Adversary Modelling[1]

Evaluating the Feasibility of a Symbolic Adversary Model on Smart Transport Ticketing Systems

Authors
Arthur Sheung Chi Chan, MSc (Royal Holloway, 2014)
Keith Mayes, ISG, Royal Holloway

Overview

Adversary modelling is a formal modelling method which aims to illustrate the abilities of an adversary in a system, and how the system behaves in the presence of an active adversary. Although there is much research on many modelling techniques, there has been relatively little work in smart transport ticketing systems where there is continuous updating of adversary knowledge. This article aims to dig a little deeper into adversary modelling by considering one of the process algebra languages known as Communicating Sequential Processes (CSP), which is used in formal modelling. In particular we seek possible extensions of the CSP language to support modelling of smart transport ticketing systems, taking into account the continuous updating of adversary knowledge.

Introduction

The security level of a system is always an interesting point to estimate and analyse in the field of Information Security. It can help to predict possible attacks by adversaries and to implement appropriate protection. In order to find a standard and comparative index for the security level within different systems and protocols, there are many researchers in the information security field using formal modelling frameworks to assist in the security level estimation of systems.

One of the formal modelling directions is known as *adversary modelling*, which aims to illustrate the abilities of an adversary in a system or network, and how the system behaves in the presence of the active adversary. The adversary modelling approach can for example, determine the potential for secret information leaking to an adversary which if exploited may create a security violation of the system. This approach can help to identify loopholes and warn the system developer to fix the security vulnerabilities in advance of launching a live product or system. In summary, by using a formal adversary modelling framework, we can more easily identify system security vulnerabilities at an early stage and take remedial action before they are exploited.

Although there is already much research on different formal modelling techniques, relatively little work has targeted smart transport ticketing systems and solutions that are concerned with continuous

[1] This article is to be published online by Computer Weekly as part of the 2015 Royal Holloway info security thesis series. The full MSc thesis is published on the ISG's website.

updating of the adversary knowledge. This article aims to dig a little deeper into adversary modelling by considering one of the process algebra languages known as Communicating Sequential Processes (CSP), which is used in formal modelling. In particular we seek possible extensions of the CSP language to support modelling of smart transport ticketing systems, with consideration to the continuous updating of adversary knowledge.

What is formal symbolic adversary modelling and why is it important?

In the world of information security, modelling is a technique and framework to help make protocols and systems easier to understand and analyse. With a better understanding, it is easier to identify potential problems and vulnerabilities and fix them in advance of incidents, and increase the security protection of the system. There are many modelling techniques, including adversary modelling, system environment modelling and process modelling, as illustrated in Figure 1.

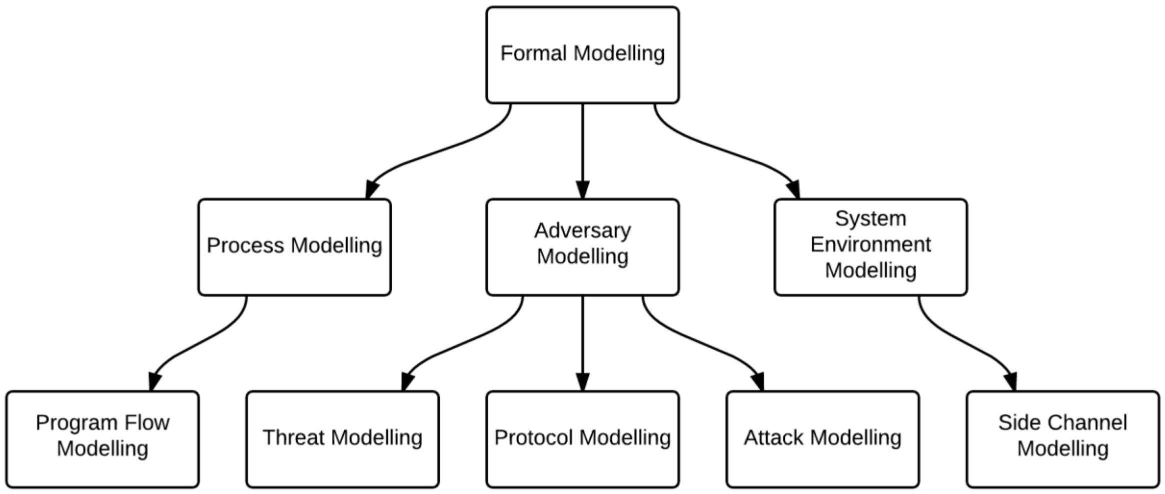

Figure 1 Types of Formal Modelling

Adversary modelling is a kind of modelling technique used to review the security level of systems, protocols or other logical settings within a set of assumptions and estimation of the computation power, initial knowledge and ability of the possible adversaries (can be active or passive attackers). It attracts interest because it can help to discover potential problems at the design stage and reduce risk well before a system goes live. There are many different branches in adversary modelling, including threat modelling, attack modelling and protocol modelling:

