Studies in Socio-Technical Security Analysis: Authentication of Identities with TLS Certificates

Ana Ferreira*†, Rosario Giustolisi*, Jean-Louis Huynen*†, Vincent Koenig*†, Gabriele Lenzini*
* Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg
† Educational Measurement and Applied Cognitive Science (EMACS), University of Luxembourg

***Abstract*—Authenticating web identities with TLS certificates is a typical problem whose security depends on both technical and human aspects, and that needs, to be fully grasped, a socio-technical analysis. We performed such an analysis, and in this paper we comment on the tools and methodology we found appropriate. We first analysed the interaction ceremonies between users and the most used browsers in the market. Then we looked at user's understanding of those interactions. Our tools and our methodology depend on whether the user model has a non-deterministic or a realistic behaviour. We successfully applied formal methods in the first case. In the second, we had to define a security framework consistent with research methods of experimental cognitive science.**

***Index Terms*—Socio-Technical Security, Ceremony Analysis, Human Computer Interaction**

I. Introduction

Let us assume you are meeting a person that is expected to do some job for you. You know nothing about him except a few things such as his name and affiliation. How can you trust a stranger introducing himself and claiming to be that person?

This "are you a friend or a foe?" problem boils down to assessing the validity of an identity. Plenty of solutions have been proposed to solve it; a few require sharing passwords beforehand (which is feasible only rarely) and the majority requiring a certain degree of trust. For example, trust is required when the stranger shows a document vouching his identity, because you need to trust the authority that has issued the document you are given. This problem of assessing someone's identity is exactly that which, on the Internet, a browser faces when it asks a server to authenticate itself. Here, the proof has the form of a Transport Layer Security (TLS) certificate, which, to be valid, should be ultimately signed by an authority that the browser recognises as trustworthy. Indeed, if all certificates were signed by world-wide known and unquestionably honest authorities the role of trust would be negligible: no trust is required when there is full knowledge and control of events. Such a situation is unrealistic for TLS-based authentication (e.g., see [1]), and therefore, someone, somewhere, has to decide whether to trust an entity to be honest or not. But users do not usually have the understanding and security knowledge to take secure decisions. In TLS certificate validation users can be asked to take such decisions, for instance, when a server presents a self-signed certificate.

A self-signed certificate is issued and signed by the server itself. Thus, to trust a self-signed certificate one should trust the server already, which is a useless circular reasoning. What solutions are available that help security with self-signed TLS? The protocol that is responsible for the validation of certificates does not give answers: for TLS, self-signed certificates are technically unverifiable and thus "*MUST either notify the user (clients MAY give the user the opportunity to continue with the connection in any case) or terminate the connections*" [2]. This recommendation has been picked up by browsers that implement it in various ways; but since so many Internet frauds still happen, a reliable standard solution seems yet to be found.

Frauds can happen when users are involved in security decisions as, these can be very complex to explain and understand. Preferably users should better not take such decisions, as suggested in [3], but for the problem of validating an identity on the Internet this seems not possible: in particular the question whether to trust a self-signed certificate is a process that is inherently socio-technical, for it is made of interactions between users, user interfaces, browsers, and even servers. Its security should be analysed by considering all those components and their interaction.

TLS technical security has been intensively studied (e.g., see [4], [5]), as well its usability (e.g., [6], [7], [8])) but a framework for a combined socio-technical security analysis still does not exist. In such a framework, system models should comprise not only technical but also social/human components; security depends on how these components interact and cooperate, because it is this collaboration that builds or undermines security. Since traditional security has mainly focused on the technical perspective, socio-technical security is a new frontier, which needs to be tackled.

Contribution. This paper proposes and follows such an innovative approach to security. It discusses a framework for socio-technical security analysis and a suitable threat model, and applies it to study two relevant problems with TLS certificate validation and self-signed certificates. The first is (a) the analysis of the interaction ceremonies between users and the most famous browsers in the market. The second problem is (b) the analysis of the understanding that users have about how TLS works. This latter study is still to be completed. The here-proposed framework needs first to be instantiated into research methods proper of cognitive science. From these two studies we gained valuable knowledge about tools that can be realistically applied for socio-technical analysis of security.

