Hey TPM, Sign My Transaction

Ijlal Loutfi and Audun Jøsang
University of Oslo,
ijlall@uio.no,josang@ifi.uio.no

Abstract. Online banking services have been fighting malware for the last 10 years. However, the emergence of targeted Man-in-the-Browser (MitB) banking malware has given the upper hand to attackers in this fight. MitB Trojans hook themselves into end users browsers, intercept their banking credentials, alter their transaction details, and then transparently alter the HTML of the bank web pages they are viewing. The end user then approves the transaction unsuspectingly. MitB is able to evade traditional defense mechanisms such as intrusion detection systems, anti-fraud policies, as well as strong authentication mechanisms. In this paper, we present a solution aimed at detecting and preventing MitB attacks. The solutions rely on concepts related to the trusted computing paradigm. It defines a trusted path in the end user platform, which allows it to take a screen-capture of the displayed transaction details displayed by the end users screen, and forward it in the same TLS session as the transaction details to the bank. The trusted path is hardware-protected by the TPM, and ensures that the screen-capture has not been altered by any malware. The solution also relies on the TPM PKI in order to give assurance to the bank that the screen-capture originates form a genuine user. The solution is aimed at corporate end users and industries.

1 Introduction And Motivation

Our ambition to build an interconnected modern information society has exposed us to a new vector of threats, which is that of cybercrime. The Federal Bureau of Investigation (FBI) reports that we are witnessing a decline in physical crimes, such as bank robberies, as opposed to an increasing rate of cybercrime [1]. Because cybercriminals are often motivated by monetary rewards, financial and banking institutions are one of their main targets [2]. Financial institutions have been fighting malware that targets online banking for over ten years [3]. However, one can observe that financial Trojans are only getting more flexible and advanced. One of their latest advances is the Man-in-the-Browser (MitB) Trojan attack, which will be the focus of this paper. What makes MitB attacks popular is the ease by which they can be deployed to many systems at once, via phishing links, botnets and through compromising legitimate sites [4]. What makes them dangerous is their ability to go beyond a traditional Man-In-The- Middle attack: Instead of just intercepting or piggybacking traffic via a proxy page, MitB attacks fully take over a user's view of a website, and control the browser in an effort to trick the user into thinking that everything is normal while the attack is being executed. By slightly altering the HTML of banking websites, attackers can steal money without a user's knowledge. Once the user logs in, MitB Trojans can also redirect any sensitive traffic to an attacker's system, while leaving the TLS/SSL communication protections intact [4], [5].Traditionally, malware has been guarded against in one of two ways:

Presented at the Norwegian Information Security Conference 2016 (NISK-2016).

- ✓ Defense mechanisms aimed at preventing the end user platform from getting compromised: mainly through end point intrusion detection systems.
- ✓ Defense mechanisms aimed at preventing the malware (once I has infected the end users' platform) from conducting the attack, mainly through defining and implementing anti-fraud policies, that are induced from intelligent data analysis of past fraudulent behavior.

We believe that one of the reasons why these defense mechanisms are inefficient, is because they are trying to solve all MitB attack scenarios at once. This stands in sharp contrast to how MitB attacks are conceived. According to the 2015 RSA Cybercrime report, there are 4 main new trends that are changing the threat landscape. One of these main trends is that threats continue to grow more targeted and mode advanced [2]. Motivated by this trend, the present paper focuses on a specific use case. The targeted users of our solution are enterprise employees, mainly in the finance and banking sectors, who are bound by their corporate policies to perform all work related tasks from a specific machine that is owned and managed by the firm. We think this does not hinder the usefulness of the solution, as corporate users are the most vulnerable target of MitB attacks. Indeed, the criminal community is heavily focusing its attacks today on corporate- banking customers, as the available funds are often greater, transaction limits are higher and the corporate customer has access to a wire transfer or Automated Clearing House (ACH) services through the online banking interface [6]. However, the solution is also relevant for the end-user market, which would only require service providers to keep track of platforms used by their customers. The solutions has no restriction on which browser is being used. The rest of the paper is organized as follows: we first explain the workings of the Man-In-The-Browser attack. The literature review introduces previously proposed solutions as well as their shortcomings. Subsequently, we explain the architecture as well as the technical details of our solution. Finally, the paper closes with a discussion and a set of conclusions, as well as an overview of the expected future work.

