifier, this allows an adversary to correlate the internal information about these particular messages in a probabilistic attack with an increasing number of malicious nodes as described in our adversary model.

If we only look at the route request messages, assume the padding length is drawn from the discrete probability distribution $\mathcal{R}$, where r_l is the probability that the padding length $l_{min} \leq l \leq l_{max}$ is chosen. A padding length of l means that the route request appears to a neighboring node of N_s to have been originated l hops from the real source node.

In our measurement model this would mean that if a node observes that according to the onion length the message was generated k hops away, the message could have been originated at a node as far as $k + l_{max}$ hops away. The probability $P(y_k|x_i)$ in Equation 4 would then be equal to r_l if the node N_i is $k + l$ hops away. The anonymity set for the possible sender nodes would also increase in size giving:

$$c_k = (1/2)\rho \sum_{i=k}^{k+l_{max}} (2i - 1)$$

So for the ARM protocol we are reducing the anonymity set to all nodes between k and $k + l_{max}$ hops away, and scaling the probabilities according to the a priori probability distribution as well as the probability distribution for the padding scheme. We can now analyze the anonymity in terms of the conditional entropy for different numbers of adversarial nodes by using Equation 5 as explained above. As an illustration of the anonymity measure we have in Figure 1 plotted the conditional entropy measure as a function of the proportion of adversary nodes for the ANODR and ARM protocols. In our

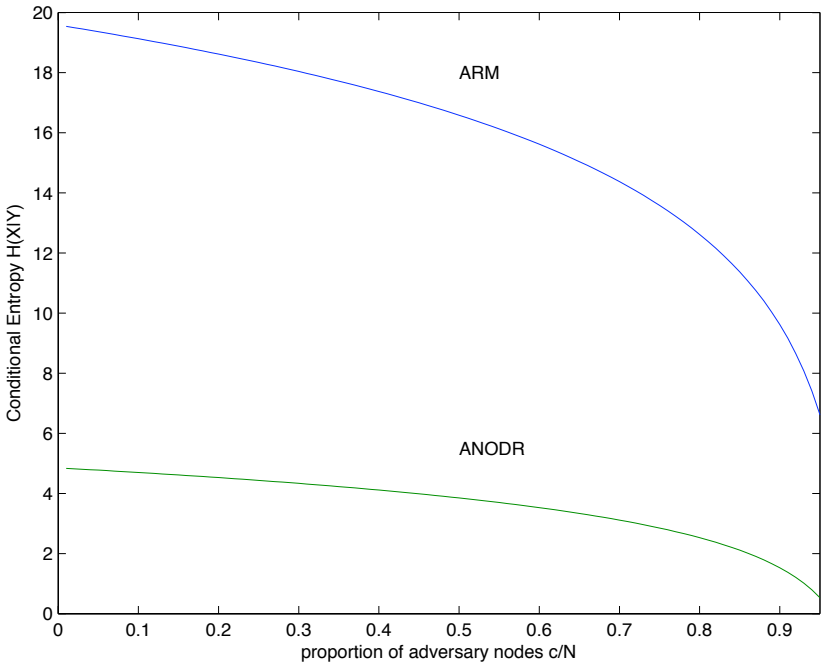


Figure 1. Illustration of the conditional entropy measure applied to the ANODR and ARM protocols

calculations we used the parameters $\rho = 8$, $\lambda = 6$ and $l_{max} = 3$, for simplicity we assumed that the distributions $\mathcal{P}_0$ and $\mathcal{R}$ were uniform.

5. Discussion and Conclusions

We have proposed a probabilistic system model for anonymous ad hoc routing protocols and showed how the information theoretical measure conditional entropy could be used for quantification of the average anonymity of the system as the adversary captures an increasing number of nodes in the network. We illustrated our approach by the examples of the ANODR and the ARM protocol, but the approach could be generally applied to ad hoc routing protocols that are using probabilistic mechanisms to achieve anonymity.

It should be noted that the weakness of the padding of onions in the ANODR protocol, allowing for an internal node to deduce the number of hops from source node, was fixed in an updated version of the protocol. To achieve secure onion routing the padding of onions should be done in such a way that a node receiving a padded onion is unable to tell if it was padded or not, the cryptographic issues involved in such a padding scheme were treated formally

by Camenisch and Lysyanskaya [CL05]. However, in an ad hoc routing setting we need to be concerned about the efficiency of computations, so there is always a trade-off between the security and usability of a protocol, which sometimes rules out the use of provable secure but computationally heavy solutions.

There are many possible directions for further research based on this approach. When designers of a protocol want to achieve a statistical notion of anonymity, meaning that the probability of determining the sender or recipient of a message should not exceed a certain threshold, as described by [KEB98] and [THV04], our approach could possibly be used in an analysis for maximising anonymity while minimising the computational cost. We proposed an analytical model for calculating anonymity in terms of entropy, giving a weighted average entropy measure. We used a simple network topology model for our calculations, to further improve the measurement model we could in our future work replace the topology model with simulations of many different network topologies and routes, with a varying number of adversarial nodes.

References

[BEKXK04] Azzedine Boukerche, Khalil El-Khatib, Li Xu, and Larry Korba. SDAR: A Secure Distributed Anonymous Routing Protocol for Wireless and Mobile Ad Hoc Networks. In *29th Annual IEEE International Conference on Local Computer Network, LCN'04*, pages 618–624. IEEE, 2004.

[Cha81] David L. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[CL05] Jan Camenisch and Anna Lysyanskaya. A Formal Treatment of Onion Routing. In *Advances in Cryptology, Crypto 2005*, LNCS 3621, pages 169–187. Springer, 2005.

[CS06] Sebastian Clauß and Stefan Schiffner. Structuring anonymity metrics. In *DIM '06: Proceedings of the second ACM workshop on Digital identity management*, pages 55–62, New York, NY, USA, 2006. ACM.

[DSCP02] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[DTD07] Claudia Diaz, Carmela Troncoso, and George Danezis. Does Additional Information Always Reduce Anonymity? In *WPES '07: Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 72–75, New York, NY, USA, 2007. ACM.

[HP04] Yih-Chun Hu and Adrian Perrig. A Survey of Secure Wireless Ad Hoc Routing. *IEEE Security & Privacy*, 2(3):28–39, 2004.

[KEB98] Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Proceedings of Information Hiding Workshop (IH 1998)*. Springer-Verlag, LNCS 1525, 1998.

[KH03] Jiejun Kong and Xiaoyan Hong. ANODR: ANonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks. In *ACM MobiHoc'03*. ACM, 2003.

[PH05] Andreas Pfitzmann and Marit Hansen. Anonymity, Unobservability, and Pseudonymity: A Consolidated Proposal for Terminology. Draft, December 2005.

[RR98] Michael Reiter and Aviel Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.

[SD02] Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[Sey06] Stefaan Seys. *Cryptographic Algorithms and Protocols for Security and Privacy in Ad Hoc Networks*. PhD thesis, Katholieke Universiteit Leuven, 2006.

[SP06] Stefaan Seys and Bart Preneel. ARM: Anonymous Routing Protocol for Mobile Ad hoc Networks. In *Proceedings of the 20th IEEE International Conference on Advanced Information Networking and Applications - Workshops (AINA 2006)*. IEEE, 2006.

[SW06] Vitaly Shmatikov and Ming-Hsiu Wang. Measuring Relationship Anonymity in Mix Networks. In *WPES '06: Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 59–62, New York, NY, USA, 2006. ACM.

[THV04] Gergely Tóth, Zoltán Hornák, and Ferenc Vajda. Measuring Anonymity Revisited. In Sanna Liimatainen and Teemupekka Virtanen, editors, *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, Espoo, Finland, November 2004.

[ZLL05] Yanchao Zhang, Wei Liu, and Wenjing Lou. Anonymous Communications in Mobile Ad Hoc Networks. In *IEEE INFOCOM 2005*. IEEE, 2005.

[ZWK+04] Bo Zhu, Zhiguo Wan, Mohan S. Kankanhalli, Feng Bao, and Robert H. Deng. Anonymous Secure Routing in Mobile Ad-Hoc Networks. In *29th Annual IEEE International Conference on Local Computer Network, LCN'04*, pages 102–108. IEEE, 2004.

Paper D

TSR: Trust-based Secure MANET Routing using HMMs

Marie E. G. Moe, Bjarne E. Helvik and Svein J. Knapskog

In Proceedings of the 4th ACM International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2008)

ACM. Vancouver, Canada. October 27-31, 2008

Is not included due to copyright

Paper E

Learning Trust in Dynamic Multiagent Environments using HMMs

Marie E. G. Moe, Mozhgan Tavakolifard and Svein J. Knapskog

In Proceedings of The 13th Nordic Workshop on Secure IT Systems (NordSec 2008)

Copenhagen, Denmark. October 9-10, 2008

LEARNING TRUST IN DYNAMIC MULTIAGENT ENVIRONMENTS USING HMMS

Marie Elisabeth Gaup Moe, Mozhgan Tavakolifard and Svein Johan Knapskog
Centre for Quantifiable Quality of Service in Communication Systems
Norwegian University of Science and Technology
O.S. Bragstads plass 2E, N-7491 Trondheim, Norway
{marieeli,mozhgan,knapskog}@q2s.ntnu.no

Abstract In open multiagent systems, agents are owned by a variety of stakeholders and can enter and leave the system at any time. Therefore, trust is a fundamental concern in effective interactions which is a key component of such systems. In this paper, we propose a trust model for autonomous agents in mulitagent environments based on hidden Markov models and reinforcement learning. By this combination, the reliability of the hidden Markov model will be improved since its parameters are re-estimated after training of the model with the reinforcement learning module.

1. Introduction

The rapidly changing environments of the Internet suffer from problems related to fragile trustworthiness of its millions of active entities, which can be humans or mobile agents. This problem is nontrivial, as more and more commercial transactions get carried out over the Internet. Therefore, devising an effective approach for verification of trustworthiness in such complex environments is essential, since trust mechanisms play a key role in the security of the entities. Also the trust establishment is nontrivial, since the traditional and social means of trust cannot be applied directly to the virtual settings of these environments, because in many cases the involved parties did not have any previous interaction. In such scenarios, trust management techniques may be used to stimulate service quality and acceptable user behavior in online markets and communities, and also sanction possible unacceptable user behavior.

Application of autonomous agents in large-scale open distributed systems presents a number of new challenges such as:

- Agents with different characteristics can enter the system and interact with one another.

- Each agent tries to maximize its individual utility because it represents a specific stakeholder with various objectives.
- Agents may change their identities on re-entering the system to avoid punishment for any past wrong doing.
- Agents should decide how, when, and with whom to interact without any guarantees that the interaction will actually achieve the desired benefits.

Agents are faced with significant degrees of uncertainty in making decisions since it is impossible to obtain perfect information about the environment and the interaction partners properties. In such circumstances, agents have to establish appropriate trust in each other in order to minimize the impact of the uncertainty associated with interactions [RHJ05].

The goal of an agent in a dynamic environment is to make optimal trust decisions over time. Learning trust serves such a purpose by biasing the agent's action choices through information gathered over time. An agent can base its action choice on prediction of the other agents' behaviors or directly on the reward (the outcome of the interaction) received from them. Reinforcement learning is a systematic method that associates an agent's action with its rewards.

In reinforcement learning, an agent need not explicitly model other agents since its action can be directly based on the rewards. Thus this learning method is particularly useful for cases where agents have little knowledge about each other. An agent in a multiagent system may know little about others because information is distributed. Even when an agent has some prior information about others, the behavior of others may change over time. It is therefore natural to apply a learning algorithm.

Our model consists of trust estimation and trust learning modules. The former and latter are constructed from *Hidden Markov Models* (HMM) and *Reinforcement Learning (RL)*, respectively. The model parameters of the HMM are re-estimated after having learnt about its environment from the reinforcement learning module. The proposed method enables us to improve the model reliability when dealing with a dynamic environment that changes over time.

2. Related Work

In [HCD05] a trust model using a Markov model is proposed. In this work the Markov chain is defined as the chain of aggregated reputation values corresponding to a sequence of consecutive time slots. The current state vector shows the repute value of the reputation queried at time slot N.The Markov matrix of a given agent denotes the probability of that agent transiting from one trustworthiness level to another trustworthiness level based on its past behavior captured using the Markov chain. In order to determine the probability of an agent transiting from trustworthiness level A to trustworthiness level B, based on the Markov chain, they use the ratio of the number of times

that agent has transited from trustworthiness level A to trustworthiness level B to that of the total number of time that the agent has transited from trustworthiness level A to any other trustworthiness level. The future state vector of the agent is determined by multiplying the current state vector with the Markov matrix. The same authors also proposed a method for determining the effectiveness of their Markov model for predicting the future trustworthiness value of a given agent by utilizing simulation methods [HCD06]. This paper presents the simulation method that they employed in order to determine the effectiveness of the Markov model in detail.