Structure. Section II presents the socio-technical security analysis model. Section III describes the security analysis applied from the "Computer" to the "Network" layer; Section IV describes the needed HCI (Human Computer Interaction) methods and research processes to perform that same analysis, specifically between the "Human" and the "Computer" layers. Section V comments the related work, and Section VI discusses the obtained results, presents some future work, and

concludes the paper.

II. Socio-technical security analysis

Our focus on socio-technical security compels us to revise traditional analysis techniques. Depending on the objective of the analysis, in fact, we may need different methodologies and tools. An analysis focusing on the technical side (communicating processes, applications and interfaces), with attackers controlling the networks or the interfaces, usually requires tools to reason about the behaviour of software components. Conversely one addressing the social side (persona and user behaviour) may require a research methodology to observe and reason about users interacting with the system.

We are interested in defining a framework where to model and analyse these two aspects together i.e., a system's social and technical components, in an integrated manner. We describe here a variant of Bella *et al.*'s concertina model [9]. Therein Alice and Bob are not metaphors for communicating processes, as in traditional protocol security, but personae linked to a set of interaction layers that connect humans and computers and, via the network, them with other computers and users (see Fig. 1). All the procedural elements that concern Alice, subscripted with an A, are drawn on the left side. The interactions between those elements are represented in grey boxes. The full description of these elements is given in [9]; here we recall that S_A is Alice's self, that expresses Alice's persona in a specific situation, P_A. It can be for example Alice at work. UI_A is an abstraction for all the interfaces that she uses to interact with the networked application p_A. This application interacts, through the network, with p_B, the only element of the here incomplete Bob's side. (Bob's side can have the same layered structure as Alice's.) Our variant (see Fig. 1) has also (1) a context, C_A; and (2) a threat model (black triangles). Context is a complementary element that affects our system's interactions, especially the user's interactions, for example, how often or with which probability, a user acts in a certain fashion. Context could be the user's physical space, such when visiting the university of Luxembourg (UNILU), which influences user's perception of security when receiving an invalid certificate warning from a web site in the "uni.lu" domain, versus one with domain "med.up.pt". As a threat model we assume an intruder who controls, not only the network, as in the classical Dolev-Yao model [10], but also the interactions between the application, the user interfaces, the persona, and the context. The places where the attacker can strike are indicated as black triangles. The intruder's abilities are however as in the traditional Dolev-Yao model, with the adjunct ability of creating new interfaces with the elements he interacts with. By acting with a wider variety of interactions as well as with the context, the intruder can influence both the technical components and the user behaviour.

In our model, security depends thus on what happens across the layers, and its analysis results richer: we therefore talk of *socio-technical security* analysis.

In the sequel, we introduce the methodologies and tools that we have selected to analyse socio-technical security systems and evaluate/comment on those selected tools in

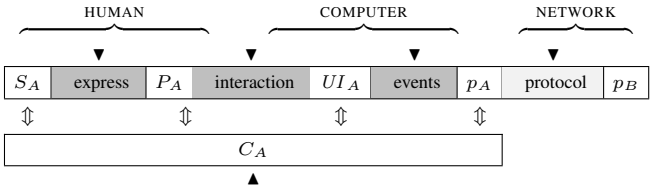

Fig. 1. The multi-layered security and threat model. Arrows are possible attacks. The intruder can control context, interaction, events between application and user interface, and the network.

two studies concerning socio-technical understanding of the security of TLS certificate validation. We have successfully applied formal methods (model checking) when considering layers from "network" until "computer", and when assuming users behave non-deterministically. This confirms Martina and Carlos's assertion that formal methods are useful to analyse security ceremonies [11]. To study instead security with more realistic user models we had to look into research methods of human-computer interaction (HCI) and cognitive science practices; as, to our knowledge, there is no framework for security analysis in those practices, we had to define one.

III. Study 1 - TLS Certificate Validation

From Sect. I we recall that a necessary condition for a server's identity to be authenticated is that the browser verifies the server's TLS certificate. If it cannot, because, for example, the certificate is self signed, the success of authentication may depend on the user: a browser can ask him to decide whether to proceed or abort the session.