2 Man-In-The-Browser (MITB)

2.1 Technical Overview

The idea of MitB attacks was first presented by Agusto Paes de Barros in 2005. By 2007, financial fraud Trojans was already using it against end users. MitB spreads through Trojan malware. It takes the form of an application hooked into the browser [3]. It defines a new breed of attacks whose primary objective is to spy on browser sessions (mostly banking), and in that process intercepts and transparently modifies the HTML of the web page being viewed by the end user. In a classic MitB attack, it is very likely that what the user is seeing on his/her browser window is not something which the actual server sent. Similarly, what the server receives on the other end might not be what the user intended to send. MitB attacks happen at the presentation layer. Since browsers have high level privileges on a system, if an attacker is able to execute a process through the browser, then that process can be executed with high level privileges [7]. More specifically, MitB malware mostly leverages browser

extensions in order to exploit the operating system. Browser extensions are typically used to enhance users' experience within the browser while surfing the Internet. Browser extensions and features to which the Trojan hooks can include [4]:

- ✓ Plugins,
- ✓ Browser Helper Objects (BHO),
- ✓ JavaScript,
- ✓ Add-on features,
- ✓ Ajax calls,
- ✓ Browser API Hooking,
- ✓ DOM Object models.

For the sake of brevity, we will not explain how the Trojan attaches itself to each one of the above mentioned browser add-ons and/or features. Once a MitB Trojan has infected the machine and hooked itself to the browser, it can be controlled via a configuration file or a web injection file, which is updated at certain time intervals as part of a botnet. These configuration files may be obfuscated with different types of encoding. The configuration file and web injection file allow an attacker to control sessions and inject custom code into HTTP traffic. They also allow the Trojan to run when certain websites are visited such as banking institutions. MitB attacks can also erase traces of its actions from the browser's history, including cookies [4].

A successful MitB attack can show the user a completely consistent picture of the transaction he or she is executing, while actually executing a totally different transaction with their bank. Transaction details may be modified or totally unrelated transactions may be launched, all without the user or the bank ever understanding that an attack is underway [6].

All MitB attacks have two main phases: infection, and transaction takeover. The first phase can happen through phishing or browser vulnerabilities. An example of the second phase, can have the structure illustrated in Table II-A. MitB attacks have proven to be a very effective form of attack. In a documented attack against a German bank in 2011 the following has been reported: from 11 August until 26 August, the cyber gang stole a total of EUR 193,606, or about EUR 12,000 per day. From 30 August at 15:50h until 1 September at 11.48h, they stole a total of EUR 42,527, or about EUR 21,000 per day. There was a gap in their cybercrime activities of four days. If we include this gap, we see that the cyber gang made an estimated total of around EUR 300,000 in just 22 days. The longer the Trojan remains active, the more money is it able to steal for this cyber gang. On an annual basis, this cyber gang could make close to EUR 5 Million. Or US$ 7.3 Million annually [8].

3 Proposed Solutions And Their Shortcomings

Given the danger posed by Man-In-The-Browser attacks, a number of solutions have been proposed for their detection and prevention. In this section, we explore these defenses, as well as why they are failing to protect end users from MitB attacks.

- ✓ Many banks have been proposing solutions that would strengthen their authentication mechanism. These solutions range from two-factor authentication, biometrics Grid cards, OTP tokens and out of band OTPs. All of these defense

mechanisms can be bypassed by MitB malware, as it can intercept or wait until the user is past these challenges before taking over [6], [9].

- ✓ Network defenses such as web application firewalls, IDS and IPS systems have difficulty detecting this attack since it occurs locally on the client side. Furthermore, while decrypting TLS banking sessions may be a solution, this is not practical as it poses a big breach to user's privacy [4].
- ✓ Preventing browser extensions and scripting has been proposed as a solution to mitigate MitB attacks, or preventing scripts to run over TLS connections. There are available methods to restrict browser extensions from running, but this means that certain websites may not operate properly, and restricting browsers is difficult in today's age of multimedia operations [4].
- ✓ Geo-location, which is Based on the end-users computer IP address, to determine the users geographic location and compare it to typical locations used by this user. While effective when credentials are stolen and used elsewhere, these techniques fail against MitB because the malware is in the user's regular browser, at the user's typical location.