In [SKN06], the authors compare the effectiveness of probabilistic computational trust systems. They conclude that most probabilistic trust models are unrealistic, as the models allows for no dynamic behavior, and outline the idea of a trust model based on a hidden Markov model to cope with this problem. In [SPX04], the authors have developed a hidden Markov model based approach to measuring an agent's reputation as a recommender. This approach models chained recommendation events as an HMM. The features of the trust model are: (1) no explicit requirement of chained recommendation reputations; (2) flexible recommendation network with presence of loops; and (3) integration of learning speed into trust evaluation reliability. In [NTH+06] an architecture for trust management in ubiquitous environments that deals with digital signatures and user presence in a uniform framework is proposed. This architecture includes inferences about user presence from incomplete sensor signals based on an HMM.

3. The Proposed Model

Our model consists of trust estimation and trust learning modules constructed from *Hidden Markov Models* (HMM) and *Reinforcement Learning (RL)* as depicted in Figure 1. The model parameters of the HMM are re-estimated after having learnt about its environment from the reinforcement learning module. The proposed method enables us to improve the model reliability when dealing with a dynamic environment that changes over time. Similar approaches combining reinforcement learning and HMMs applied to motion recognition can be found in [HYU02] and [HMN04].

In the following sections we will start with describing some assumptions and limitations of this work, thereafter we present the stochastic modeling approach and details of the hidden Markov modeling.

3.1 Model States

An agent can be in a *trusted*, *neutral* or *untrusted* state at any given time. An agent is in an untrusted state if it has been behaving in a malicious way in previous interactions, it is in a trusted state if it has shown good behavior. If its behavior has been a mixture of good and bad, or if it has not yet given any signs of its behavior, it is in the neutral state.

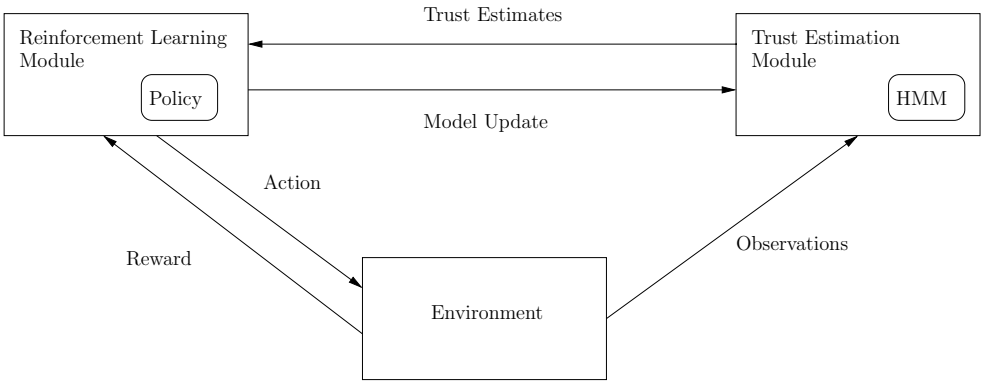


Figure 1. Architecture of the proposed trust model

When an agent joins the system it is most likely in a neutral state, but there is also a small probability that it is in the untrusted state. Since it has not yet interacted with any of the other agents, no positive observations of its behavior have been made, so it can not be in a trusted state.

After joining the system the agent has a probability of becoming trusted or untrusted as it is interacting with other agents, depending on its behavior. We model trust as a dynamic variable, changing with time. This allow us to capture the behavioral characteristics of agents that are behaving good for a certain time, but then suddenly start misbehaving.

3.2 Modeling of Agent Trustworthiness

In this section we will present an HMM for the trust relationship of agents in a multiagent system.

We model the agent interaction as a stochastic process. This means that we assume that there is a random time interval between each agent interaction and that the behavior of an agent is only dependent on the current state of the agent. The *state* of an agent can be characterized by whether or not it is behaving in a malicious manner in its interactions with other agents. When using a continuous time Markov model to model the state of an agent, we make the following assumptions; all information about the agent is contained in the state, observations are independent given the current state, and state occupation time is negatively exponentially distributed.

When agents are interacting, an agent makes its opinion about the trustworthiness of the other agent based on the outcome of the interaction. After a random time interval these two agents meet again, and based on their belief about the other agent's trustworthiness they may decide whether or not to make an interaction. Since an agent's behavior can be changing with time it is not necessarily the case that an agent is in the same state as it were at the

last encounter. An agent can only do its best guessing about the trustworthiness state of an other agent based on its own previous direct experiences with said agent and *recommendations* from other agents in the system. This means that the system state is hidden, and hence we use the HMM approach.

The system we consider is a multiagent system and we want to use the model to estimate the behavior of each single agent. An agent in the system rates all of the other agents after an interaction and uses an HMM per agent to decide and predict whether or not another agent is malicious. The HMM is updated from observations, that is the ratings after direct experiences or recommendations requested from other agents.

An HMM consists of a finite set of N hidden states $S = \{s_1, \ldots, s_N\}$ with an associated probability distribution. The state of the monitored agent is described by a discrete time Markov chain $\boldsymbol{x}_k = x_1, x_2, \ldots$ where $x_k \in S$ is the possibly hidden state of the node at sampling instant k. $\boldsymbol{P}_k = \{p_{ij}^k\}$ is the set of state transition probabilities, $p_{ij}^k = P(x_{k+1} = s_j \mid x_k = s_i)$, $1 \leq i, j \leq N$, where x_k is the current state of the system. $\pi = \{\pi_i\}$ is the initial state distribution, where $\pi_i = P(x_1 = s_i)$, $1 \leq i \leq N$. The output from the agent ratings is classified by the set of observation symbols $V = \{v_1, \ldots, v_M\}$. Let $\boldsymbol{y}_k = y_1, y_2 \ldots$ denote the sequence of observations, where $y_k \in V$ is the observation made at sampling instant k. The HMM consists of two stochastic processes; the hidden process $\boldsymbol{x}_k$, and the observable process $\boldsymbol{y}_k$ that depends on $\boldsymbol{x}_k$. The relation between $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ is described by the probability distribution matrix $\boldsymbol{B} = \{b_j(m)\}$, where $b_j(m) = P(y_k = v_m \mid x_k = s_j)$, for $1 \leq j \leq N$, $1 \leq m \leq M$. See for instance [Rab90] for a more extensive introduction to HMMs.

In our model we define three states, that can be characterized by the behavior of the agent, thus $N = 3$ and each individual state is denoted $S = \{s_1, s_2, s_3\}$. The first state is the *trusted* state s_1, where the agent is not showing any malicious behavior, the second state is the *neutral* state s_2, where the outcome of interactions can be ambiguous, the third state is the *untrusted* state s_3 where the agent is showing malicious behavior.

We have not made any assumptions about time between observations, and there is no direct relation between observations and state-changes. As a consequence the system could have made zero, one or more transitions during the time between to successive observations.

The time when observation number k is produced is denoted t_k. Time between observation $k-1$ and observation k is denoted $\delta_k = t_k - t_{k-1}$.

3.3 State Probability Distribution

The transition rate matrix $\Lambda = (\lambda_{ij})$ is describing the dynamics of the system. To simplify the notation of equations and algorithms we will use i and j instead of s_i and s_j. The relation between system states and the transition rates is given by

$$\lambda_{ij} = \begin{cases} \lim_{dt \to 0} \dfrac{P(\boldsymbol{x}(t+dt) = j | \boldsymbol{x}(t) = i)}{dt} & \text{if } i \neq j \\ \sum_{j \neq i, j=1}^{N} -\lambda_{ij} & \text{if } i = j \end{cases}. \tag{1}$$

Since observations are received at irregular intervals, the running transition probabilities $p_{ij}^k = P(x(t+\delta_k) = j | x(t) = i)$ depend on the time since last observation δ_k, and have to be calculated each time an observation is received. The running transition probability matrix $\boldsymbol{P}_k = (p_{ij}^k)$ can be derived from Kolmogorov's equations [Ros03] as follows

$$\boldsymbol{P}_k = e^{\Lambda \delta_k}. \tag{2}$$

For large state spaces this calculation can be quite expensive, but in our case the state space is small, and the calculations inexpensive. Let $\gamma_k = (\gamma_k(i))$ denote the state probability distribution at time t_k given all observations received until time t_k, $\gamma_k(i) = P(x_k = i | \boldsymbol{y}_k)$ where $\boldsymbol{y}_k = y_1, \ldots, y_k$. We will use ten observation symbols $V = \{v_1, v_2, \ldots, v_{10}\}$, where the first five symbols are the ratings an agent make after a direct interaction and the last five symbols are ratings received as recommendations from other agents. The ratings are given in the form of trustworthiness values ranging from 1 to 5, where 1 corresponds to "very trustworthy", 2 to "trustworthy", 3 to "moderate", 4 to "untrustworthy", and 5 to "very untrustworthy". Algorithm 4 is used to update the current state distribution γ_{k-1}, based on the following inputs: k the observation index, $\gamma = \gamma_{k-1}$ the current state distribution, $y = y_k$ the current observation, and $\delta = \delta_k$ the time between the current observation and last observation. In addition to the dynamic variables listed above, the following parameters are assumed to be available for the algorithm: the transition rates Λ, the initial state distribution π , and the two observation probability matrices $\boldsymbol{B}^{\psi}$, where $\boldsymbol{B}^1$ is used for the direct observations $v_1, \ldots, v_5$, and $\boldsymbol{B}^2$ is used for the implicit observations in the form of recommendations $v_6, \ldots, v_{10}$.

Algorithm 4 was originally proposed in [HMK08], it is based on dynamic programming and uses a set of temporary variables. During the processing of observation y_k the value stored in $\alpha(i)$ represents the following probability $\alpha(i) = P(\boldsymbol{y}_k, x_k = s_i)$, also known as the *forward variable*. By using dynamic programming in the estimation of γ, the complexity of an update is reduced from $O(2kN^k)$ for a straight forward calculation, to $O(N^2)$. Scaling of the $\alpha(i)$ is used in order to prevent problems related to underflow, for more details see the forward-backward procedure described in [Rab90]. It should be noted that Algorithm 4 is an on-line algorithm and very efficient, it does not require the agents to keep any history of past observations in memory. The history of observations and the values of some of the running variables are, however, required for the more computationally expensive re-estimation of model parameters as explained later.

Algorithm 4 Update state probability distribution

Require: $k, \psi, \gamma, y, \delta$
  $P_k \leftarrow e^{\Lambda\delta}$
  $B \leftarrow B^{\psi}$
  if $k = 1$ **then**
    for $i = 1$ to N **do**
      $\alpha(i) \leftarrow b_i(y)\pi_i$
      $\gamma(i) \leftarrow \frac{b_i(y)\pi_i}{\sum_{j=1}^{N} b_j(y)\pi_j}$
    end for
  else
    for $i = 1$ to N **do**
      $\alpha(i) \leftarrow b_i(y) \sum_{j=1}^{N} \gamma(j) p_{ji}^{k}$
    end for
    for $i = 1$ to N **do**
      $\gamma(i) \leftarrow \frac{\alpha(i)}{\sum_{j=1}^{N} \alpha(j)}$
    end for
  end if
  return γ

4. Learning of Model Parameters

The parameters of an HMM are usually set by offline training of the model with a large data set. Since we want the model to reflect the more realistic dynamic behavior of the multiagent system and also optimize the agents' trust-related behavior, we will use an online learning of the HMM parameters with *reinforcement learning*. Reinforcement learning (RL) [SB98] is a machine learning technique for solving decision problems of mapping actions to states based on interactions with the environment. The actions of an agent in the multiagent system could for instance be whether or not it should interact with another agent, based on its belief about the state of the other agent derived from the HMM. Such a mapping from state to action is called a *policy*. In RL the agents will learn policies based on feedback from the environment that is calculated based on a *reward function*.

A simple reward function for the multiagent trust model can be defined as follows:

1 If an interaction was made, and the agent's rating of the other agent's behavior was given the values 1, 2 or 3, a positive reward is given.

2 If an interaction was made, and the agent's rating of the other agent's behavior was given the values 4 or 5, a negative reward is given.

3 If no interaction was made, a zero reward is given.

The RL framework also includes a *value function* $Q(s, a)$ which estimates the reward obtained if action a is performed in state s.

Q-learning [WD92] is a well-known RL algorithm that updates the value function in each step so that the agent policy converges to the optimal one. Q-learning works even though the state transition probabilities are unknown to the agent. In our approach we will use the output of the Q-learning to improve the HMM by updating the state transition rate matrix according to the learned optimal policy.

Since the current state of an agent is hidden in our trust model, we will instead use the state probability distribution γ that is calculated after each observation, and learn the function $Q(\gamma, a)$. A variant of the Q-learning algorithm suitable for this case where the current state is only partially observable, and accounting for the fact that the domain of Q is not discrete and finite, can be found in [Chr92]. Following this approach we associate a value $q(i, a)$ with each hidden state s_i, and approximate $Q(\gamma, a)$ as

$$Q(\gamma, a) \approx \sum_{i=1}^{N} \gamma(i) q(i, a).$$

Learning Q is done by adjusting all the q values after each action a and immediate reward r according to the Q-learning rule:

$$q(i, a) = (1 - \eta\gamma_k(i))q(i, a) + \eta\gamma_k(i)(r + \sigma \max_a Q(\gamma_{k+1}, a)), \tag{3}$$

where η is the learning rate and σ is a discount factor. The learning proceeds as follows, when an agent encounters another agent it will get the current state probability distribution γ_k belonging to this particular agent from its corresponding HMM and execute the action with the largest $Q(\gamma_k, a)$. After the action is performed, the agent receives the reward, the next step state probability distribution γ_{k+1} is output from the HMM, and the Q-learning updating rule from Equation 3 is applied. The process is repeated at the next encounter between the agents. It should be noted that the observations to the HMM coming from recommendations will not result in actions or rewards, only the direct experiences in the form of agent interaction will trigger the reinforcement learning module in the trust model.