In this study we analysed four of the most popular browsers – Chrome, Firefox, Internet Explorer, and Opera Mini – and how they interact with users when they encounter a self-signed certificate. Since the four browsers run different engines (i.e., p_A) and ceremonies with users (i.e., interactions between P_A and UI_A), the analysis of the structure of the dialogue browser-user is rich in possibilities.

This analysis (cf. Fig. 2) is about the layers that span from p_B (server) to P_A (user). In fact, we modelled a server, a browser, an abstraction of the interface, and a simple model of a user who chooses non-deterministically among the options that are offered to him. Context is not necessary here, for the simple model of users is context-independent.

We tried different formalisms to model the entities in agreement with our multi-layered security model. We first used flow charts but, although intuitive, this formalism was not the best to model multi-layered interactions, aside from the fact that they lack formal semantics, a limitation that precludes any formal analysis. We thus chose UML activity diagrams, a fortunate choice for three reasons. They fit well with the layered representation, they give immediately an easy reading of TLS sessions (i.e., a quick glance at the diagrams of the browsers under study shows clearly their different validation mechanisms), and they can be easily translated into a formal

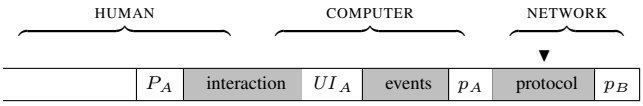

Fig. 2. The multi-layered security and threat model used in Study 1. There is no context. The intruder is a man-in-the-middle.

language. We built the diagrams of Chrome and Firefox by looking at their official documentation and code. Internet Explorer and Opera Mini are closed-source, so we studied them empirically.

The pictures show a combination of mechanisms: Chrome complies with HTTP Strict Transport Security (HSTS) policy –a policy whereby a web server announces using only HTTPS– , Internet Explorer uses warnings extensively, Firefox is the most complete having a complex engine and elaborate user interactions, and Opera Mini clearly aims at being lightweight (as is designed for mobile platforms).

While modelling them in UML, we did not describe their full functionalities but limited them to how each browser treats certificate validation. Because of space limitations we only show here the Chrome diagram (Fig. 3). Our diagrams have four columns each representing communicating elements. Three are entities: User, Browser, and Server. Browser distinguishes two standard sub-entities: User Interface and Engine. Entities have a begin circle that points to their first activity. Thick arrows depict the flow of activities among different entities, while thin arrows stand for the internal entity flow. Arrow labels define the objects that are exchanged between activities. Some activities need to access data-stores, which are linked to activities via dashed arrows. Most activities are self-explanatory and common to all browser diagrams, such as *Display Webpage* and *Type/Click URL*. To keep the focus on the browser, the server activities are reduced to *Init.TLS*, whereby the server starts the TLS handshake on its side, and *Finish TLS*, where it concludes the handshake. There is no room to describe in further detail the diagram, but it can be easily understood with a basic UML background.

To carry on the analysis we coded the UML diagram in a variant of CSP (Communicating Sequential Processes) [12] called CSP#; however, a prototyped tool that translates UML activity diagrams in CSP will be available soon [13], a tool we would like to test in the future. We also modelled an intruder, a Dolev-Yao controlling the network, and the user. Capturing the complexities of user behaviour by a formal model is a challenging open issue. As explained in the introduction, we modelled the user as a non-deterministic process: this is the weakest assumption about the user skills: a ceremony that is secure for a non-deterministic user, is also secure for any user.

The last step of our prototype methodology consisted in defining relevant security properties. We identified four socio-technical properties that bind TLS session, validation mechanisms, and user choices. We expressed them in linear temporal logic. One property is meant to evaluate the user involvement: it assesses whether the browser always warns the user when certificate validation fails. Two properties aim to evaluate whether the mechanisms that browsers adopt to manage failed certificate validations protect users from man-in-the-middle (MIM) attacks (e.g., if browsers can prevent users accessing a page controlled by the intruder). The last property is about informing the user that a MIM attack might have occurred in previous TLS sessions.