Assumptions about adversaries:

- Computation power
- Initial knowledge
- Ability

- *Threat modelling* aims to examine the goal of the potential adversary to minimize the effect of the adversary on the system.

- *Attack modelling* analyses not only the goal of the adversaries, but the attack paths of the adversaries. For example, an attacker may seek to escalate his system privilege level to mount an attack.
- *Protocol modelling* aims to analyse the message exchange in the communication channels, considering information leakage.

Symbolic adversary modelling: mathematical symbols to describe protocols and mathematical approach to analyse protocol.

Within the different kinds of adversary modelling techniques, *symbolic adversary modelling* is primarily aimed at protocol analysis. It makes use of some mathematical symbols and notation to define the communication and the settings of the protocol. It then uses mathematical approaches such as set theory to analyse the protocols. In some of the symbolic adversary models, a new set of protocol algebra language is defined to represent each of the communications and knowledge items in order to simplify the analysis process and the complexity of the modelling work. It can also help to provide proof that the simplified version is analogous to the full protocol. Some of the symbolic adversary models are very famous (like Dolev-Yao and CSP language) and some researchers have even developed automatic tools to assist in the modelling and analysis process.

What makes formal modelling important to Smart Transport Ticketing Systems?

Nowadays, smart cards have been adopted in many public transport ticketing systems. This trend makes the security and mutual authentication of the card and reader an important factor to consider. In the case of a smart transport ticketing system, the security level of the protocol and system use is a major concern and challenge as the transaction time of smart cards in transport ticketing systems is very short. Although there are many adversary modelling frameworks, there are very few that target the limited computational power of the smart card and the associated protocols and systems used in public transport ticketing. In fact, consideration of adversaries and exploitation of vulnerabilities appears to be handled in a rather ad-hoc and reactive manner within this field. In order to have a more scientific way to understand and study the security of smart transport ticketing system, the existing adversary model requires study. This study can help to see if it is feasible to model and analyse the protocol and system by formal modelling with the goal of ensuring a high level of security level in such systems.

Process Algebra CSP

Communicating **S**equential **P**rocesses: a type of symbolic adversary modelling language.

Process Algebra CSP is an example of a formal modelling framework; it is a kind of symbolic adversary modelling language, which was introduced by C. A. R. Hoare in his book "Communicating Sequential Processes" published in 1985. The concept has been extended by many groups of researchers to provide a more thorough definition and framework based on the language and to allow a full set of automated verifications for security levels and possible attacks on different kind of systems, protocols and network communication environments.

The analysis of the CSP model can be separated into two approaches, the model-checking approach and the model theorem-proving approach. They provide outcomes and illustrations of the target security level and vulnerabilities of the system under a predefined set of abilities and initial knowledge of an adversary. The model-checking approach makes use of some automated tools to complete the checking. The theorem-proving approach makes use of the Rank Function, introduced by Steve Schneider, to check if information is leaking in the communication channel, which increases the space of the adversary's knowledge.

Casper: Transform protocols into CSP, like a compiler.

One of the most important part in the CSP approach is to read the details of the protocol design and to write out the symbolic format of the message exchange and all actors' behaviour in the communication channel. In practice, this part will be very time consuming because many protocol are complex and include a lot of message exchange even after simplification. Furthermore, errors are easily made in the creation of a CSP model for a complicated protocol, and this of course will affect the correctness of the analysis. For example, a single specification error in the input of the Failures Divergences Refinement tool (FDR) (an automatic tool to check if there are security loopholes on the system modelled by CSP language) can lead to a false positive case. For this reason, Gavin Lowe developed a compiler-like software named Casper to help to transform easy to understand wording into the mathematical symbols used by CSP. This method is very similar to how a compiler for a high-level language like C or Java creates the low-level language of assembly and byte code. The compiler will generate a CSP model as output, which can be used in the rank function theorem-proving approach or act as an input to FDR for the model checking approach.