When the agent encounters another agent for the first time, the parameters of the HMM will be set to default values, and the model might not properly predict the system dynamics. A newcomer to the system should not be expected to be in the trusted state, as such a starting point would encourage agents to change their identities and re-enter the system frequently. It is therefore better to assume that agents are most likely neutral or untrustworthy to start with, and then they have to prove themselves trustworthy by behaving good in the interactions.

As the agent learns about the behavior of the other agent through direct experience and recommendations, the HMM parameters should be updated in order to improve the predictiveness of the model. We suggest that the state transition rates Λ and the observation probabilities $\boldsymbol{B}$ of the HMM are updated after a predetermined number of Q-learning steps, e.g. by the Baum-Welch algorithm which finds the maximum likelihood parameter estimate. See [Rab90] for a detailed explanation of the Baum-Welch algorithm.

Given a sequence of K observations the Baum-Welch algorithm makes use of the *backward variable* $\beta_k(i) = P(y_{k+1}, \ldots, y_K | x_k = s_i)$, which is the probability of the observation sequence from next step and until the end given that we are in the state s_i at time-step k. The backward variable is found inductively by setting $\beta_K(i) = 1$ and then recursively calculating $\beta_k(i) = \sum_{j=1}^{N} p_{ij} b_j(y_{k+1}) \beta_{k+1}(j)$. The re-estimation procedure calculates the joint probability $\xi(i,j) = P(x_k = s_i, x_{k+1} = s_j | \boldsymbol{y})$ of being in state s_i at time-step k and state s_j in time-step $k+1$ as

$$\xi(i,j) = \frac{\alpha_k(i) p_{ij} b_j(y_{k+1}) \beta_{k+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_k(i) p_{ij} b_j(y_{k+1} \beta_{k+1}(j))}.$$

For re-estimation of p_{ij} according to Baum-Welch we use

$$p_{ij} = \frac{\sum_{k=1}^{K-1} \xi(i,j)}{\sum_{k=1}^{K-1} \gamma_k(i)},$$

where the nominator is the expected number of transitions from state s_i to state s_j, and the denominator is the expected number of transitions from state s_i. For re-estimation of the observation symbol probabilities the following equation is used

$$b_j(m) = \frac{\sum_{k=1, \text{s.t.} y_k = v_m}^{K} \gamma_k(j)}{\sum_{k=1}^{K} \gamma_k(j)},$$

where the nominator is the expected number of times in state s_j and observing the symbol v_m and the denominator is the expected number of times in state s_j.

Algorithm 5 implements the re-estimation of the parameters. In order to avoid problems related to underflow we use scaling of the backward variable as described in [Rab90] and [Rah00]. The proof of correctness of the scaling procedure is given in the Appendix A. Algorithm 5 takes as input all the K different α_k and γ_k vectors, which are calculated by Algorithm 4, as well as all the different $\boldsymbol{P}_k$. This means that these values need to be stored in the agent's memory together with the history of observations for the re-estimation procedure. To simplify the implementation we will combine the two observation probability matrices into one single matrix $\boldsymbol{B}$ for the re-estimation algorithm. The observation probabilities are modeling the uncertainties associated with

Algorithm 5 Re-estimation of parameters

Require: $\boldsymbol{P}, \boldsymbol{\gamma}, \boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{B}, K$
  for $i = 1$ to N **do**
    $\hat{\beta}_K(i) \leftarrow \frac{1}{\sum_{i=1}^{N} \alpha_K(i)}$
  end for
  for $k = K - 1$ to 1 **do**
    for $i = 1$ to N **do**
      $\hat{\beta}_k(i) \leftarrow \frac{1}{\sum_{j=1}^{N} \alpha_k(j)} \sum_{j=1}^{N} p_{ij}^k b_j(y_{k+1}) \hat{\beta}_{k+1}(j)$
      for $j = 1$ to N **do**
        $\xi^k(i,j) \leftarrow \gamma_k(i) p_{ij}^k b_j(y_{k+1}) \hat{\beta}_{k+1}(j)$
      end for
      $\hat{\gamma}_k(i) \leftarrow \sum_{j=1}^{N} \xi^k(i,j)$
    end for
  end for
  for $i = 1$ to N **do**
    for $j = 1$ to N **do**
      $p_{ij} \leftarrow \frac{\sum_{k=1}^{K-1} \xi^k(i,j)}{\sum_{k=1}^{K-1} \hat{\gamma}_k(i)}$
    end for
  end for
  for $m = 1$ to M **do**
    for $j = 1$ to N **do**
      $b_j(m) \leftarrow \frac{\sum_{k=1, \text{s.t.} y_k = v_m}^{K} \hat{\gamma}_k(j)}{\sum_{k=1}^{K} \hat{\gamma}_k(j)}$
    end for
  end for
  return $\boldsymbol{B}, \boldsymbol{P}$

observations. This means that when we re-estimate $b_j(m)$ elements associated with the direct observations, i. e. the agent's own ratings after interactions, we are evaluating the reliability of the agent itself when it comes to making good trust decisions. The elements associated with the recommendations are re-estimated as an evaluation of the reliability of the recommending agents. In this context we do not model the trustworthiness of each recommender separately, but this could be done as an extension of the model.

5. Conclusions and Future Work

We have proposed a novel trust model for multiagent systems where the goal of an agent is to make optimal trust decisions over time in a dynamic environment. An agent bases its action choice on a prediction of the other agents' behaviors according to the HMM trust estimation module following the Q-learning greedy policy. Since this is a policy that tends to the optimal

one over time, the model learning algorithm will be training the HMM with sequences of observations that will positively impact the end goal.

As this is just a preliminary theoretical model, without any simulation results yet, there are many directions of research that may be explored to improve our work. We would like to see if it is possible to additionally train the model to making the best decisions about when to ask other agents for recommendations. Application of the model to a variety of trust scenarios and a comparison to other proposed trust models is of course the prime interest of our future work.

Appendix: Scaling of the forward and backward variables

We want to use scaling of the forward variable and the backward variable, to avoid problems related to underflow. Since, without scaling, these variables consist of a large number of terms of value significantly less than 1. This appendix will give a detailed explanation of the scaling procedure for the implementation of the Algorithms 4 and 5, as proposed in [Rab90] and [Rah00]. We have that

$$\alpha_k(i) = \begin{cases} \pi_i b_i(y_k) & \text{if } k = 1 \\ b_i(y_k) \sum_{j=1}^{N} \alpha_{k-1}(j) p_{ji} & \text{if } k > 1 \end{cases} \tag{A.1}$$

where $\alpha_k(i)$ is the forward variable without any scaling. We want to compute the scaled forward variable

$$\hat{\alpha}_k(i) = C_k \alpha_k(i), \tag{A.2}$$

with scaling coefficient

$$C_k = \frac{1}{\sum_{j=1}^{N} \alpha_k(j)}. \tag{A.3}$$

Let us use the notation $\bar{\alpha}_k(i)$ for the running value of the forward variable as it is used in the implementation before scaling, and c_k for the running value of the scaling coefficient. The recursion for calculating the scaled forward variable can then be expressed as:

Initialization:

$$\bar{\alpha}_1(i) = \alpha_1(i)$$

$$\hat{\alpha}_1(i) = \frac{\alpha_1(i)}{\sum_{j=1}^{N} \alpha_1(j)}$$

For $k > 1$:

$$\bar{\alpha}_k(i) = b_i(y_k) \sum_{j=1}^{N} \hat{\alpha}_{k-1}(j) p_{ji}$$

$$c_k = \frac{1}{\sum_{j=1}^{N} \bar{\alpha}_k(j)}$$

$$\hat{\alpha}_k(i) = c_k \bar{\alpha}_k(i)$$

We want to prove that this recursion realizes the scaling as expressed in Equation A.2. For $k = 1$ the scaling coefficient is $c_1 = C_1 = \frac{1}{\sum_{j=1}^{N} \alpha_1(j)}$, so the scaling is exactly as in Equation

A.2. For $k > 1$ we can use a proof of induction as follows:

$$\begin{aligned}
\bar{\alpha}_k(i) &= b_i(y_k) \sum_{j=1}^{N} \hat{\alpha}_{k-1}(j) p_{ji} \\
&= b_i(y_k) \sum_{j=1}^{N} C_{k-1} \alpha_{k-1}(j) p_{ji} && \text{(by application of Equation A.2)} \\
&= C_{k-1} \alpha_k(i) && \text{(by application of Equation A.1)}
\end{aligned}$$

This gives us the relation

$$c_k = \frac{1}{\sum_{j=1}^{N} \bar{\alpha}_k(j)} = \frac{1}{C_{k-1} \sum_{j=1}^{N} \alpha_k(j)} \tag{A.4}$$

We can then express $\hat{\alpha}_k(i)$ as

$$\hat{\alpha}_k(i) = c_k \bar{\alpha}_k(i) = \frac{C_{k-1} \alpha_k(i)}{C_{k-1} \sum_{j=1}^{N} \alpha_k(j)} = \frac{\alpha_k(i)}{\sum_{j=1}^{N} \alpha_k(j)}$$

so the recursion used is indeed realizing the scaling as given in Equation A.2, which was what we wanted to show. Furthermore, by combining Equations A.3 and A.4, we get the relation

$$C_k = C_{k-1} c_k = \prod_{\kappa=1}^{k} c_\kappa$$

Recall the definition of the backward variable:

$$\beta_k(i) = \begin{cases} 1 & \text{if } k = K \\ b_i(y_{k+1}) \sum_{j=1}^{N} \beta_{k+1}(j) p_{ij} & \text{if } k > K \end{cases}$$

For the scaling of the backward variable let us define the scaling coefficient

$$D_k = \prod_{\kappa=k}^{K} c_\kappa$$

which gives us the relation

$$C_k D_{k+1} = \prod_{\kappa=1}^{k} c_\kappa \prod_{\kappa=k+1}^{K} c_\kappa = C_K$$

Let us denote by $\hat{\beta}_k(i)$, the scaled backward variable

$$\hat{\beta}_k(i) = D_k \beta_k(i) \tag{A.5}$$

and let us denote by $\bar{\beta}_k(i)$, the running backward variable as it is used in the implementation before scaling. The recursion for calculating the scaled backward variable is given by

$$\begin{aligned}
&\text{Initialization:}\\
&\qquad \bar{\beta}_K(i) = \beta_K(i) = 1\\
&\qquad \hat{\beta}_K(i) = D_K = c_K\\
&\text{for } k < K\text{:}\\
&\qquad \bar{\beta}_k(i) = b_i(y_{k+1}) \sum_{j=1}^{N} \hat{\beta}_{k+1}(j) p_{ij}\\
&\qquad \hat{\beta}_k(i) = c_k \bar{\beta}_k(i)
\end{aligned}$$

Note that the running value of the scaling coefficient will be the same c_k as was used in the scaling of the forward variable. For $k = K$ the Equation A.5 is trivially fulfilled. The correctness of the recursion for $k < K$ can be shown by induction as follows

$$\begin{aligned}
\bar{\beta}_k(i) &= b_i(y_{k+1}) \sum_{j=1}^{N} \hat{\beta}_{k+1}(j) p_{ij}\\
&= b_i(y_{k+1}) \sum_{j=1}^{N} D_{k+1} \beta_{k+1}(j) p_{ij}\\
&= D_{k+1} \beta_k(i)
\end{aligned}$$

we can then rewrite $\hat{\beta}_k(i)$ as

$$\hat{\beta}_k(i) = c_k \bar{\beta}_k(i) = c_k D_{k+1} \beta_k(i) = D_k \beta_k(i)$$

hence, the recursion is realizing the scaling in Equation A.5.

References

[Chr92] Lonnie Chrisman. Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 183–188. AAAI Press, 1992.

[HCD05] F. K. Hussain, E. Chang, and T. S. Dillon. Markov model for modeling and managing dynamic trust. In *Proceedings of the 3rd IEEE International Conference on Industrial Informatics, INDIN'05*, pages 725–733. IEEE, 2005.

[HCD06] F. K. Hussain, E. Chang, and T. S. Dillon. Quantification of the Effectiveness of the Markov Model for Trustworthiness Prediction. In *Proceedings of the International Conference 9th Fuzzy Days*. Springer, 2006.

[HMK08] Kjetil Haslum, Marie Elisabeth Gaup Moe, and Svein Knapskog. Real-time Intrusion Prevention and Security Analysis of Networks using HMMs. In *Proceedings of the 4th IEEE LCN Workshop on Network Security (WNS 2008)*. IEEE, 2008.

[HMN04] K. Hamamoto, K. Morooka, and H. Nagahashi. Motion Recognition By Combining HMM and Reinforcement Learning. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2004.