4 Proposed Solution

4.1 Profile Of The Solution's Target Audience

The trusted platform module is a central piece of our proposed solution. This makes it bound to a specific platform. While there are no technical reasons why our solution cannot be applied to all of end users, we believe that from a usability perspective, the ideal target audience for the deployment of this solution would be enterprise employees, mainly in the finance and banking sectors, who are bound by their corporate policies to perform all work related tasks from a specific machine, that is owned and managed by the firm. The solution has no restriction over which the browser is being used. We think this does not hinder the usefulness of the solution, as corporate users are the most vulnerable target of MitB attacks. Indeed, the criminal community is heavily focusing its attacks today on corporate-banking customers, as the available funds are often greater, transaction limits are higher and the corporate customer has access to a wire transfer or Automated Clearing House (ACH) services through the online banking interface [6].

4.2 Solution Design Principles

This section introduces the design principles that motivate the architecture of the solution. The solution should not rely on end-point intrusion detection systems.

Table 1. Vulnerable Workflow [4]

Step	End User	Malware	Financial System
A	Visits financial institution site	Wakes up as this financial institution is on its target list	Displays login screen
B	Login with username and password	May harvest this, or more likely just wait	Processes login
C	Requests fund transfer form (ACH or wire transfer)	Waits	Displays form
D	Enters origin and destination accounts and amount	Intercepts users request, substitutes alternate amount and destination	Receives malwares request, sends transaction details for review and requests one-time-passcode (OTP)
E		Intercepts sites transaction detail confirmation, modifies them to correspond to users initial request	
F	Views transaction details (which look fine) then consults OTP token and enters the numeric code into their Web browser	Waits	Receives users valid OTP code and executes

1 The solution should not rely on end-point intrusion detection systems.
2 It is assumed that the end-user platform can be infected by malware, and thus cannot be trusted.
3 The solution should not rely on fraud detection policies.
4 The solution does not assume that the displayed HTML pages of banking transactions are in accordance with what the server is processing.
5 The solution should provide a reliable mechanism for communicating the end-user's genuinely intended trans- action to the service provider.
6 The solution should have a trusted path that is hardware protected, in order to assess that the transaction the user intends to submit and execute is what the service provider is also receiving.

Solution Requirements
The end user platform should be equipped with a trusted platform module that can be enabled and activated.

4.3 Trusted Platform Modules

Our research has led us to conclude that a solution centered on the trusted platform module would allow us to satisfy the above mentioned design principles. Thus, we would like to present in this section TPM concepts that are necessary for understanding the rest of the paper. Trusted computing is a paradigm developed and standardized by the Trusted Computing Group. It aims to enforce trustworthy behavior of computing platforms by identifying a complete chain of trust, a list of all hardware and software that has been used [10]. This chain of software can then be compared to a list of known good

applications, unlike standard approaches such as virus scanners that try to recognize and eliminate bad software [11]. Trusted Platform Modules TPMs are of one of the main building blocks of this paradigm. TPM is defined by the TCG as a computer chip micro-controller that is attached to the motherboard [12].

TPM Integrity Measurement and Reporting: The TPM can securely store artifacts (encryption keys, passwords, certificates. It can also store platform integrity measurements that help ensure that the platform remains trustworthy. This is possible thanks to the 16 PCRs (Platform Configuration Registers) contained in the TPM.

A PCR is a 160 bit wide register that can hold a SHA-2 hash. Each SHA-2 hash corresponds to a measurement of a piece of software or hardware present on the platform. It is not possible to write directly to a PCR. The only PCR allowed operation is the extend(x): This operation calculates the new value of a PCR as a SHA-2 hash of the concatenation of the old value and x [13]. By definition, the extend operation is non-commutative, meaning that the order of events can be maintained. also, because the output of SHA-2 is fixed, we can store as many measurements in a sequence as we want.

One successful application of the PCR integrity measurements has been to measure the boot process: The extend operation is used to store a hash of a chain of loaded software in PCRs. The chain starts with the BIOS and includes Option ROMs2, Bootloader, OS and applications. Because the PCRs cannot be erased, this means that no program can cancel its execution from the TPM. The first item in the chain cannot be independently measured at runtime, and is referred to as the root of trust for measurement. We would aim for it to be immutable if possible. A platform is said to support authenticated boot when it follows this process, as it provides a way for users to authenticate their platform boot sequence against reference values [14], [11]. An optional follow up step to integrity measurements is integrity logging, which stores integrity metrics in a log for later use. Logging is recommended. Otherwise integrity measurements might need to be repeated in order to interpret PCR values. Integrity reporting is the process of attesting to integrity measurements recorded in PCRs [15], [12].