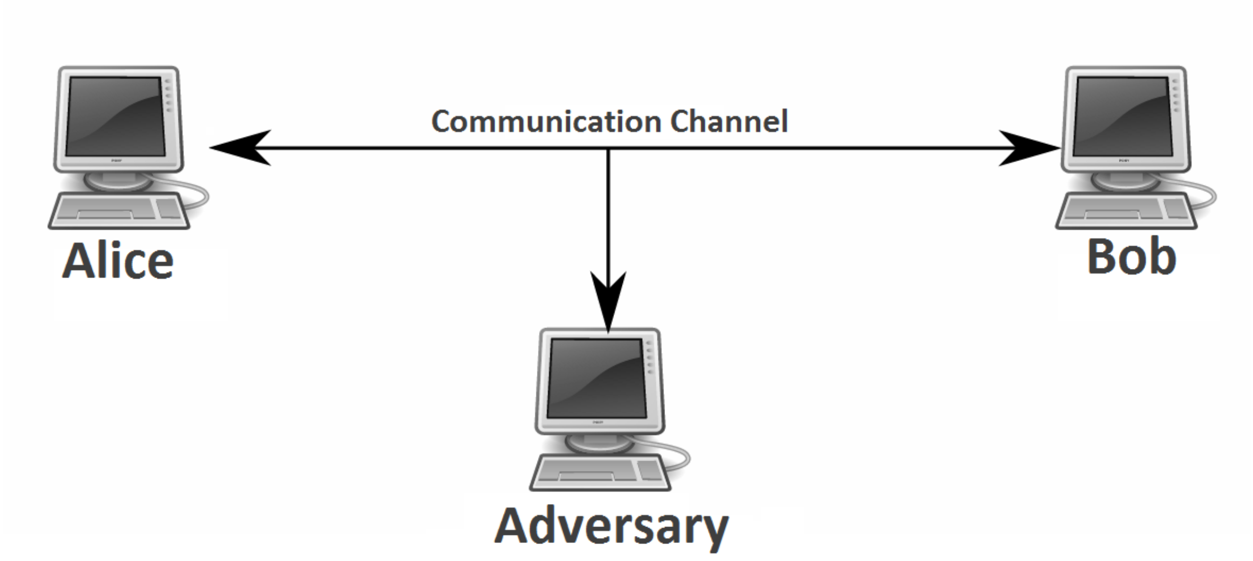

Figure 2 Conventional Adversary Access to Communications

Rank function is the most common theorem proving approach. It is used for each of the messages transmitted in the communication channel to find if there is potential for information leakage. It provides a general proof to determine if the CSP model and all its possible traces satisfy the target security level. The main idea is that each message exchanged and the initial knowledge of the opponents will be given a rank (an integer). In other words, the rank function provides a mapping between the facts, signals, messages and integers. The communicating parties will need to maintain positive rank in the communication. That is, if a party only receives a message with positive rank, then it can only send a message of positive rank, with the exception of the adversaries that are only allowed to send positive rank based on the initial knowledge and the entire observable message set. All secret values will carry a non-positive rank. If the communicating parties just send positive rank message in the channels, then the secrets will never appear on the channel so cannot increase the knowledge of the adversaries. The rank theorem proving aims to find a set of rank function that fulfil all the requirements of the rank function. If there exists a rank function for the protocol in all of the cases (maybe a considerable number), then the protocol is proven to be secure for the designed security features.

In general rank function proving, the term “security claim” appears in the communication sequence after some secret value is exchanged or proved. For example, referring to Figure 2, if Alice (A) received a message from Bob (B) and A is able to determine from the message that it now shares a secret value S with B, then A can issue a security claim that will be added to the CSP model. By using rank function proving, every user has to maintain positive rank. When any user violates this rule and broadcasts a secret value on the communication channel, the theorem proving will fail at that point and the process never have the chance to reach the issue of security claim. Thus, the leakage is notified and a security claim can only be announced in the case of a security requirement being met. In order words, a rank function can be used to check if a security claim is valid on the point in time when it is issued.

Additional consideration of the change in adversary knowledge set

Basic CSP modelling and theorem proving only consider the communication process and all the messages existing in the communication network environment; the encryption and cryptographic algorithms are ignored and treated as black boxes with no flaws. In reality, some flaws in protocols may arise from bad implementation or poor design of the cryptographic algorithm. For example, Figure 3 shows the MiFare Classic authentication protocol. Flaws in the cryptographic algorithms allow the adversary to deduce more than can be seen in the communication channels, and just a three-message exchange can let the adversary deduce the value of the shared secret k. This shows that additional consideration is needed to identify any problems and leakage.