[HYU02] M. A. T. Ho, Y. Yamada, and Y. Umetani. An HMM-based Temporal Difference Learning with Model-Updating Capability for Visual Tracking of Human Communicational Behaviors. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR'02)*. IEEE, 2002.

[NTH+06] J. Noda, M. Takahashi, I. Hosomi, H. Mouri, Y. Takata, and H. Seki. Integrating presence inference into trust management for ubiquitous systems. In *Proceedings of the eleventh ACM symposium on Access control models and technologies*, pages 59–68. ACM Press New York, 2006.

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.

[Rah00] Ali Rahimi. An Erratum for "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". http://alumni.media.mit.edu/~rahimi/rabiner/rabiner-errata/rabiner-errata.html, 2000.

[RHJ05] S.D. Ramchurn, D. Huynh, and N.R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(01):1–25, 2005.

[Ros03] Sheldon M. Ross. *Introduction to Probability Models*, chapter Continuous-Time Markov Chains, pages 349–390. Academic Press, New York, 8th edition, 2003.

[SB98] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

[SKN06] V. Sassone, K. Krukow, and M. Nielsen. Towards a Formal Framework for Computational Trust. In *Proceedings of the 5th International Symposium on Formal Methods for Components and Objects (FMCO 2006)*, volume LNCS 4709, page 175. Springer, 2006.

[SPX04] W. Song, V.V. Phoha, and X. Xu. The HMM-Based Model for Evaluating Recommender's Reputation. In *Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04)*, pages 209–215. IEEE, 2004.

[WD92] Christopher Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

Paper F

Comparison of the Beta and the Hidden Markov Models of Trust in Dynamic Environments

Marie E. G. Moe, Bjarne E. Helvik and Svein Knapskog

To appear in the Proceedings of the third IFIP WG 11.11 International Conference on Trust Management (IFIPTM 2009)

Springer. West Lafayette, USA. June 15-19, 2009

COMPARISON OF THE BETA AND THE HIDDEN MARKOV MODELS OF TRUST IN DYNAMIC ENVIRONMENTS

Marie Elisabeth Gaup Moe, Bjarne E. Helvik and Svein J. Knapskog
Centre for Quantifiable Quality of Service in Communication Systems
Norwegian University of Science and Technology
O.S. Bragstads plass 2E, N-7491 Trondheim, Norway
{marieeli,bjarne,knapskog}@q2s.ntnu.no

Abstract Computational trust and reputation models are used to aid the decision-making process in complex dynamic environments, where we are unable to obtain perfect information about the interaction partners. In this paper we present a comparison of our proposed hidden Markov trust model to the Beta reputation system. The hidden Markov trust model takes the time between observations into account, it also distinguishes between system states and uses methods previously applied to intrusion detection for the prediction of which state an agent is in. We show that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents, due to its larger richness in model features. This means that our trust model may be more realistic in dynamic environments. However, the increased model complexity also leads to bigger challenges in estimating parameter values for the model. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Beta reputation system.

1. Introduction

Trust is a fundamental part of social and commercial relationships, both in the real-life and the virtual world. Complex dynamic environments, like the Internet, makes it extremely hard to obtain perfect information about potential interaction partners. In e-commerce and other electronic transactions and services, where the assets of interaction partners might be at risk, trust mechanisms may facilitate the decision-making process and lower the risk. Since trust management can be assumed to decrease risk, it can also be assumed that it will increase security and can be considered as a *soft security* mechanism [RJ96]. Soft security accepts the fact that it is possible to circumvent the implemented security mechanisms, given enough time, effort and money. Since we might have users with malicious intentions in a system, the challenge is to detect them and find a way to monitor their behavior and possibly in-

fluence their actions, in order to prevent them from causing any harm. Trust management serves this purpose by evaluating the trustworthiness of users and offering different service levels to users based on a trust policy. If services are denied to untrusted users, an incentive for users not to misbehave, is created.

Computational trust and reputation models seek to quantify trust as a value derived from previous direct experiences and/or second-hand information, such as recommendations, and suggest mathematical and logical expressions for how to combine several opinions about trustworthiness into reputation values. Such models are clearly needed in the virtual world where non-human agents are making trust-based decisions. But also when the human end-user is making the decisions, such calculated trust values can be very useful as decision support. For this reason a number of different trust models have been proposed. The modeling complexity varies, ranging from very simple eBay-like models to more sophisticated models based on probability theory, e.g. the Bayesian trust and reputation models [MMH02, JI02, BLB04, NKS07, JH07].

In this paper we will present a comparison of our previously proposed hidden Markov trust model [MTK08] to a binomial Bayesian reputation system [JI02]. The comparison is done with the help of simulations of trust scenarios. The objectives of this paper is to discuss probabilistic measurement of trust, outline the models and compare their performance in a dynamic environment where the (un)trusted objects may change behavior. We show that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents, this means that our trust model may be more realistic for dynamic environments. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Bayesian reputation system.

The remainder of this paper is organized as follows. Section 2 discusses the challenges related to modeling dynamic trust and how the different models can be evaluated. Section 3 gives a brief introduction to Bayesian trust models, in particular the Beta reputation system, Section 4 discusses the hidden Markov trust model, the simulation results are presented in Section 5, and Section 6 discusses the simulation results and concludes the paper.

2. The Dynamic Trust Modeling Problem

Since trust and reputation are active fields of ongoing research, numerous different models for quantification and evaluation of trust have been proposed. A review on some of these computational trust models can be found in [SS05]. However, there seems to be no single agreed upon model that can be used for benchmarking and comparison of the different trust and reputation algorithms.

Reputation models used for electronic commerce are often based on very simple mathematical formulas for combining opinions. One example is the reputation system implemented in eBay, where a feedback score is calculated as a sum of ratings that can be either positive, corresponding to a value of $+1$, negative with a value -1, or neutral with 0 value. A survey of trust and

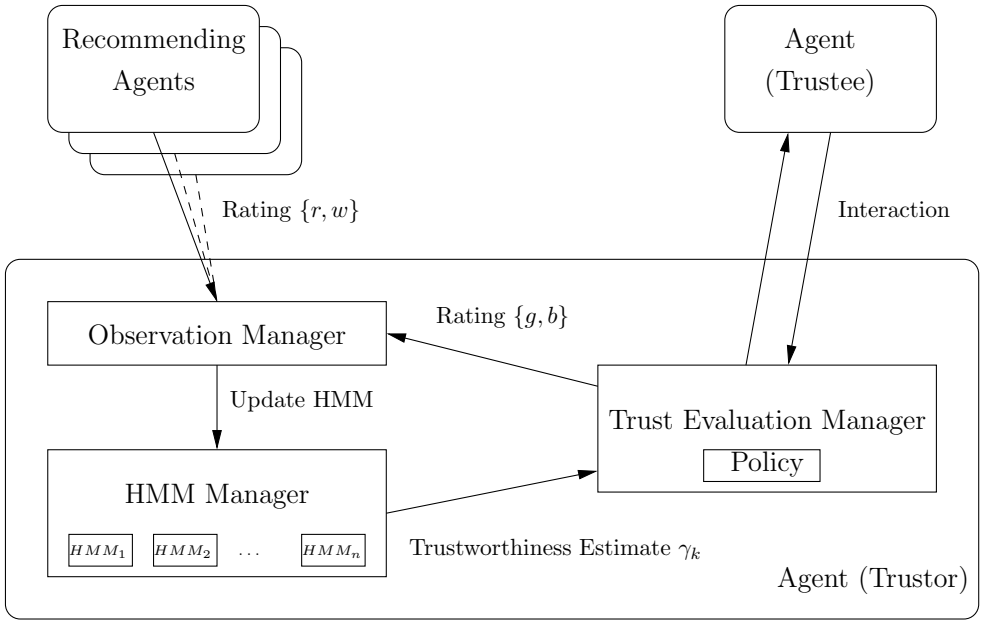


Figure 1. The architecture of a reputation system using hidden Markov trust modeling

reputation systems that are currently used in online services can be found in [JIB07].

In [DFM01], several desirable qualities of reputation systems are listed. According to the authors a reputation system should be efficient, robust against attacks, easily understandable and verifiable. It should also be *weighted toward current behavior*, meaning that it responds quickly to changes in behavior so that an entity which has performed well consistently over a long time but then suddenly changes its behavior will be detected and maybe no longer trusted. This feature is missing in many trust and reputation models as trust is modeled as a static property, not taking the time component into consideration. For some applications of reputation and trust the time dependency and response to dynamic behavior are very important, as the behavior of agents could be assumed to be highly dynamic. One example of such an application is when trust metrics are used in ad hoc routing protocols to counter malicious nodes, see for instance [MGLB00, DDB04, BLB04, BVTL07]. The common approach is that every node in the network monitors its neighbors and measures the frequency of packet dropping, misrouting and other potentially malicious behavior, and keeps a trustworthiness rating or reputation value recorded for all other nodes based on these observations. The underlying routing protocol is then modified with a trust component which selects routing paths and makes routing decisions based on the reputation values.

The computational trust algorithm used to calculate the reputation value varies. In [DDB04], a simple eBay-like scheme is used, where reputation is a sum of recommendations of +1 whenever a packet reaches its destination, and −1 if a packet is dropped, a node is considered untrusted if its reputation value falls below a certain threshold. In [BLB04] a more sophisticated scheme, based on a Bayesian reputation system, is used. A trust-based ad hoc routing protocol based on hidden Markov modeling of trust was proposed by the authors in [MHK08]. The architecture of the trust component used in this approach, presented as a more general decentralized reputation system, is illustrated in Figure 1. Every agent in the system keeps and updates hidden Markov models (HMMs), that are modeling the trust state of all the other agents. Before an agent (trustor) initiates an interaction with another agent (trustee), it looks up the trustworthiness value derived from the HMM belonging to the trustee. The trustor then decides according to a policy whether or not to interact with the trustee. After an interaction, the HMM belonging to the trustee is updated with a rating, good g, or bad b, based on the outcome of the interaction. The HMMs are also updated with observations in the form of ratings from other agents in the system, so called second-hand opinions, which are either in the form of recommendations r, or warnings w. In this study we do not include trust transitivity between different contexts, for the simplicity of the comparison, so we assume that the HMMs are only updated with observations related to the same context. We also do not consider chains of recommendations.

With this paper we would like to compare our hidden Markov modeling of trust to the Bayesian approach, in particular with regard to performance of the modeling of the dynamic aspect of trust, since this is a very important feature for applications in dynamic networking environments. A quantitative approach to comparing trust models can be found in [NKS07]. In this paper it is proposed to use the information theoretical measure *relative entropy*. However, this approach is only applicable to probabilistic trust models that share the same fundamental assumptions about the underlying probability distributions. In cases where a direct mathematical comparison of models is difficult, comparison by the help of simulations seems to be the most viable approach. Different trust scenarios can be simulated in order to see which trust and reputation system performs best with regard to reliability of the calculated values under various hostile agent strategies.

3. Bayesian Trust Modeling

Bayesian trust models, for calculating reputation scores from ratings, are based on the assumption that the behavior of an agent can be described according to a probability distribution. The trust value is a function of the expected value of the probability distribution, which gets updated with every new rating received according to Bayes' Theorem. Binomial Bayesian reputation systems, where ratings can be expressed by two values, *good* or *bad*, are modeled with

the Beta probability density function [MMH02, JI02, BLB04]. Multinomial Bayesian reputation systems, that allow for ratings with graded levels, are modeled with the Dirichlet probability density function [NKS07, JH07]. In this paper we will focus on the binomial case, and evaluate the performance of our proposed trust model compared to the Beta reputation system proposed by Jøsang et al. [JI02].

3.1 The Beta Reputation System

The Beta reputation system models the reputation formation for a trustor as a sequence of observations, where each observation is the outcome of the rating done by a trustee, based on the outcome of an interaction. A reputation centre collects ratings from all the agents, and updates each agent's reputation score.

The underlying mathematical model of the Beta reputation system considers the ratings as a sequence of trials with binomial outcomes, for each trial there is a probability p of getting a *good* rating (recommendation) and a probability $(1-p)$ of getting a *bad* rating (warning). The parameter p belonging to a trustor is initially unknown, so due to lack of information it is assumed that it is drawn from a uniform distribution on $[0,1]$. As ratings concerning this trustor start to arrive, there is more information available and we can update the distribution of p. In accordance with Bayesian inference we have a *prior* hypothesis X about the outcome of a trial, which is updated *a posteriori* to the actual outcome Y in accordance with Bayes' Theorem

$$P(X \mid Y) = \frac{P(X)P(Y \mid X)}{P(Y)}. \tag{1}$$

The Beta distribution

$$Beta(\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1} \tag{2}$$

is a *conjugate prior* for binomial trials (Bernoulli process). This means that if we assume that the prior X hypothesis is described by $Beta(\alpha, \beta)$, and Y is a sequence of ratings, out of which r is the number of good ratings (recommendations) and w is the number of bad ratings (warnings), then the posterior $P(X \mid Y)$ is also described by a Beta distribution $Beta(\alpha+r, \beta+w)$. The initial prior is given by $Beta(1,1)$, which corresponds to the uniform distribution on $[0,1]$. The reputation value is given as a function of the expectation value of the Beta distribution $E(p) = \alpha/(\alpha+\beta)$, for the posterior hypothesis the expectation is found by setting $\alpha = r+1$ and $\beta = w+1$. This results in a very simple calculation of the probability expectation value. Let (r_k, w_k) denote the ratings received at iteration step k, we then get the following recursion for

deriving the Beta parameters:

$$\alpha_k = \alpha_{k-1} + r_k, \quad \beta_k = \beta_{k-1} + w_k, \quad \alpha_0 = \beta_0 = 1. \tag{3}$$

For finding the probability expectation value at iteration step k we get:

$$E(p_k) = \frac{\alpha_k}{\alpha_k + \beta_k}. \tag{4}$$

The probability expectation value given in Equation 4 gives a reputation rating in the range $[0, 1]$, where the value 0.5 represents a neutral reputation value. To make the reputation model more realistic, several modifications to the calculation of the reputation value are introduced. These variations include *discounting* of ratings based on the reputation of the agent providing the rating, *forgetting* old ratings by giving old ratings less weight than more recent ratings, and *weighting* of ratings according to the value of the rated transaction.