We verified the properties with the PAT (Process Analysis Toolkit) model checker [14]. The most interesting results regard Firefox. PAT reports a trace showing that Firefox does not warn the user when a certificate validation fails. This is due to the drawbacks of storing server certificates permanently, which Firefox allows its users to do. Moreover, it is worth noting that no browser keeps records of past warnings, exposing users to vulnerabilities when they bootstrap with MIM. This finding suggests a novel, more secure, strategy for browsers. Browsers should maintain a cache of invalid certificate hashes. In doing so, it would be possible for browsers to warn users when a different invalid certificate is presented by a server with which the browser has communicated in the past. Looking into this strategy is a matter of future work.

IV. Study 2 - Human Behavioural Model

This second study is on-going work on the layers that go from UI_A to S_A, thus it focuses on the human facing several options proposed by the computer. Studying user behaviour requires tools and research methods commonly employed in experimental psychology like surveys, diaries, focus-groups, interviews or non-interfering observations in a laboratory setting. By using those methods we do not intend to create a model of human behaviour, but rather to understand both quantitative (e.g., find whether patterns of secure/insecure behaviour exist) and qualitative aspects (e.g., explain those patterns) of human behaviour with regard to socio-technical attacks.

In our multi-layered security and threat model, we assume that socio-technical attacks may strike as indicated in Fig. 1. To study how users behave therein, we use the traditional hypothetical-deductive research process [15] that we have re-adapted to our needs.

The process starts when we define a hypothesis about an attack scenario. Starting from the multi-layered security and threat model, we identify key components such as the user (i.e., Human side), the application (i.e., Computer side), the context and the attack (and eventually some defence). Attack and Defence may compete: the former pushing users to an insecure behaviour, the latter, if present, to a secure behaviour.

To clarify the roles of the components and how they relate to one another during an attack scenario, we give every component a state and the possibility to act with input/output actions according to a Behaviour Control Process (BCP). A component's state is its being in a certain moment that will be used by the BCP to determine the next action to perform (like communicating/interacting with another component, changing its own state or even doing nothing). The state, like the BCP, can be very complex and we may not be able to formalize them fully. For instance, a browser integrates code acting as BCP (as seen in Section III) but we do not have an equivalent for the user, we can only inquire properties of the resulting behaviour by observing it during a laboratory experiment.

A hypothesis about potential patterns in user behaviours and the role of the different components is thus expressed using our model Fig. 4. Insights regarding the human behaviour can be gained from human information processing models, the decision taking research domain, error modeling and so on. The analysis of some traces of socio-technical attacks issued by incident response teams can also be a relevant source of information regarding the attack scenario.