TPM Attestation: The previous paragraph establishes the process by which the TPM measures and records the integrity of (selected) soft- ware and hardware that is running on the platform. Integrity measurement can be a much more useful operation if we had a mechanism for the TPM to report these measurements to a third party it is communicating with. This third party can then check if the platform configuration is known, and depending on policy, decide if the platform should be trusted for the transactions it wants to execute with that platform. This process, called attestation, is facilitated by a PKI that has been specifically defined for TPMs [15], [16]. The following are the parts of the TPM PKI that are relevant to this paper:

Endorsement key: An Endorsement Key is a special purpose TPM- resident RSA key that is never visible outside of the TPM. Because the EK can only be used for encryption, possession of the private EK can only be proved indirectly, by using it to decrypt a value that has been encrypted with the public EK. Therefore, while the EK cannot be used to produce a digital signature, it is able to provide for TPM authentication based on decryption operations [17].

Attestation Identity Key: An Attestation Identity Key is a special purpose TPM-resident RSA key that is used to provide platform authentication based on the attestation capability of the TPM. All AIK key present on the platform are encrypted based on the Endorsement Key. A challenger could use this information, along with other information in the credential to trust the platform via an attestation protocol [18].

Indeed, when challenged, the TPM can create a signed copy of its PCR values. The signing key (attestation identity key AIK) being demonstrably linked to a trusted platform, with privacy protection where necessary. This is then given to the challenger for inspection, along with the measurement log. The software running at the platform can be identified by matching the hash values in the log with reference data. This requires a list of reference integrity measurements (RIMs) contained within a Reference Manifest Database [19]. These measurements are collected from their original source: the software and hardware manufacturers [10]

4.4 Putting It All Together

Top Of The Mountain View: In order for the bank to execute a money transfer, it needs to make sure that the amount it has received from the user was generated and approved by him, and not by a MitB Trojan. In a parallel world, one way to solve this would be for the bank have access to the screen of the end user and check the amount being displayed to him against the amount it has received before executing the transaction. Given the impracticality of this solution, what we propose in this paper is a solution that provides equal guarantees, and that can be implemented by the aid of a TPM. The requirements for the solutions are as follows:

1. Every time an end user is making a money transfer, they should send a copy of their screen information to the bank
2. The end user should have a mechanism which ensures that the screenshot he is sending is not altered by a platform malware (MitB or not).
3. The bank should have a mechanism which ensures that the screenshot has not been altered by MitB malware, after having been generated by a trustworthy platform.
4. The bank should have a mechanism which ensures that he screenshot originates from a legitimate end-user platform. Section 4.4.2 explains how 1 and 2 and achieved. Section 4.4.3 explains how 3 and 4 are achieved.

End User Trusted Path Trusted Path: In order to take a screen capture of what is being displayed to the end user in a trustworthy manner, we cannot rely on a browser extension to do that, as we assume that the browser might be infected with a MitB Trojan. Hence, we need to trigger a call directly to the Graphical display driver, which would allow us to access directly the RAM memory space where the displayed information on the screen is being rendered. Furthermore, we need to ensure that neither the graphics driver nor the API calls have been infected by malware. In order to satisfy all of these requirements, the following steps should be implemented:

- ✓ Enable and Activate the client platform.

- ✓ Enable the authenticated boot.
- ✓ If the authenticated boot is successful, then we are sure that the components of the boot process haven't been altered by malware. hence, we can choose the graphics driver as our application's root of trust, as it is part of the boot sequence that has been already measured.
- ✓ A trusted path from the device driver until the RAM memory space where the content displayed on the screen should have already been defined. n a PC platform, the trusted path sequence is as follows:

    1 Graphics driver.
    2 API call from the driver to the GPU RAM(referred to thereafter API1).
    3 API call in the GPU RAM executing the driver call (referred to thereafter as API2).
    4 The program code executing the screen capture (referred to thereafter as PROG1).
    5 The program code copying the screen capture into the TPM protected memory (referred to thereafter as PROG2).
- ✓ Measure each component in the trusted path, and record the measurement value in the designated PCR for our solution(*MVAL= Measured Value):

    PCR(x)=SHA1 (SHA1 ((SHA-1(SH 1([Graphics Driver MVal*])
    +[API1 MVal*])+[API2 MVal*])+[PROG1 MVal*])+[PROG2 MVal*]).