3.2 Evaluation of the Beta Model

The Beta reputation system without forgetting factor is efficient, easily understandable and verifiable, but it is not weighted toward current behavior. This is due to the underlying Bayesian framework, which assumes that the behavior of agents can be approximated by a *fixed* probability distribution. Since agents may change behavior over time, this static modeling is not realistic. The forgetting factor $0 \leq \phi \leq 1$ was introduced in [JI02] to overcome this problem. It is used to scale the parameters (α, β) in every update of the Beta distribution, so that the we get

$$\alpha_k^* = \alpha_{k-1}^* \phi + r_k, \quad \beta_k^* = \beta_{k-1}^* \phi + w_k, \quad \alpha_0^* = \beta_0^* = 1. \tag{5}$$

A forgetting factor $\phi = 1$ means that all ratings are weighted equally, and nothing is forgotten, with $\phi = 0$ only the last rating is remembered. In Figure 2 we can see how the Beta model responds to a sequence with 20 good ratings followed by 20 bad ratings, with different forgetting factors.

As noted by the authors of [NKS07], the forgetting factor is a form of exponential decay on the parameters of the Beta model giving an effective bias towards newer information, but it is unclear if this fading mechanism is really modeling dynamic behavior of the agents. If agents were likely to change their behavior in such a way that the probability p of getting good ratings slowly increases or decreases, this fading of parameters seems like a good modeling approach. However, if we consider a disruptive agent that follows a strategy where it behaves good for a certain amount of time, building up a good reputation value, and then suddenly starts to misbehave taking advantage of its reputation, this slowly adapting model might not be good enough.

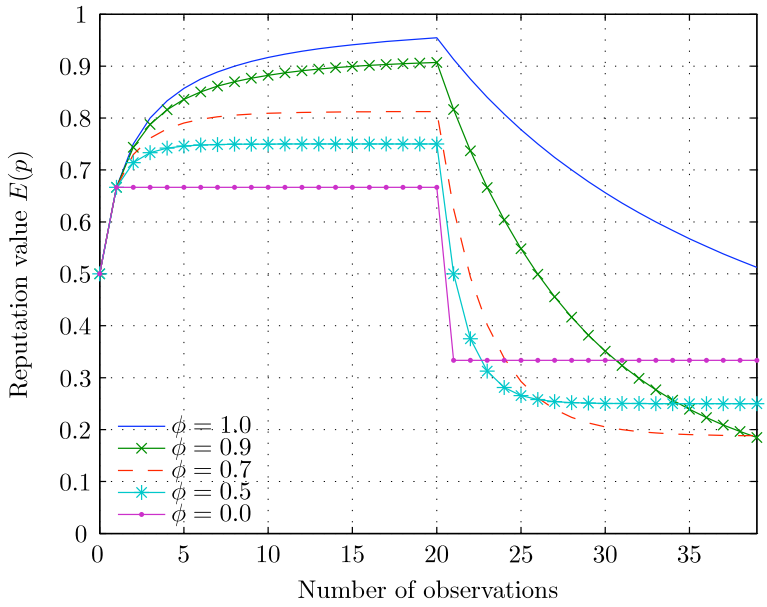


Figure 2. The Beta model with different forgetting factors, the observations are 20 good ratings followed by 20 bad ratings.

Another problem with the Beta model is the lack of time component. The reputation formation is only depending on the number of ratings, without taking the time between ratings into account. If we assume that ratings are not received at regular intervals, the claim that the forgetting factor takes care of adjusting the model towards *new* information may not be valid anymore. A simple way of rectifying this is by introducing a time stamp on the ratings, like suggested in [WJI04].

4. The Hidden Markov Trust Model

The hidden Markov trust model takes the time between observations into account, it also distinguishes between system *states* and uses methods previously applied to intrusion prevention [HMK08] for the prediction of which state an agent is in. The hidden Markov trust model was originally proposed by the authors as a component in a trust-based ad hoc routing protocol [MHK08]. It was further developed with a parameter learning component for multiagent environments [MTK08].

4.1 Hidden Markov Modeling

A hidden Markov model (HMM) consists of a finite set of N hidden states $S = \{s_1, \ldots, s_N\}$ with an associated probability distribution. The state of the

monitored agent is described by a discrete time Markov chain $\boldsymbol{x}_k = x_1, x_2, \ldots$ where $x_k \in S$ is the possibly hidden state of the agent at sampling instant k. $\boldsymbol{P}_k = \{p_{ij}^k\}$ is the set of state transition probabilities, $p_{ij}^k = P(x_{k+1} = s_j \mid x_k = s_i)$, $1 \leq i, j \leq N$, where x_k is the current state of the system. $\pi = \{\pi_i\}$ is the initial state distribution, where $\pi_i = P(x_1 = s_i)$, $1 \leq i \leq N$. The output from the agent ratings is classified by the set of observation symbols $V = \{v_1, \ldots, v_M\}$. Let $\boldsymbol{y}_k = y_1, y_2 \ldots$ denote the sequence of observations, where $y_k \in V$ is the observation made at sampling instant k. The HMM consists of two stochastic processes; the hidden process $\boldsymbol{x}_k$, and the observable process $\boldsymbol{y}_k$ that depends on $\boldsymbol{x}_k$. The relation between $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ is described by the probability distribution matrix $\boldsymbol{B} = \{b_j(m)\}$, where $b_j(m) = P(y_k = v_m \mid x_k = s_j)$, for $1 \leq j \leq N$, $1 \leq m \leq M$. See for instance [Rab90] for a more extensive introduction to HMMs.

In the hidden Markov trust model considered in this paper, we choose to use two hidden states $\{$*trusted, untrusted*$\}$, and four observation symbols $\{g, b, r, w\}$, corresponding to the observations *good, bad, recommendation* and *warning*. The reason for choosing two states is to make it easier to compare the model with the binomial Bayesian model. A comparison of our model with more states and more observation symbols to a multinomial Bayesian model would be an interesting topic for our future work. An agent is in an untrusted state if it has been behaving in a malicious way in previous interactions, it is in a trusted state if it has shown good behavior. We model trust as a dynamic variable, changing with time. This allows us to capture the behavioral characteristics of agents that are behaving good for a certain time, but then suddenly start misbehaving. Since an agent's behavior can be changing with time it is not necessarily the case that an agent is in the same state as it were at the last encounter. An agent can only do its best guessing about the trustworthiness state of an other agent based on its own previous direct experiences, which were either *good* or *bad*, and *recommendations* or *warnings* from other agents in the system. This means that the system state is hidden, and hence we use the HMM approach.

We consider a decentralized reputation system where each agent updates its own trust value for the other agents based on its own direct experiences, and from feedback in the form of ratings communicated from other agents in the multiagent system. We model the agent interaction as a stochastic process. This means that we assume that there is a random time interval between each agent interaction and that the behavior of an agent is only dependent on its current state. When using a Markov model to model the state of an agent, we make the following assumptions; all information about the agent is contained in the state, observations are independent given the current state, and state occupation time is negatively exponentially distributed.

4.2 State Probability Distribution

From the HMM we can derive a prediction of the probability distribution over the states, and we use the probability of being in the trusted state as reputation value. Our modeling approach is different from the Beta model as we do not assume that there is an underlying fixed probability p of getting a good rating. Instead we assume that an agent is in one of the hidden states, and that the ratings are characterized by different values of p dependent on the current state of an agent. The rating process is similar to the monitoring process in an intrusion detection system, and the challenge is to predict the current state of an agent and detect a possible state change.

We have not made any assumptions about time between observations, and there is no direct relation between observations and state-changes. As a consequence the system could have made zero, one or more transitions during the time between to successive observations. The time when observation number k is produced is denoted t_k. Time between observation $k-1$ and observation k is denoted $\delta_k = t_k - t_{k-1}$.

The transition rate matrix $\Lambda = (\lambda_{ij})$ is describing the dynamics of the system. To simplify the notation we will use i and j instead of s_i and s_j. The relation between system states and the transition rates is given by

$$\lambda_{ij} = \begin{cases} \lim_{dt \to 0} \dfrac{P(\boldsymbol{x}(t+dt) = j | \boldsymbol{x}(t) = i)}{dt} & \text{if } i \neq j \\ \sum_{j \neq i, j=1}^{N} -\lambda_{ij} & \text{if } i = j \end{cases}. \tag{6}$$

Since observations are received at irregular intervals, the running transition probabilities $p_{ij}^k = P(x(t+\delta_k) = j | x(t) = i)$ depend on the time since last observation δ_k, and have to be calculated each time an observation is received. The running transition probability matrix $\boldsymbol{P}_k = (p_{ij}^k)$ can be derived from Kolmogorov's equations [Ros03] as follows

$$\boldsymbol{P}_k = e^{\Lambda \delta_k}. \tag{7}$$

There are several analytical and numerical methods for solving these ordinary differential equations, in our case the state space is very small, so calculations are inexpensive. Let $\gamma_k = (\gamma_k(i))$ denote the prediction of the state probability distribution at time t_k given all observations received until time t_k, $\gamma_k(i) = P(x_k = i | \boldsymbol{y}_k)$ where $\boldsymbol{y}_k = y_1, \ldots, y_k$. The algorithm for calculating γ_k is given in [HMK08], it is an on-line algorithm derived from the *forward-backward* procedure described in [Rab90], and is very efficient. It does not require the agents to keep any history of past observations in memory.

4.3 Parameter Estimation

The parameters that need to be set in the HMM are the initial state distribution π, the observation symbol probabilities $\boldsymbol{B}$ and the state transition rates

Λ. In [MTK08] we describe a method for learning the model parameters by the combination of the machine learning technique *reinforcement learning* [SB98] and the forward-backward procedure, which finds the maximum likelihood parameter estimate from a training sequence of observations. As this parameter learning technique is not the main focus of this paper, we will assume that these parameters are available to the model and perform the simulations with a few different representative values for the parameters.

We will set the initial state distribution π to be uniform over the states, so we have that $\pi_1 = 0.5$ and $\pi_2 = 0.5$, in order to get the same starting condition as the Beta model. But to overcome the problem of agents changing their identities and re-entering the system frequently, it might be better to change the starting condition so that a newcomer to the system is most likely not in a trusted state.

The state transition rates can be calculated from estimated expected state sojourn times $H = (h_1, h_2)$, the relation between transition probabilities and transition rates is given by

$$\lambda_{ij} = \frac{p_{ij}}{h_i} \quad \text{for } i \neq j. \tag{8}$$

The transition rate models the tendency of the agent to change its trustworthiness over time, large state transition rates will lead to faster response to indications of state changes in the model.

The observation symbol probabilities models the uncertainty of the observations. If we for instance have the parameter $b_1(g) = 0.9$, this means that we have a probability of 0.1 of getting a good observation even though the agent really is in an untrusted state. In other words we have a certainty of 90% of getting correct observations. Figure 3 shows how the hidden Markov trust model responds to an input of 20 good followed by 20 bad observations for different observation symbol probabilities. The time between observations is fixed, and we have used the estimated state sojourn times $h_1 = 100$, and $h_2 = 100$. We have used symmetric observation probabilities in this example, i.e. if $b_1(g) = 0.9$ we also have that $b_2(b) = 0.9$.

As we can see from Figure 3, the observation symbol probabilities influence the response to state transitions in the model. This is natural, since if the observations are unreliable, we would like to have more observations indicating a state change before we believe that an actual state change has occurred. It would make sense to assign a higher observation symbol probability to the first-hand observations $\{g, b\}$ than to the second-hand observations $\{r, w\}$. For the second-hand observations we could choose to model each recommender separately, this means that we assign different observation symbol probabilities $\{r, w\}$ to every recommender. If we have a history of previous recommendations and warnings coming from a specific recommender, we can learn the parameters from this sequence of observations.