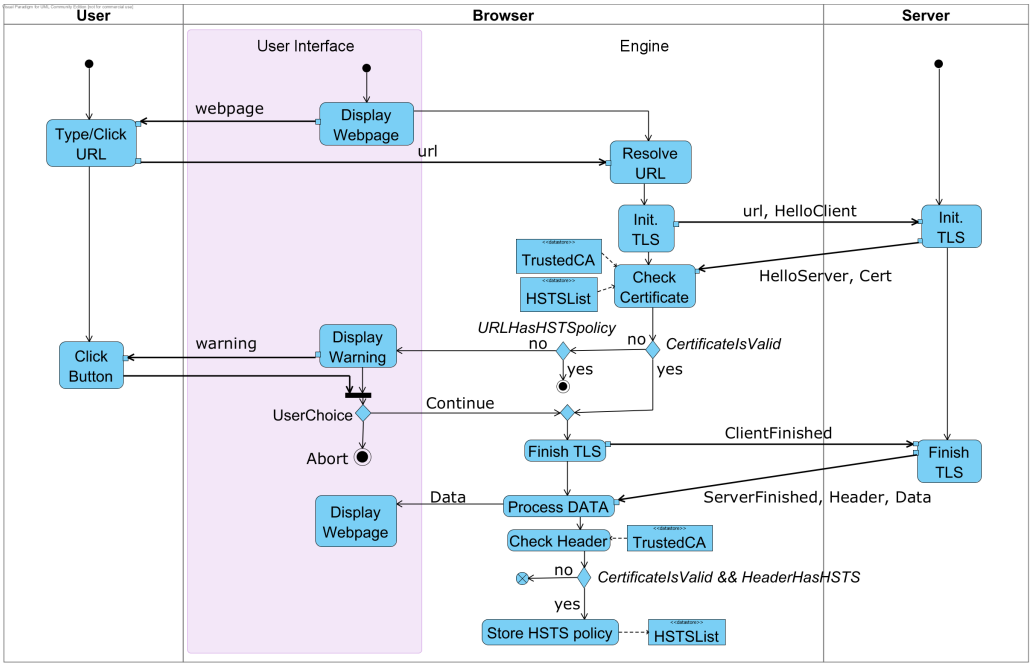

Fig. 3. Activity diagram for TLS certificate validation in Chrome

From this preliminary work the researcher will be able to choose the methodology that suits best the elicited hypothesis. The most important will be to define what is considered a secure behaviour in the following experiments, how this will be observed and what methodologies will be used to query the qualitative aspects of this behaviour. Those questions addressed, we will be able to design and implement the experiment to run.

As this study is still ongoing work we cannot contextualize the framework in a research process, but we can instantiate it to describe an attack scenario. We demonstrate the use of this framework by considering an attack against the TLS certificate verification in Google Chrome. We know that users already ignore 60% of Interstitial Warnings (IW) in Google Chrome [16] and this rate may increase if an Attack changes the user's state just before he makes a choice among the options UI_A offers him. By ignoring, we mean that users prefer to choose the option "Proceed anyway" and then store the self-signed certificate, over "Go back to safety" (equivalent to the back action) and closing the tab. An Attack controlling the user interface (i.e., Man-in-The-Browser) can send some specially crafted information (i.e., the payload) in order to force the user (BCP) to deviate from the prescribed (i.e., secure) behaviour. In Fig. 4 we see that the attack strikes in the P_A. UI_A interaction. We consider the application "Warning" being all the "Computer" layers until UI_A and the user being all the "Human" layers until P_A.

Here we choose to convey the payload by a fake IW which will be shown before the genuine one. The payload aims at misleading the user in interpreting a self-signed certificate as SELF-signed certificate, that is "certificate signed by S.E.L.F.", where SELF is a new certification authority yet unknown by the user's browser, but introducing itself as trusted in the text

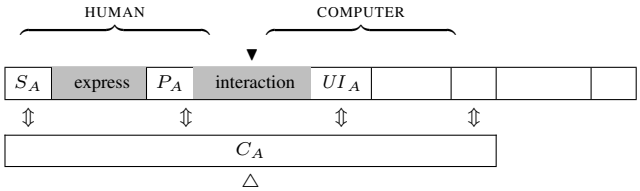

Fig. 4. The multi-layered security and threat model used in Study 2.

of the IW. This introduces a polysemy on the word "self" that may lead the user to misinterpret the meaning of the word (called equivocation fallacy e.g. "The sign said "fine for parking here", and since it was fine, I parked here.").

As presented by the Sequence Diagram in Fig. 5, the fake IW only offers a "continue" action that leads to the genuine one. The genuine warning offers the option to bring the user back to the fake interstitial warning (loop) or to store the self-signed certificate (then the attack succeeds). The only way for the user to escape this loop and make the attack fail is to hit the browser's close button at any time or to hit the back button when he is on the fake IW.

The context C_A does not actively play a role in this attack scenario but could be used by a defence (see the empty arrow in Fig. 4). For instance, in addition to the explanations the Genuine Warning given to the user, a link to some education / information material could be proposed. This could affect the user's state and change his following behaviour, hopefully to one that tends to be more secure.