- ✓ Compare the measured value to the reference measurement of the trusted path sequence.
- ✓ If the measured value matches the reference value, then we are sure that the trusted path has not been infected with malware. Hence we can proceed.
- ✓ Following the now measured trusted path command, take back screen capture of the screen.
- ✓ Copy the screen capture to a TPM protected storage memory. The sequence of events described above, satisfies the first band second requirements.
- ✓ It guarantees that:1) At the moment an end-user makes a money transfer, they send a copy of their screen information to the bank, and 2) The end user should have a mechanism which ensures that the screen capture they send is not altered by malware.

Bank's Remote Attestation: Once the screen capture has been copied into a TPM protected storage area, the following sequence should be followed:

1. The attestation Identity key pair is used to sign the screen capture.
2. The signed screen capture is sent with the transaction details in the same TLS session.
3. The bank uses the corresponding public key to verify the signature.
4. The bank verifies that the AIK belongs to the same end user making the transaction (through the above described TPM PKI).
5. The bank extracts the transaction amount from the end user's screen capture using Optical Character Recognition(OCR), and compares it to the amount it has received in the same TLS session.

6. If the amounts match, then the bank can proceed with the transaction, otherwise the bank suspends the transaction and notifies the end user.

In order to match the trusted screen capture with the un- trusted transaction data, the server must use OCR (Optical Character Recognition) to extract the transaction data from the received screen capture. The combination of a digitally signed screen capture with OCR can be considered as the Robust WYSIWYS system [20]. Commercial and open source OCR software packages are available. In its simplest form, OCR software takes scanned documents in bitmap format, and converts them into text files. More advanced graphical layout of digital documents will require a standard for geometrically formatting documents, so that the translation from analogue bitmap format to digital document format is unambiguous. Since the service provider generates the HTML content, the service provider has full control of the graphical layout of web pages, and can adapt the OCR to the graphical layout of their choice. The above described message flow assumes that:

1 The bank has a way to acquire the AIK public key.
2 There is an appropriate PKI structure to verify that the signature is from a genuine TPM.
3 The bank can verify that the signature belongs to the same end user.

TCG (Trusted Computing Group) defines a PKI structure that satisfies the first and second requirements This can be achieved through the use of Direct Anonymous Attestation, that links the AIK to the unique endorsement key present on each TPM [16]. However, it is up to the end users and their corresponding bank to find a mechanism that would satisfy the third requirement. For the context of this paper, this can be achieved during the new employee's registration phase with the bank. Indeed, in a corporate context, the platform a user is using is owned and managed by the corporate IT department. The IT department can decide on a way to communicate to the bank the new employee's AIK certificate during the registration phase (this can be done both online or out-of- band).

Table 2. Workflow Of The Proposed Solution

Step	End User	Malware	Financial System
1	Visits financial institution site	Wakes up as this financial institution is on its target list	Displays login screen
2	Login with username and password	May harvest this, or more likely just wait	Processes login
3	Requests fund transfer form (ACH or wire transfer)	Waits	Displays form
4	1)Enters origin and destination accounts and amount	Intercepts users request, substitutes alternate amount and destination 2) Takes a screencapture of the displayed screen and stores it in a TPM protected memory 3)Signs the Screen capture with the corresponding AIK	1)Receives malwares request and the signed screen capture 2) Verify that screencapture signature is both genuine and extract to which user it belongs to. 3) Extract the transaction amount from the screen capture.

Solution Message Flow

Now that we have explained the technical details of the solution, we will step back in this section, and abstract those details, in order to formulate a message flow of the end user/bank interactions. Table II illustrates how the proposed solution successfully detects the presence of a MitB Trojan of the client's platform, as well stops the bank from executing its fraudulent actions. If the bank detects that the amount it has received is not the same one that was displayed to the end user, it can immediately halt the transaction and not go forward with money transfer. This action successfully neutralizes the MitB Trojan. The bank can also notify the corresponding client about the fact that their platform is infected, so appropriate measures can be taken.