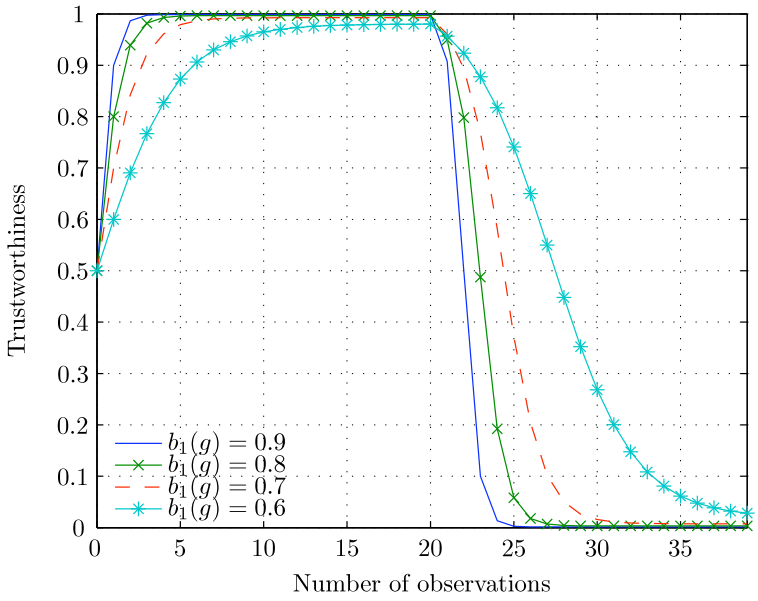


Figure 3. The hidden Markov trust model with different observation probabilities, the input is 20 good direct observations followed by 20 bad direct observations.

5. Simulation Results

In this section we will present some simulation results from our comparison study of the hidden Markov trust model and the Beta reputation model. We describe a selection of trust scenarios and compare the performance of the models in these situations.

5.1 Simulation Assumptions and Parameters

When we do the comparison of the Beta model and the hidden Markov trust model in the following, we will consider a decentralized version of the Beta reputation system, where we let each agent calculate its own reputation value for the other agents instead of calculating the reputation values in a reputation centre. We assume that there is a trusted reliable communication protocol in place that allows agents to obtain feedback from other agents in the form of ratings.

For the model parameters, we have used the Beta model with a forgetting factor $\phi = 0.9$, and the hidden Markov trust model with state sojourn times $h_1 = 100$, $h_2 = 100$ and observation symbol probabilities $b_1(g) = 0.8$, $b_2(b) = 0.8$. For the Beta model, we can see from Figure 2 that a high value of ϕ gives the best response to state changes as it gives the largest variation in the reputation value. Small values of ϕ seems to give quicker response, but

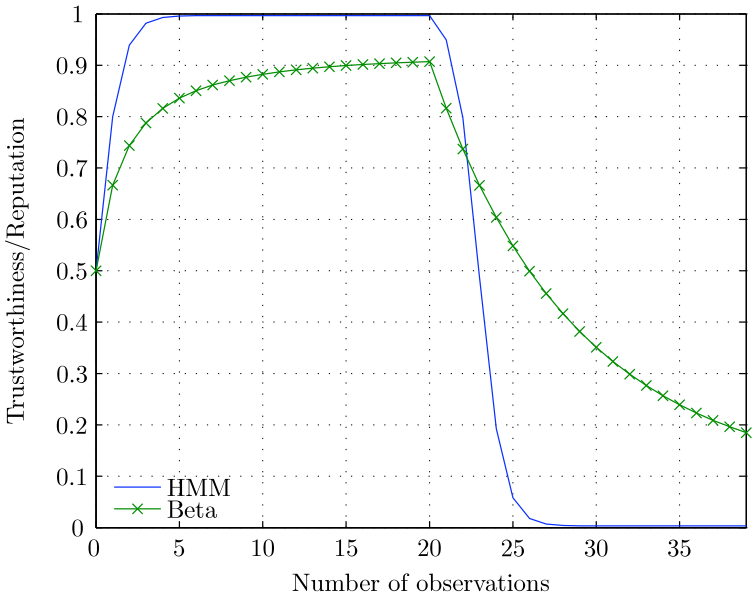


Figure 4. The hidden Markov trust model compared to the Beta model, the input is 20 good observations followed by 20 bad observations.

leads to a convergence of the reputation score to a less extreme value. This means that the reputation value becomes more average and does not clearly distinguish between states. Since we want to model these state changes with our hidden Markov trust model, we have used the Beta model with a high value of ϕ. For the hidden Markov trust model, we have used relatively high observation probabilities following the same reasoning. In the last simulation we have used other parameters for the hidden Markov trust model, because we want to illustrate the flexibility of the model by showing how we can adjust the parameters so that it responds similarly to the Beta model.

5.2 Response to State Changes

We have already seen from Figures 2 and 3 how the two models respond to a state change, we have 20 good observations followed by 20 bad observations. In Figure 4 we see the difference between the models more clearly. Such an input set of observations could come from a trust scenario where an agent builds its reputation value by behaving good for a certain amount of time, and then decides to take advantage of its good reputation by suddenly changing its behavior. From Figure 4 we see that the slope of our model is much steeper than the slope of the Beta model. The Beta model has a lower reputation value for the first observation, but this is due to the slower convergence of the Beta model to the good state. If we for instance had a threshold for detecting state

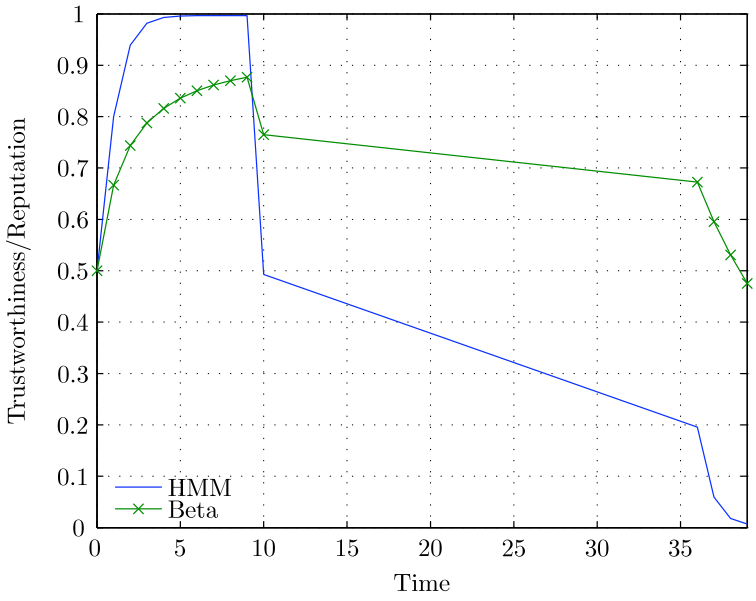


Figure 5. The hidden Markov trust model compared to the Beta model, the input is 9 good observations, 1 bad observation at time $t = 10$, then no observations until time $t = 35$ followed by 5 bad observations.

changes at the reputation value 0.5, the hidden Markov trust model would detect this already at the third bad observation while the Beta model would detect it after six bad observations.

5.3 Time Component

The Beta model does not take the time component into consideration, it only models the reputation value in terms of number of ratings. In the hidden Markov trust model we include the time between observations in our model. To illustrate the advantage of including the time aspect, we consider the following scenario. We assume that an agent has been compromised, i.e. 'taken over' by a malicious agent. The agent then proceeds with a strategy of 'laying low', meaning that is waits for a long time without acting malicious, so that when it starts to show malicious behavior it can take full advantage of the good reputation that the previous owner of the agent had built up. From Figure 5 we can see an example of such a scenario, where we have 9 good observations, then one bad observation at time $t = 10$, then no observations until time $t = 35$, followed by 5 bad observations. We can observe from the plot that the hidden Markov trust model gives a steeper slope and continues the negative trend over time, while the Beta model is just stretched at the x-axis.

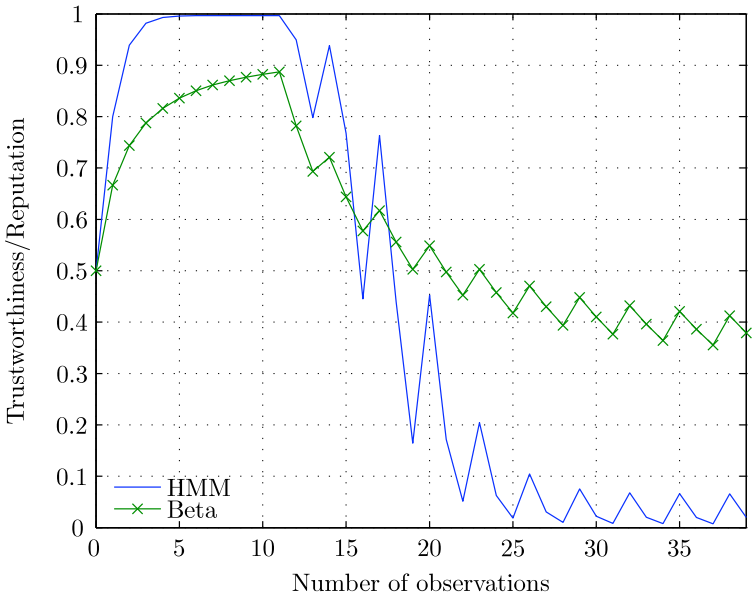


Figure 6. The hidden Markov trust model compared to the Beta model, the input is 10 good observations, followed by a disruptive behavior giving a pattern of 2 bad observations, one good observation, 2 bad observations, and so on.

5.4 Disruptive Behavior

We want to see how the models react to a disruptive agent that changes it strategy in order to adapt to the rules of threshold-based intrusion detection. In particular, we consider an agent that follows a pattern of misbehavior adapted to a detection rule of 'three strikes and you're out'. In Figure 6 we have an example of this scenario, where an agent is showing good behavior for 10 observations to build up its reputation, and then proceeds with the disruptive behavior giving a pattern of 2 bad observations, one good observation, 2 bad observations, and so on. We can see from the plot that the Beta model picks up this behavior with a decreasing reputation value, but the hidden Markov trust model detects the state change faster and converges to much lower trustworthiness values.

5.5 Model Flexibility

We have shown some examples where the hidden Markov trust model performs better than the Beta model in detecting state changes. This is not very surprising as the Beta model is not based on the assumption that agents can be in different *states* when it comes to trustworthiness. The performance of both models is of course dependent on the model parameters. The Beta model in the variant that we used in our simulations has fewer parameters than the

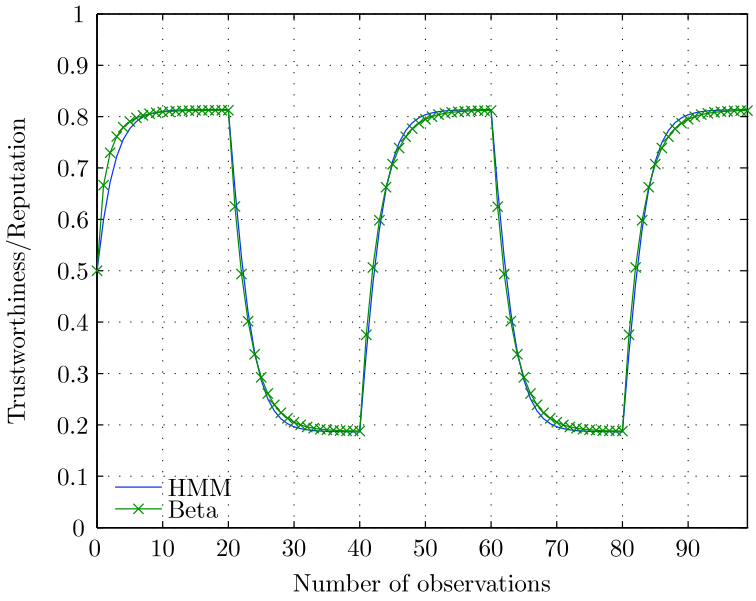


Figure 7. The hidden Markov trust model compared to the Beta model, the parameters of the models have been adjusted to make them respond similarly, input is 20 good observations followed by 20 bad observations, 20 good observations and so on.

hidden Markov trust model, albeit we have included the forgetting factor as a parameter in order to study the variant of the Beta model which is most sensitive to dynamic behavior. We used the parameters that seemed to give the most beneficial results for both models.

Now we want to illustrate the flexibility of the hidden Markov trust model, by adjusting its parameters so that it responds similarly to the Beta model. The results of this adjustment can be seen in Figure 7. We have used the Beta model with a forgetting factor of $\phi = 0.7$, and adjusted the parameters of our model to make it respond close to the Beta model. For the hidden Markov trust model we have used the observation symbol probabilities $b_1(g) = 0.6$, $b_1(b) = 0.4$, $b_2(b) = 0.6$ and $b_2(g) = 0.4$. We also adjusted the state sojourn times to $h_1 = 8$ and $h_2 = 8$. The hidden Markov trust model with these parameters describes a situation where observations are very uncertain and state transition rates are high. With such parameters we could say that the state modeling aspect of our model has been suppressed.

