Improving Internet Trust and Security

George Staikos, KDE

March 15, 2006

Abstract

The rapid increase in the popularity of electronic commerce and online access to valuable and sensitive information in recent years has resulted in a sharp increase in *phishing* attacks and other exploitation of the security mechanisms in use on the World Wide Web. We believe that in order to address these security and trust threats, action needs to be taken by all stakeholders and that no single solution or technology is sufficient on its own.

1 Introduction and Position

The security model in use today in web browsers, mobile devices, e-mail clients, and other client-server network applications is still substantially the same one developed over ten years ago when electronic commerce was first evolving. It is based on a public key cryptography system using X.509 certificates issued by a *trusted* signing authority and an implementation of the SSL and TLS protocols. Not only are the public key infrastructure and SSL/TLS protocols relatively dated and in some cases flawed for this purpose, but the user-agents which implement them have wide ranging flaws which need to be addressed, primarily in terms of the ability of content to *spoof* security mechanisms and indicators. While we are not advocating a complete redesign of the security measures in use, we do feel that virtually every aspect needs to be examined and enhanced and that any weak point in the system is one too many. Moreover, the goal of increasing trust between two parties who do not have the benefit of a tangible or *real-world* interface is a very difficult problem and cannot necessarily be solved through purely technical means. It is possible that the lack of trust is the most significant inhibitor of e-commerce expansion today.

2 Encryption

The encryption algorithms available today for securing communications and transactions will be generally good enough for many years. The main concerns we have involve the use of older, weaker algorithms which are typically considered obsolete but still deployed in many legacy applications. It is very important that we work to completely deprecate and disable the use of weaker hashes and ciphers such as MD5 and DES. Short key lengths should also be avoided. While it is unrealistic for an individual to break even some of the weaker keys today due to cost of computing required, it is not so unrealistic that these algorithms will be cost-effective to break within a few years. This means that anything encrypted with weaker algorithms today, if archived, could become vulnerable in a relatively short time. Historical data could potentially have significant value. Unfortunately many e-commerce services and user-agents still default to relatively weak cryptography despite the availability of much stronger alternatives.

Perhaps even more significant than the increasing power of any given individual's computing power is the proliferation of *botnets*. A botnet is a collection of slave computers which have been co-opted for massive distributed or parallel computing tasks. They are typically formed without the knowledge of the owners of the computers and assembled via the spread of a virus or worm. Some botnets have been discovered with many thousands of slave nodes. The thread due to the rise of botnets is clear; while one individual may not possess the computing power required to break modern cryptography, an individual in control of a significant botnet most likely can. This could apply to the interception of communications as well as theft of an X.509 identity. If encryption standards are not improved to address this new threat, we might soon discover a co-ordinated attack so significant that it causes very wide-scale financial damage. End-user trust in current e-commerce technology could be irreparably damaged under such circumstances.

3 Identity

Securing a transaction requires both that the identity of the parties (or at least one party) can be verified and that the communication channel is secure. Cryptography is used for both of these tasks and while the implementation and deployment of the latter is fairly complete, the implementation and deployment of the former is not. Most transactions today are done via a single identity check of the *server* by the client and rarely is the identity of the client checked cryptographically. While a merchant may not care who pays for a widget as long as the money is received and the product is delivered, this apathy actually enables fraud and identity theft. Most client identity checks, if they exist at all, are done via a simple one-factor password authentication. This is a very poor indicator of identity and very easy to circumvent. It is our belief that the use of client-side certificates will significantly reduce the opportunity for identity theft, but that the current implementations are far too cumbersome and inconvenient to allow for widespread adoption. If the identity of both parties can be strongly established, then the ability to commit fraud should be significantly reduced.