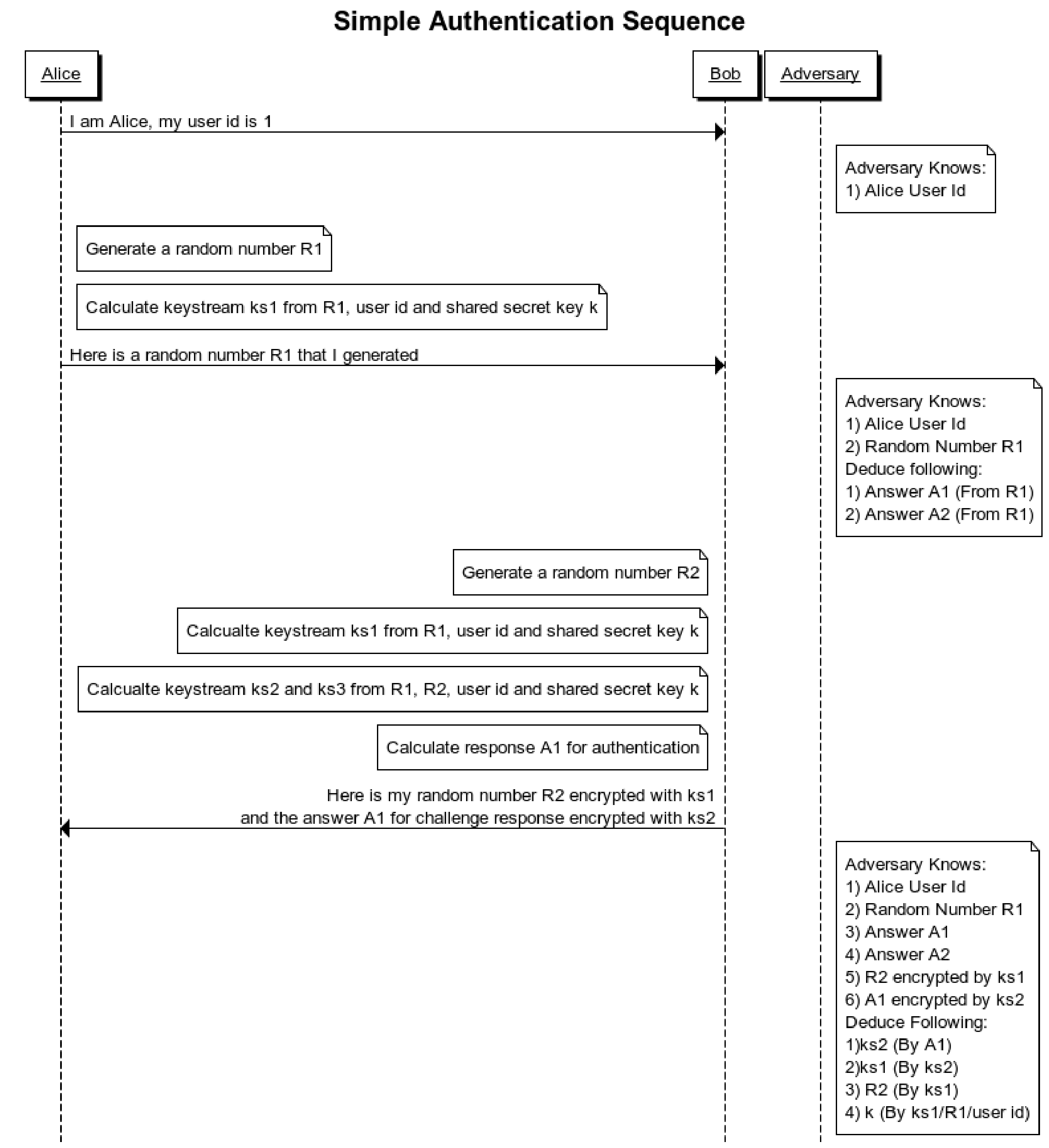
Simple Authentication Sequence
Alice
Bob
Adversary
I am Alice, my user id is 1
Adversary Knows:
1) Alice User Id
Generate a random number R1
Calculate keystream ks1 from R1, user id and shared secret key k
Here is a random number R1 that I generated
Adversary Knows:
1) Alice User Id
2) Random Number R1
Deduce following:
1) Answer A1 (From R1)
2) Answer A2 (From R1)
Generate a random number R2
Calcualte keystream ks1 from R1, user id and shared secret key k
Calcualte keystream ks2 and ks3 from R1, R2, user id and shared secret key k
Calculate response A1 for authentication
Here is my random number R2 encrypted with ks1
and the answer A1 for challenge response encrypted with ks2
Adversary Knows:
1) Alice User Id
2) Random Number R1
3) Answer A1
4) Answer A2
5) R2 encrypted by ks1
6) A1 encrypted by ks2
Deduce Following:
1)ks2 (By A1)
2)ks1 (By ks2)
3) R2 (By ks1)
4) k (By ks1/R1/user id)