6. Discussion and Conclusion

We have seen from the simulated examples that the Beta model and the hidden Markov trust model performs differently. We will now explain the fundamental differences between the two models, and discuss the findings from

the simulations in this light. The hidden Markov model assumes an underlying state, observations are uncertain and we have an uncertainty of which state an agent is in. The Beta model does not assume that an agent is either good or bad, but rather seeks to pinpoint the trustworthiness of an agent on a continuous scale from 0 to 1. The interpretations of the observations in this model are deterministic. The difference between the models can be seen as an analogy to the difference between fuzzy sets and probabilities. In fuzzy logic an agent can be partially trusted, in the sense that he is 70% honest and 30% dishonest. This is different from a situation where we are 70% certain that an agent is 100% honest. This fundamental difference between the two models explains why the hidden Markov model performs better when it comes to the detection of changes in behavior of the agents over time. While the hidden Markov model recognizes a state transition, the Beta model is instead modeling an agent that gradually becomes partially more dishonest. This difference is clearly demonstrated in the simulation illustrated in Figure 6, where we consider an agent with a disruptive strategy. Additionally, we have the effect of the different time constants in the models. While the Beta model is relying on the 'lifetime' of old observations, the time constant in the hidden Markov trust model is associated with the underlying state transition process.

The hidden Markov trust model has more parameters then the Beta model, thus it can be more fine-tuned and adaptable to dynamic environments. However, this also leads to challenges related to the parameter estimation. In [MTK08] it is discussed how its parameters can be learned using a combination of the machine learning technique *reinforcement learning* [SB98] and the forward-backward procedure [Rab90], which finds the maximum likelihood parameter estimate. Both the Beta model and the hidden Markov trust model can be further refined by introducing more dimensions or states. The multinomial Bayesian models, which allow for graded ratings, introduce more dimensions to the Bayesian modeling. It would have been interesting comparing a multinomial Bayesian trust model to a hidden Markov trust model with more states and more observation symbols. However, such a comparison would be challenging due to the big number of parameters that would need to be managed in the simulations. Including trust transitivity between different contexts is also an important issue that should be addressed in future work.

We have presented a comparison of the hidden Markov trust model and the Beta reputation system. Due to its larger richness in model features, the hidden Markov trust model shows a better ability to deal with dynamic environments, where we are unable to obtain perfect information and agents can be assumed to change their behavior over time. However, the increased model complexity also leads to larger challenges in finding representative parameters for the model. A disadvantage of both models might be that they are not easily understandable to human users, since they build on much more advanced mathematics than the simple eBay-like systems. These models are therefore

maybe better suited for applications in multiagent systems, routing protocols and other distributed networking environments with non-human interpreters of the trustworthiness calculations.

References

[BLB04] S. Buchegger and J.Y. Le Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[BVTL07] Venkat Balakrishnan, Vijay Varadharajan, Uday Tupakula, and Phillip Lucs. Team: Trust enhanced security architecture for mobile ad-hoc networks. *ICON 2007: 15th IEEE International Conference on Networks*, pages 182–187, Nov. 2007.

[DDB04] Prashant Dewan, Partha Dasgupta, and Amiya Bhattacharya. On using reputations in ad hoc networks to counter malicious nodes. In *ICPADS '04: Proceedings of the Tenth International Conference on Parallel and Distributed Systems*. IEEE, 2004.

[DFM01] R. Dingledine, M. Freedman, and D. Molnar. Accountability. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly, 2001.

[HMK08] Kjetil Haslum, Marie Elisabeth Gaup Moe, and Svein Knapskog. Real-time Intrusion Prevention and Security Analysis of Networks using HMMs. In *Proceedings of the 4th IEEE LCN Workshop on Network Security (WNS 2008)*. IEEE, 2008.

[JH07] Audun Jøsang and Jochen Haller. Dirichlet Reputation Systems. In *Proceedings of the Second IEEE International Conference on Availability, Reliability and Security (ARES'07)*. IEEE, 2007.

[JI02] Audun Jøsang and Roslan Ismail. The Beta Reputation System. In *Proceedings of the 15th Bled Electronic Commerce Conference*, June 2002.

[JIB07] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

[MGLB00] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265, New York, NY, USA, 2000. ACM.

[MHK08] Marie Elisabeth Gaup Moe, Bjarne E. Helvik, and Svein J. Knapskog. TSR: Trust-based Secure MANET Routing using HMMs. In *Proceedings of the 4th ACM International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'08)*. ACM, 2008.

[MMH02] L. Mui, M. Mohtashemi, and A. Halberstadt. A Computational Model of Trust and Reputation. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pages 2431–2439, Jan. 2002.

[MTK08] Marie Elisabeth Gaup Moe, Mozhgan Tavakolifard, and Svein J. Knapskog. Learning Trust in Dynamic Multiagent Environments using HMMs. In *Proceedings of the 13th Nordic Workshop on Secure IT Systems (NordSec 2008)*, 2008.

[NKS07] Mogens Nielsen, Karl Krukow, and Vladimiro Sassone. A Bayesian Model for Event-based Trust. *Electronic Notes on Theoretical Computer Science (ENTCS)*, 172:499–521, 2007.

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.

[RJ96] Lars Rasmusson and Sverker Jansson. Simulated Social Control for Secure Internet Commerce. In *Proceedings of the 1996 Workshop on New Security Paradigms (NSPW'96)*. ACM, 1996.

[Ros03] Sheldon M. Ross. *Introduction to Probability Models*, chapter Continuous-Time Markov Chains, pages 349–390. Academic Press, New York, 8th edition, 2003.

[SB98] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

[SS05] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

[WJI04] Andrew Whitby, Audun Jøsang, and Jadwiga Indulska. Filtering out unfair ratings in bayesian reputation systems. In *In Proceedings of the 7th International Workshop on Trust in Agent Societies*, 2004.

Bibliography

[ABV06] Gergely Acs, Levente Buttyan, and Istvan Vajda. Provably secure on-demand source routing in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(11):1533–1546, 2006.

[ALRL04] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 01(1):11–33, 2004.

[ANK+06] Eric Alata, V Nicomette, M Kaâniche, Marc Dacier, and M Herrb. Lessons learned from the deployment of a high-interaction honeypot. In *EDCC'06, 6th European Dependable Computing Conference, October 18-20, 2006, Coimbra, Portugal*, Oct 2006.

[ÅSH+05] André Årnes, Karin Sallhammar, Kjetil Haslum, Tønnes Brekne, Marie Elisabeth Gaup Moe, and Svein Johan Knapskog. Real-time risk assessment with network sensors and intrusion detection systems. In *International Conference on Computational Intelligence and Security (CIS)*, volume LNAI 3802, pages 388–397, Dec 2005.

[ÅVVK06] André Årnes, Fredrik Valeur, Giovanni Vigna, and Richard A. Kemmerer. Using hidden markov models to evaluate the risk of intrusions. In *Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection, RAID 2006, Hamburg, Germany, September 20 – 22, 2006.*, September 2006.

[BB02] Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the confidant protocol. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236, New York, NY, USA, 2002. ACM.

[BEKXK04] Azzedine Boukerche, Khalil El-Khatib, Li Xu, and Larry Korba. SDAR: A Secure Distributed Anonymous Routing Protocol for Wireless and Mobile Ad Hoc Networks. In *29th Annual IEEE International Conference on Local Computer Network, LCN'04*, pages 618–624. IEEE, 2004.

[BGFI+98] J. S. Balasubramaniyan, J. O. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni. An architecture for intrusion detection using autonomous agents. In *Proceedings of the 14th Annual Computer Security Applications Conference*, page 13. IEEE Computer Society, 1998.

[BH01] L. Buttyán and J. Hubaux. Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks. Technical Report DSC/2001/001, Swiss Federal Institute of Technology, 2001.

[BLB04] S. Buchegger and J.Y. Le Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[BV04] Levente Buttyán and István Vajda. Towards Provable Security for Ad Hoc Routing Protocols. In *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 94–105, New York, NY, USA, 2004. ACM Press.

[BVTL07] Venkat Balakrishnan, Vijay Varadharajan, Uday Tupakula, and Phillip Lucs. Team: Trust enhanced security architecture for mobile ad-hoc networks. *ICON 2007: 15th IEEE International Conference on Networks*, pages 182–187, Nov. 2007.

[BVTM07] Venkat Balakrishnan, Vijay Varadharajan, Udaya Tupakula, and Marie Elisabeth Gaup Moe. Mitigating Flooding Attacks in Mobile Ad-hoc Networks Supporting Anonymous Communications. In *Proceedings of AusWireless 2007*. IEEE, 2007.

[Cha81] David L. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[Chr92] Lonnie Chrisman. Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 183–188. AAAI Press, 1992.

[CHSP00] Curtis A. Carver Jr., John M.D. Hill, John R. Surdu, and Udo W. Pooch. A methodology for using intelligent agents to provide automated intrusion response. In *Proceedings of the IEEE Workshop on Information Assurance and Security*, 2000.

[CL05] Jan Camenisch and Anna Lysyanskaya. A Formal Treatment of Onion Routing. In *Advances in Cryptology, Crypto 2005*, LNCS 3621, pages 169–187. Springer, 2005.

[CS06] Sebastian Clauß and Stefan Schiffner. Structuring anonymity metrics. In *DIM '06: Proceedings of the second ACM workshop on Digital identity management*, pages 55–62, New York, NY, USA, 2006. ACM.

[DCF05] H. Debar, D. Curry, and B. Feinstein. Intrusion detection message exchange format (IDMEF) – Internet-Draft, 2005.

[DCF07] H. Debar, D. Curry, and B. Feinstein. The intrusion detection message exchange format (IDMEF), 2007. IETF RFC 4765.

[DDB04] Prashant Dewan, Partha Dasgupta, and Amiya Bhattacharya. On using reputations in ad hoc networks to counter malicious nodes. In *ICPADS '04: Proceedings of the Tenth International Conference on Parallel and Distributed Systems*. IEEE, 2004.

[DFM01] R. Dingledine, M. Freedman, and D. Molnar. Accountability. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly, 2001.

[Dou02] John Douceur. The sybil attack. *Peer-to-Peer Systems*, LNCS 2429:251–260, 2002.

[DSCP02] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[DTBCC06] H. Debar, Y. Thomas, N. Boulahia-Cuppens, and F. Cuppens. Using contextual security policies for threat response. In *DIMVA '06: Proceedings of the 3th International Conference on Detection of Intrusions and Malware & Vulnerability Assessment, LNCS 4064*, pages 109–128. Springer-Verlag, 2006.

[DTD07] Claudia Diaz, Carmela Troncoso, and George Danezis. Does Additional Information Always Reduce Anonymity? In *WPES '07: Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 72–75, New York, NY, USA, 2007. ACM.

[DW01] Hervé Debar and Andreas Wespi. Aggregation and correlation of intrusion-detection alerts. In *RAID '00: Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection*, pages 85–103, London, UK, 2001. Springer-Verlag.

[FKM+05] Karen K. Fullam, Tomas B. Klos, Guillaume Muller, Jordi Sabater, Zvi Topol, K. Suzanne Barber, Jeffrey S. Rosenschein, and Laurent Vercouter. A demonstration of the agent reputation and trust (art): testbed for experimentation and competition. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 151–152, New York, NY, USA, 2005. ACM.

[Gam88] Diego Gambetta. Can we trust trust? In D. Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*. Basil Blackwell, 1988.

[GK04] Ashish Gehani and Gershon Kedem. Rheostat: Real-time risk management. In *Recent Advances in Intrusion Detection: 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004. Proceedings*, pages 296–314. Springer, 2004.

[GPWW+01] Katerina Goseva-Popstojanova, Feiyi Wang, Rong Wang, Fengmin Gong, K. Vaidyanathan, K. Trivedi, and B. Muthusamy. Characterizing intrusion tolerant systems using a state transition model. In *Proceedings of DARPA Information Survivability Conference and Exposition II, DISCEX '01.*, volume 2, pages 211–221, 2001.

[HAK07] Kjetil Haslum, Ajith Abraham, and Svein Knapskog. Dips: A framework for distributed intrusion prediction and prevention using hidden markov models and online fuzzy risk assessment. In *Third International Symposium on Information Assurance and Security, IEEE Computer Society press*, volume I, pages 183–188, 2007.

[HCD05] F. K. Hussain, E. Chang, and T. S. Dillon. Markov model for modeling and managing dynamic trust. In *Proceedings of the 3rd IEEE International Conference on Industrial Informatics, INDIN'05*, pages 725–733. IEEE, 2005.

[HCD06] F. K. Hussain, E. Chang, and T. S. Dillon. Quantification of the Effectiveness of the Markov Model for Trustworthiness Prediction. In *Proceedings of the International Conference 9th Fuzzy Days*. Springer, 2006.

[HJP02] Yih-Chun Hu, David B. Johnson, and Adrian Perrig. Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks. In *WMCSA '02: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, Washington, DC, USA, 2002. IEEE Computer Society.

[HMK08] Kjetil Haslum, Marie Elisabeth Gaup Moe, and Svein Knapskog. Real-time Intrusion Prevention and Security Analysis of Networks using HMMs. In *Proceedings of the 4th IEEE LCN Workshop on Network Security (WNS 2008)*. IEEE, 2008.

[HMN04] K. Hamamoto, K. Morooka, and H. Nagahashi. Motion Recognition By Combining HMM and Reinforcement Learning. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2004.

[HP04] Yih-Chun Hu and Adrian Perrig. A Survey of Secure Wireless Ad Hoc Routing. *IEEE Security & Privacy*, 2(3):28–39, 2004.

[HPJ02] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *MobiCom '02: Proceedings of the 8th annual international conference on Mobile computing and networking*, pages 12–23, New York, NY, USA, 2002. ACM Press.