5 Conclusion And Future Work

In this paper, we have proposed a solution aimed at detecting Man-In- The-Browser attacks, and preventing their execution by the bank. Because the core of this solution is the trusted platform module, it is mostly usable for people whose default way of doing online banking is through a dedicated machine. This is the case of corporate employees. This group of users also constitutes the main target of MitB attacks, especially within the banking and finance industries. Just as MitB at- tacks are growing to be more sophisticated and tailored to specific platforms and users, so should the solutions aimed at preventing them. This paper also brings forward a novel way in which TPMs can be leveraged. While TPMs are currently being shipped with most end user devices, their functionalities remain largely underutilized. This is mainly due to the lack of attractive applications that can drive its activation and consumption. Furthermore, most of the currently available TPM applications such as Microsoft Bitlocker and authenticated boot are local client centric. However, our proposed solution is leveraging TPM so as to increase the security at the web application level, linking this way a platform's hardware to web applications. We believe that trusted computing, and hardware enabled computing in general, should be further investigated for it to become part of our defense arsenal against web threats. In line with this direction, our future work aims at specifying the architecture for the proposed solution for Windows computers and Android devices, and follow up with a prototype implementation for the Windows computer solution, so as it can be tested in real world scenarios

References

[1] R. Samani and F. Paget, “Cybercrime exposed: Cybercrime-as-a- service,” MCfee, Tech. Rep., 2013.
[2] RSA, “Cybercrime 2015: An inside look at the changing threat landscape,” EMCorporation, Tech. Rep., 2015.
[3] P. Krysiuk and S. Doherty, “The World of Financial Trojans,” Symantec, Tech. Rep., 2013.
[4] C. Cain, “Analyzing Man-in-the-Browser (MITB) Attacks,” SANS Institute, Tech. Rep., 2014.
[5] IBM, “Man-in-the-Browser (MitB) Glossary,” IBM, Tech. Rep., 2013, http://www.trusteer.com/glossary/man-in-the-browser-mitb.
[6] Entrust, “Defeating Man-in-the-Browser Malware: How to prevent the latest malware attacks against consumer and corporate banking,” Entrust, Tech. Rep., 2014.
[7] W. Alcorn, C. Frichot, and M. Orru, The Browser Hacker’s Handbook. Wiley, 2014.
[8] Finjan, “Cybercrime intelligence: Cybercriminals use trojans & money mules to rob online banking accounts,” Finjan Malicious Code Research Center, Tech. Rep. Cybercrime Intelligence Report, Issue 3, August 2009.
[9] G. Ollmann, “Man in the Browser Attack Vectors,” IBM – Computer Security Institute, Tech. Rep., 2008, presentation at CSI2008.
[10] A. Martin, “The Ten Page Introduction to Trusted Computing,” University of Oxford Tech. Rep., 2008.
[11] J. Lyle and A. Martin, “Trusted computing and provenance: Better together,” in Proceedings of the 2Nd Conference on Theory and
Practice of Provenance, ser. TAPP’10. Berkeley, CA, USA: USENIX Association, 2010.
[12] TCG, “Trusted Platform Module (TPM) Summary,” Trusted Computing Group, Tech. Rep., 2011.
[13] ——, “TCG Specification: Architecture Overview,” Trusted Computing Group, Tech. Rep., 2007.
[14] Symposium, ser. SS’07. Berkeley, CA, USA: USENIX Association, 2007.
[15] TCG, “TPM Main: Part 1 Design Principles,” Trusted Computing Group, Tech. Rep. Specification Version 1.2, Revision 116, 2011.
[16] ——, “TPM Main: Part 2 TPM Structures,” Trusted Computing Group, Tech. Rep. Specification version 1.2, Level 2 Revision 116, 2011.
[17] ——, “Endorsement Key (EK) and Platform Certificate Enrollment Specification: Frequently Asked Questions,” Trusted Computing Group, Tech. Rep., 2013.
[18] ——, “Attestation Identity Key (AIK) Certificate Enrollment Specification.Frequently Asked Questions,” Trusted Computing Group, Tech. Rep., 2011.
[19] J. Lyle and A. Martin, “On the feasibility of remote attestation for web services,” in Proceedings of Computational Science and Engineering, 2009. CSE’09 (Volume:3), 2009.
[20] A. Jøsang and B. AlFayyadh, “Robust WYSIWYS: A Method For Ensuring that What You See Is What You Sign.” in The Proceedings of the Australasian Information Security Conference (AISC2008), Wollongong, Australia, January 2008.