Figure 3 Simple Authentication Sequence Example

Furthermore, it is possible for information to leak to adversaries via side-channels (e.g. from power or timing variations). It is also possible that some key elements needed to deduce the secret value are leaked separately throughout the communication channels. Therefore the knowledge deduction and wider dynamic abilities of the adversaries also require analysis, because they will directly affect the attackers' knowledge and thus affect the security proof of the protocol. We need to keep track of the change in the adversaries' knowledge set in order to understand if some secret value remains secret over time. For example, suppose the adversary needs value A, B and C in order to deduce the secret value S. If at first he only knows A then by definition he cannot know S. By observing the communication channels, over which two legitimate agents exchange values B and C, the adversary obtains all the knowledge necessary to deduce S. Cleary the adversary's knowledge set is dynamic and increasing over time. In order to model this kind of situation, the change in an adversary's knowledge set becomes a key factor to monitor throughout the modelling process, and is one of the points studied to extend the CSP modelling framework to use on smart transport ticketing systems.

With the new consideration of the indirect message generation and deduction, there exist great changes in the theorem proving steps. The main reason for the differences is that we now also need to keep track of the content and change of the adversary knowledge set because some indirect message generation will increase the size of the knowledge set. There may also be secondary effects as an increased knowledge set may possibly increase the ability of an adversary to extract more information or to fabricate more messages. By the rules of the rank function, a message that can be seen or created by an adversary using his knowledge set should be classified as positive rank, and protocol and message exchange should maintain the positive rank throughout the communication. So, the indirect message generation will need to be considered because it may affect the rank function. This setting becomes an additional part in the rank function theorem proving, and should help to increase the scope and effectiveness of the overall adversary modelling and protocol analysis.

Final Thoughts

Normally, in formal adversary modelling and protocol analysis, flaws in the cryptographic algorithm, partial value leakage and implementation side-channel leakage are ignored. By using an extended approach, we can add consideration of some known attacks on a cryptographic algorithm into the modelling and protocol analysis in order to make the formal modelling closer to the real world situation. This practical approach puts additional emphasis on the whole network environment and not just the communication and message exchange. It can help to analyse what the adversary can create or deduce from the knowledge set in addition to the information and data directly leaking out from the communication channel. This approach also has its limitations; it can only consider known attacks and flaws in the known cryptographic algorithms and some implementation, and so if all the algorithm details (and implementation vulnerabilities) are obscured or kept as business secrets, then this approach works no differently to the basic CSP modelling method. The extended approach and the new consideration of dynamic adversary knowledge is a promising way to model security protocols and system communications using published cryptographic algorithms, and can provide a more reliable result when system vulnerabilities and algorithm flaws are known in advance.

Biographies

Arthur Sheung Chi Chan serves as a programmer and software developer in a software warehouse in Hong Kong which provide business, web and mobile application solutions with security awareness to the clients. He completed his MSc in Information Security study in 2014 and passed the Master with a high distinction.

Keith Mayes B.Sc. Ph.D. CEng FIET A.Inst.ISP, has spent much of his life working in/with industry, yet is also an active researcher/author with 100+ publications. He is Director of the ISG Smart Card Centre at Royal Holloway University of London and of Crisp Telecom Limited. He has worked in hardware/software development, DSP and sensors, standardisation, mobile communications, smart cards/RFIDs, embedded systems, systems modelling plus diverse aspects of information security. Current interests include (but are not restricted to) mobile comms and trusted execution technologies, NFC, Smart cards/RFIDs, transport ticketing systems, automotive security, m-commerce, attacks and attack resistant system implementations.