After having been clearly defined, the attack scenario is implemented. The implementation heavily relies on the choice of some tools. If the attack uses software, such as a phishing attack, then we need a software tool to implement and launch it; if the attack is of the social-engineering kind, implementing it will probably mean to train an actor. In the case of this

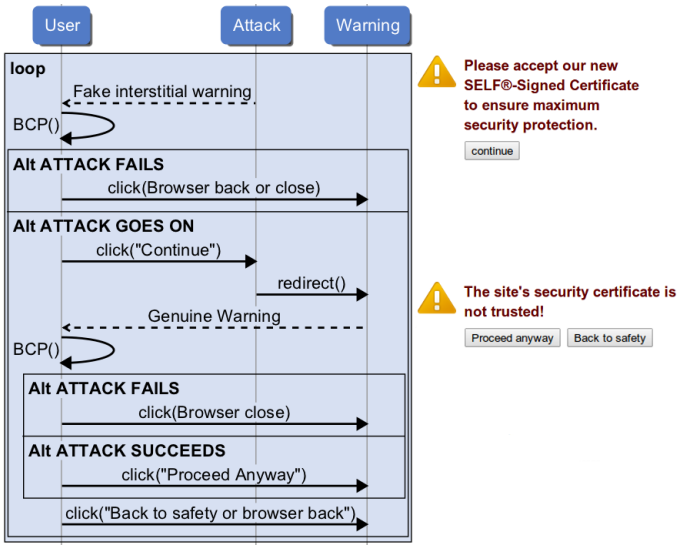

Fig. 5. Sequence diagram for the studied attack scenario. We use alternatives this way to emphasize the needed actions to make the attack fail or succeed. The possible use of the Context C_A as a defence is not represented here for the sake of simplicty.

attack scenario, it is fairly easy to design a fake IW from the Chromium's source code. A tool like BeEF [17] could also come handy to reproduce experiments amoung the participants.

To test any hypothesis about how people resist such misleading inputs we need to instantiate the whole research process (e.g., to decide how to launch the attack, what and how to observe, what to ask users afterwards, etc.) and to run the corresponding experiments. Then we could observe the patterns that lead users to resist or to fall for the attack and try to explain those patterns with the analysis of the qualitative investigations. This is future work that we plan to do in the usability laboratory.

V. Related Work

In this section we present some related work that has also focused on the security analysis and usability of browser security warnings and, in particular, of TLS certificates. There is much work available on the analysis of the TLS protocol but we only refer to two recent works which summarise most security problems. SSL/TLS certificate validation has many implementation vulnerabilities specially in e-commerce websites [5]. This paper analyses SSL/TLS certificate validation in non-browser applications. In addition, little attention is paid to the problem of correctly authenticating the service provider by the users. Josang's et al. work [4] tries to develop a framework to provide for user authentication assurance. However, these researchers do not focus on the fact that a user authenticating the service provider will provide more ways to perform socio-technical attacks. Attackers will probably focus even more on social engineering to implement successful attacks. These authors' main conclusion is that it is essential to integrate user and server authentication in the same framework. So to devise a platform where both socio and technical aspects of TLS security can be developed and implemented together. We also share this view.

Works about security usability and browser warnings conclude that users do not look at browsers' cues or security indicators and that they are lead to the incorrect decision in 40 percent of the time [18]. Some works try to improve user's awareness, for example with visual augmented security [19] which informs users about security decisions or even with improved interaction techniques to prevent spoofing [20]. These are small improvements as visual deception attacks can fool knowledgeable users even when these are alert to find security problems [18]. Sunshine's et al. [8] further concludes that even custom/improved warnings are not enough and ideally security designers should avoid them altogether.

For a more integrated security analysis of the socio-technical aspects in general, we could not find much research available, much less even for the TLS authentication scenario. As already mentioned, Cranor's work [3] introduces a generic framework to reason about the human in the loop regarding security to help system's designers to avoid mistakes within the communications between several system's components. However, it does not focus on analysing and implementing socio-technical attacks. Other research has been done to develop socio-technical frameworks to anticipate organizational threats [21] or define warning systems for socio-technical incidents [22]. It is not referred what will happen if some non-specified vulnerability is exploited or how they would change their procedures in order to mitigate and help the user to avoid these in practice, as we plan to do in future research.