[Hua06] Dijiang Huang. On Measuring Anonymity For Wireless Mobile Ad-hoc Networks. In *Proceedings of the 31st IEEE Conference on Local Computer Networks*, pages 779–786. IEEE, Nov. 2006.

[HWH+03] Guy Helmer, Johnny S. K. Wong, Vasant G. Honavar, Les Miller, and Yanxin Wang. Lightweight agents for intrusion detection. *J. Syst. Softw.*, 67(2):109–122, 2003.

[HYU02] M. A. T. Ho, Y. Yamada, and Y. Umetani. An HMM-based Temporal Difference Learning with Model-Updating Capability for Visual Tracking of Human Communicational Behaviors. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR'02)*. IEEE, 2002.

[JH07] Audun Jøsang and Jochen Haller. Dirichlet Reputation Systems. In *Proceedings of the Second IEEE International Conference on Availability, Reliability and Security (ARES'07)*. IEEE, 2007.

[JI02] Audun Jøsang and Roslan Ismail. The Beta Reputation System. In *Proceedings of the 15th Bled Electronic Commerce Conference*, June 2002.

[JIB07] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

[JMB01] David B. Johnson, David A. Maltz, and Josh Broch. *Ad hoc networking*, chapter DSR: the dynamic source routing protocol for multihop wireless ad hoc networks, pages 139–172. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[JO97] Erland Jonsson and Tomas Olovsson. A quantitative model of the security intrusion process based on attacker behavior. *IEEE Transactions on Software Engineering*, 23(4):235–245, 1997.

[Jon98] Erland Jonsson. An integrated framework for security and dependability. In *NSPW '98: Proceedings of the 1998 workshop on New security paradigms*, pages 22–29, New York, NY, USA, 1998. ACM.

[Jon06] Erland Jonsson. Towards an integrated conceptual model of security and dependability. In *ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security*, pages 646–653, Washington, DC, USA, 2006. IEEE Computer Society.

[JSL00] Erland Jonsson, Lars Strömberg, and Stefan Lindskog. On the functional relation between security and dependability impairments. In *NSPW '99: Proceedings of the 1999 workshop on New security paradigms*, pages 104–111, New York, NY, USA, 2000. ACM.

[Jul01] K. Julisch. Mining alarm clusters to improve alarm handling efficiency. In *ACSAC '01: Proceedings of the 17th Annual Computer Security Applications Conference*, page 12, Washington, DC, USA, 2001. IEEE Computer Society.

[KAN+06] M Kaâniche, E Alata, V Nicomette, Yves Deswarte, and Marc Dacier. Empirical analysis and statistical modeling of attack processes based on honeypots. In *WEEDS 2006 - Workshop on empirical evaluation of dependability and security (in conjunction with the international conference on dependable systems and networks, DSN 2006), June 25 - 28, 2006, Philadelphia,USA*, Jun 2006.

[KEB98] Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Proceedings of Information Hiding Workshop (IH 1998)*. Springer-Verlag, LNCS 1525, 1998.

[KH03] Jiejun Kong and Xiaoyan Hong. ANODR: ANonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks. In *ACM MobiHoc'03*. ACM, 2003.

[KL06] Rahul Khanna and Huaping Liu. System approach to intrusion detection using hidden markov model. In *IWCMC '06: Proceeding of the 2006 international conference on Communications and mobile computing*, pages 349–354, New York, NY, USA, 2006. ACM Press.

[KL07] Rahul Khanna and Huaping Liu. Distributed and control theoretic approach to intrusion detection. In *IWCMC '07: Proceedings of the 2007 international conference on Wireless communications and mobile computing*, pages 115–120, New York, NY, USA, 2007. ACM.

[KR90] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990.

[KR03] Syed A. Khayam and Hayder Radha. Markov-based Modeling of Wireless Local Area Networks. In *Proceedings of MSWiM'03*. ACM, 2003.

[LBF+93] B. Littlewood, S. Brocklehurst, N. Fenton, P. Mellor, S. Page, D. Wright, J. Dobson, J. McDermid, and D. Gollmann. Towards operational measures of computer security. *Journal of Computer Security*, 2:211–229, 1993.

[Mea95] Catherine Meadows. Applying the dependability paradigm to computer security. In *NSPW '95: Proceedings of the 1995 workshop on New security paradigms*, pages 75–79, Washington, DC, USA, 1995. IEEE Computer Society.

[MGLB00] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265, New York, NY, USA, 2000. ACM.

[MGPVT04] B. Madan, K. Goseva-Popstojanova, K. Vaidyanathan, and K. Trivedi. A method for modeling and quantifying the security attributes of intrusion tolerant systems. In *Performance Evaluation*, volume 56, 2004.

[MHK08] Marie Elisabeth Gaup Moe, Bjarne E. Helvik, and Svein J. Knapskog. TSR: Trust-based Secure MANET Routing using HMMs. In *Proceedings of the 4th ACM International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'08)*. ACM, 2008.

[MM02] Pietro Michiardi and Refik Molva. CORE: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *CMS'2002, Communication and Multimedia Security 2002 Conference, September 26-27, 2002, Portoroz, Slovenia*, 2002.

[MMH02] L. Mui, M. Mohtashemi, and A. Halberstadt. A Computational Model of Trust and Reputation. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pages 2431–2439, Jan. 2002.

[MTK08] Marie Elisabeth Gaup Moe, Mozhgan Tavakolifard, and Svein J. Knapskog. Learning Trust in Dynamic Multiagent Environments using HMMs. In *Proceedings of the 13th Nordic Workshop on Secure IT Systems (NordSec 2008)*, 2008.

[MVT02] B. Madan, K. Vaidyanathan, and K. Trivedi. Modeling and quantification of security attributes of software systems. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'02)*, 2002.

[NKS07] Mogens Nielsen, Karl Krukow, and Vladimiro Sassone. A Bayesian Model for Event-based Trust. *Electronic Notes on Theoretical Computer Science (ENTCS)*, 172:499–521, 2007.

[NST04] David M. Nicol, William H. Sanders, and Kishor S. Trivedi. Model-based evaluation: From dependability to security. *IEEE Transactions on Dependable and Secure Computing*, 01(1):48–65, 2004.

[NTH+06] J. Noda, M. Takahashi, I. Hosomi, H. Mouri, Y. Takata, and H. Seki. Integrating presence inference into trust management for ubiquitous systems. In *Proceedings of the eleventh ACM symposium on Access control models and technologies*, pages 59–68. ACM Press New York, 2006.

[OMSH03] Dirk Ourston, Sara Matzner, William Stump, and Bryan Hopkins. Applications of hidden markov models to detecting multi-stage network attacks. In *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS)*, 2003.

[PH02] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure Routing for Mobile Ad hoc Networks. In *SCS Communications Networks and Distributed Systems Modeling and Simulation Conference, CNDS 2002*, pages 193–204, 2002.

[PH05] Andreas Pfitzmann and Marit Hansen. Anonymity, Unobservability, and Pseudonymity: A Consolidated Proposal for Terminology. Draft, December 2005.

[PN97] P. A. Porras and P. G. Neumann. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In *Proc. 20th NIST-NCSC National Information Systems Security Conference*, pages 353–365, 1997.

[PTSC00] Adrian Perrig, J. D. Tygar, Dawn Song, and Ran Canetti. Efficient authentication and signing of multicast streams over lossy channels. In *SP '00: Proceedings of the 2000 IEEE Symposium on Security and Privacy*, Washington, DC, USA, 2000. IEEE Computer Society.

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.

[Rah00] Ali Rahimi. An Erratum for "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". http://alumni.media.mit.edu/~rahimi/rabiner/rabiner-errata/rabiner-errata.html, 2000.

[RHJ05] S.D. Ramchurn, D. Huynh, and N.R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(01):1–25, 2005.

[RJ96] Lars Rasmusson and Sverker Jansson. Simulated Social Control for Secure Internet Commerce. In *Proceedings of the 1996 Workshop on New Security Paradigms (NSPW'96)*. ACM, 1996.

[Ros03] Sheldon M. Ross. *Introduction to Probability Models*, chapter Continuous-Time Markov Chains, pages 349–390. Academic Press, New York, 8th edition, 2003.

[RR98] Michael Reiter and Aviel Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.

[Sal07] Karin Sallhammar. *Stochastic Models for Combined Security and Dependability Evaluation*. PhD thesis, Norwegian University of Science and Technology, 2007.

[SB98] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

[SBD+91] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur. DIDS (distributed intrusion detection system) - motivation, architecture, and an early prototype. In *Proceedings of the 14th National Computer Security Conference*, pages 167–176, Washington, DC, 1991.

[SCCC+96] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. GrIDS – A graph-based intrusion detection system for large networks. In *Proceedings of the 19th National Information Systems Security Conference*, 1996.

[SCS03] S. Singh, M. Cukier, and W.H. Sanders. Probabilistic validation of an intrusion-tolerant replication system. In de Bakker, J.W., de Roever, W.-P., and Rozenberg, G., editors, *International Conference on Dependable Systems and Networks (DSN'03)*, June 2003.

[SD02] Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[SDL+02] Kimaya Sanzgiri, Bridget Dahill, Brian Neil Levine, Clay Shields, and Elizabeth M. Belding-Royer. A secure routing protocol for ad hoc networks. In *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 78–89, Washington, DC, USA, 2002. IEEE Computer Society.

[Sey06] Stefaan Seys. *Cryptographic Algorithms and Protocols for Security and Privacy in Ad Hoc Networks*. PhD thesis, Katholieke Universiteit Leuven, 2006.

[Sha76] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[SHK07] Karin Sallhammar, Bjarne E. Helvik, and Svein J. Knapskog. A framework for predicting security and dependability measures in real-time. *International Journal of Computer Science and Network Security (IJCSNS)*, 7(3):169–183, 2007.

[SKN06] V. Sassone, K. Krukow, and M. Nielsen. Towards a Formal Framework for Computational Trust. In *Proceedings of the 5th International Symposium on Formal Methods for Components and Objects (FMCO 2006)*, volume LNCS 4709, page 175. Springer, 2006.

[SKS08] A. Srivastava, A. Kundu, and S. Sural. Credit card fraud detection using hidden markov model. *IEEE Transactions on Dependable and Secure Computing*, 5(1):37–48, 2008.

[SP06] Stefaan Seys and Bart Preneel. ARM: Anonymous Routing Protocol for Mobile Ad hoc Networks. In *Proceedings of the 20th IEEE International Conference on Advanced Information Networking and Applications - Workshops (AINA 2006)*. IEEE, 2006.

[SPX04] W. Song, V.V. Phoha, and X. Xu. The HMM-Based Model for Evaluating Recommender's Reputation. In *Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04)*, pages 209–215. IEEE, 2004.

[SS05] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

[Sta04] Standards Australia and Standards New Zealand. AS/NZS 4360: 2004 risk management, 2004.

[SW06] Vitaly Shmatikov and Ming-Hsiu Wang. Measuring Relationship Anonymity in Mix Networks. In *WPES '06: Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 59–62, New York, NY, USA, 2006. ACM.

[Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*, 10(5):557–570, 2002.

[THV04] Gergely Tóth, Zoltán Hornák, and Ferenc Vajda. Measuring Anonymity Revisited. In Sanna Liimatainen and Teemupekka Virtanen, editors, *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, Espoo, Finland, November 2004.

[VVKK04] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. A comprehensive approach to intrusion detection alert correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3):146–169, 2004.

[WB05] Xiaoxin Wu and E. Bertino. Achieving K-anonymity in mobile ad hoc networks. In *1st IEEE ICNP Workshop on Secure Network Protocols*, pages 37–42. IEEE, Nov. 2005.

[WD92] Christopher Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[WFP99] C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: Alternative data models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, 1999.

[WJI04] Andrew Whitby, Audun Jøsang, and Jadwiga Indulska. Filtering out unfair ratings in bayesian reputation systems. In *In Proceedings of the 7th International Workshop on Trust in Agent Societies*, 2004.

[WMT03] Dazhi Wang, Bharat B. Madan, and Kishor S. Trivedi. Security analysis of sitar intrusion tolerance system. In *SSRS '03: Proceedings of the 2003 ACM workshop on Survivable and self-regenerative systems*, pages 23–32, New York, NY, USA, 2003. ACM.

[YNK01] Seung Yi, Prasad Naldurg, and Robin Kravets. Security-aware ad hoc routing for wireless networks. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 299–302, New York, NY, USA, 2001. ACM.

[ZCY03] S. Zhong, J. Chen, and Y.R. Yang. Sprite: a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *INFOCOM 2003. Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1987–1997. IEEE, 2003.

[ZLL05] Yanchao Zhang, Wei Liu, and Wenjing Lou. Anonymous Communications in Mobile Ad Hoc Networks. In *IEEE INFOCOM 2005*. IEEE, 2005.

[ZWK+04] Bo Zhu, Zhiguo Wan, Mohan S. Kankanhalli, Feng Bao, and Robert H. Deng. Anonymous Secure Routing in Mobile Ad-Hoc Networks. In *29th Annual IEEE International Conference on Local Computer Network, LCN'04*, pages 102–108. IEEE, 2004.