VI. Discussion and Future Work

We presented a framework for analysing socio-technical security of systems, and applied it to study the TLS certificate validation and self-signed certificates. The multi-layered security and threat analysis model presented in this paper can express and integrate socio-technical interactions for all components within the system in analysis together with the context where that system is set. As is common knowledge, security can include several layers, and if we miss one of them, we can be opening vulnerability doors where those layers can be compromised. If we are able to see the whole picture and include the whole system (both human and technical) in the security analysis, it will help us find and tackle more vulnerabilities between all the involved elements.

Human intervention is central to the success/failure of an attack, specially in the scenario of TLS certificate validation. This problem needs to be tackled first at the level of understanding. If users do not understand the concepts for which they are being asked to act upon, then their security decisions may not be based on true assumptions. How to address this situation? Training users? Improve warnings and messages in a way users can both read them and gain a good understanding of security protocols (can both be achieved)? Or simply accept users do not need to fully understand security protocols? Also, if users are in different states of mind regarding the concept of certificates, maybe the experiment with the presented socio-technical attack can generate richer and more close to reality data. This could help us encounter different patterns of behaviour regarding a user's personal context and characteristics to define better ways to apply defences to such types of attacks.

Regarding Study 1, the tools selected to perform TLS certificate validation (i.e., UML, PAT, CSP) were adequate and helpful in expressing and evaluating in detail the security analysis and properties of the different browsers in study. The introduced multi-layered model helped us to identify, not only the events and protocols between computer and network components, but also the interactions between the human user (P_A) and the user interface (UI_A). With the obtained expressiveness it is easier and quicker to identify inconsistencies and vulnerabilities in the security properties which can be further corrected.

Regarding Study 2, the same multi-layered model helped us to identify where attacks and defences could be applied for both generic socio-technical attacks as well as socio-technical attacks for the described scenario. By focusing the security analysis within the interactions between (P_A) and (UI_A) and the context where they are set, a common attack was easy to define. In addition, some defence that can be used to tackle those attacks was also introduced. Of course this is still theoretical, but using the HCI research process where we base our socio-technical studies, this defence can be evaluated in practice with real user experiments. Again UML was useful to express this socio-technical security analysis.

Another important contribution of this paper is that it introduces the concept of context, which is many times ignored in security analysis as is usually hard to define. However, within socio-technical security analysis, the context is a very important component which cannot be ignored because it can influence the user's set of mind when he is taking security decisions [23]. The context is external to the system components' interactions but can interfere with all of them, and so is even more relevant to be included in system security analysis. For socio-technical vulnerabilities, attackers use many times the context where the system is set to influence or press the user into falling for their attack. In the same way, our research can be used to find out if defences can similarly use the context to influence and help the user identifying and avoiding such attacks.

The presented framework is still ongoing research but it can already help us to instantiate with detail, and depending on the attack scenario, the several layers that integrate the security system to be analysed. More importantly, users may act differently when in different contexts (e.g. usability laboratory, at work or at home) [23]. Experiments must be designed so that context is a crucial part of the study, as presented in this paper with our multi-layered security analysis model.

Future work will include the performance and evaluation of the presented socio-technical attack scenario with users, and hopefully extract from these a set of user interaction patterns which will help us better understand why users commonly fall victims for the attacks that relate with TLS and self-signed certificates. More importantly, with such an integrated model, it will be possible to define socio-technical defences, which can be applied at several layers of the system's interactions, in similar ways as a socio-technical attack is performed. The main goal is to help the users to act in a behaviour more resilient to socio-technical attacks by reverting or interfering with the techniques that such an attack can use to deceive a user. Similar techniques can be used successfully but in a reversed manner to make the users better identify the risks that can be involved in their actions and, hopefully, "defend" themselves.

REFERENCES

[1] A. Jøsang, I. G. Pedersen, and D. Povey, "PKI seeks a trusting relationship," in *Proc. of ACISP 2000, Brisbane, Australia*, 2000.

[2] E. Rescorla, "HTTP Over TLS," RFC 2818, 2000.

[3] L. F. Cranor, "A Framework for Reasoning About the Human in the Loop," in *Proc. of the 1st Conf. on Usability, Psychology, and Security*. USENIX Association, 2008, pp. 1–15.

[4] A. Jøsang, K. A. Varmedal, C. Rosenberger, and R. Kumar, "Service provider authentication assurance," in *Proc. of PST '12*. IEEE Computer Society, 2012, pp. 203–210.

[5] M. Georgiev, S. Iyengar, S. Jana, R. A., D. Boneh, and V. Shmatikov, "The most dangerous code in the world: validating SSL certificates in non-browser software," in *Proc. of ACM CCS'12*, New York, NY, USA, 2012, pp. 38–49.

[6] S. Schechter, R. Dhamija, a. Ozment, and I. Fischer, "The Emperor's New Security Indicators," *IEEE Symposium on Security and Privacy (SP '07)*, pp. 51–65, 2007.

[7] S. Fahl, M. Harbach, T. Muders, M. Smith, L. Baumgärtner, and B. Freisleben, "Why eve and mallory love android: an analysis of android SSL (in)security," in *Proc. of ACM CCS'12*. New York, NY, USA: ACM, 2012, pp. 50–61.

[8] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor, "Crying wolf: An empirical study of SSL warning effectiveness," in *Proc. of USENIX'09*, 2009.

[9] G. Bella and L. Coles-Kemp, "Layered Analysis of Security Ceremonies," in *Proc. of the 27th IFIP Int. Conf. on Security and Privacy, 4-6 June 2012, Crete, Greece*, 2012.

[10] D. Dolev and A. Yao, "On the security of public-key protocols," *IEEE Transaction on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[11] J. E. Martina and M. C. Carlos, "Why Should We Analyse Security Ceremonies?" in *Proc. of CryptoForma, May 25, 2010, Paris, France*, 2010.

[12] A. R. Hoare, *Communicating Sequential Processes*. Prentice Hall International, 1985.

[13] I. Abdelhalim, S. Schneider, and H. Treharne, "An integrated framework for checking the behaviour of fUML models using CSP," *International Journal on Software Tools for Technology Transfer*, 2012.

[14] J. Sun, Y. Liu, J. S. Dong, and J. Pang, "PAT: Towards Flexible Verification under Fairness," in *Proc. of CAV'09*, ser. LNCS, vol. 5643. Springer, 2009, pp. 709–714.

[15] J. Lazar, J. Feng, and H. Hochheiser, *Research Methods in Human-Computer Interaction*. John Wiley & Sons, 2010.

[16] A. Langley, "Living with https," July 2012. [Online]. Available: http://www.imperialviolet.org/2012/07/19/hope9talk.html

[17] "Beef the browser exploitation framework." [Online]. Available: http://beefproject.com/

[18] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '06. New York, NY, USA: ACM, 2006, pp. 581–590.

[19] D. Shin and R. Lopes, "An empirical study of visual security cues to prevent the sslstripping attack," in *Proceedings of the 27th Annual Computer Security Applications Conference*, ser. ACSAC '11. New York, NY, USA: ACM, 2011, pp. 287–296.

[20] R. Dhamija and J. D. Tygar, "The battle against phishing: Dynamic security skins," in *Proceedings of the 2005 symposium on Usable privacy and security*, ser. SOUPS '05. New York, NY, USA: ACM, 2005, pp. 77–88.

[21] K. Worton, "Using socio-technical and resilience frameworks to anticipate threat," in *Socio-Technical Aspects in Security and Trust (STAST), 2012 Workshop on*, 2012, pp. 19–26.

[22] B. Al Sabbagh and S. Kowalski, "St(cs)2 - featuring socio-technical cyber security warning systems," in *Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), 2012 International Conference on*, 2012, pp. 312–316.

[23] A. Sotirakopoulos, K. Hawkey, and K. Beznosov, "On the challenges in usable security lab studies: lessons learned from replicating a study on ssl warnings," in *Proceedings of the Seventh Symposium on Usable Privacy and Security*, ser. SOUPS '11. New York, NY, USA: ACM, 2011, pp. 3:1–3:18.