3.1 Authentication

Two-factor authentication is considered to be a major step forward in user-authentication but in practice it is not widely deployed except in corporate environments and it is typically only used to validate user identity. Unfortunately many implementations are still cumbersome and impractical. Individuals don't have to carry electronic tokens with them in order to use their credit card at a physical store and they shouldn't need to carry such a device in order to complete a simple on-line transaction either. We need to reduce the user impact of two-factor authentication in order to make it more accessible to the average user and therefore drive deployment. To date, SSL client certificates which can, especially when combined with a smartcard, provide two-factor authentication have been a complete failure in terms of broad acceptance. We believe that this is almost entirely due to usability issues.

3.2 Signing Authorities

The technical ability to establish identity is only part of the problem. The identities we use today are linked to a cryptographic key which is considered too expensive or impossible for a malicious entity to compromise. A more subtle and challenging aspect of verifying an identity is ensuring that the entity possessing the cryptographic identity is in fact who they claim to be and that the identity was issued to that entity in a valid and precise manner. Some important criteria to consider are that the certificate was issued by an authority who is permitted to issue such an identity certificate, and to an entity who is permitted to obtain such an identity certificate. This is no simple problem. Present industry practice often involves simple verification that the applicant for a digital certificate is in control of the domain that they are requesting the certificate for, thereby *effectively basing the entire security of significant transactions on the known flawed DNS system.*

Certificate authorities should be required to update their current identity vetting procedures to more closely match those used in the physical world. A good example of this sort of vetting procedure might be the way that identities are vetted when financial transactions are executed. There is also an opportunity for a two-tiered approach since strong vetting is not necessarily required for all applications at this time. However, the fact that a major financial or government institution can get a strong identity indicator with a simple e-mail or phone call underlines the significant vulnerabilities of the existing system.

We believe that software vendors should strive for stronger signing authority guidelines and regulations, and enforce these regulations with frequency on the authorities. Software vendors should aim to form a clearing house of sorts where signing authority status and security incidents can be tracked and shared. This will decrease the time to address security incidents and improve trust database consistency across user-agents. Furthermore, user-agents should implement active updates of the certificate root database to enable timely revocation of rogue or compromised authorities.

4 User-Agent Interfaces

The user-agent (for example web browser or e-mail client) is the main interface that the user interacts with when engaging in an Internet based transaction. It is responsible for implementing the trust and security protocols and conveying the proper information to the user in an understandable way. In many respects the user-agents available today are doing a very poor job of this. The representation of the paradigms such as the padlock icon are relics of ten years ago; they're imprecise and easily simulated. User-agents need to improve the quality and significance of indicators as well as make strong distinctions between *chrome* and page content. Users need to be certain that the indicators they think they are seeing are generated by the browser and not by malicious content. Some recent initiatives in this area include prevention of chrome manipulation by site content such as removing toolbars, location indicator, and statusbar, and providing strong indicators on content generated dialogs and pop-up windows. Active content such as that generated by browser plugins remains a challenging issue. The ability for a user to easily distinguish the cryptographically verified state of identity of the content is critical to building trust and avoiding *phishing.* The reality is that current implementations fall far short of where they need to be.

5 Active Mitigation

We believe that active mitigation of *phishing* attacks can be a useful tool alongside PKI and software architectural fixes. Microsoft, Google, and other vendors have initiatives to actively track known phishing sites and provide realtime feedback and blocking mechanisms in the user-agent. This approach has proven successful in fighting unsolicited e-mail and we have reason to believe that it can be very successful in fighting phishing as well.

Another related active mitigation mechanism is OCSP. OCSP has existed for many years now but remains virtually undeployed. OCSP is critical to proper functioning of the public key cryptography system, allowing certificate authorities to revoke compromised or invalidly issued certificates in real time in order to reduce the impact of an incident. Certificate authorities should be required to implement OCSP and software vendors should implement and activate support for it as quickly as possible.

Finally there may be opportunity to introduce peer-verified identity confirmation. Following the PGP model in which a web of trust is formed by end-user experience and out-of-band verification of identity, we believe that there are opportunities to do similar things for secure web sites. A web of trust could be quite valuable as a secondary measure to verify how trustworthy an online entity may be. This would not replace the traditional CA model and vetting procedures.

6 Conclusions

In order to build trust in the system we need to build a system that users have full confidence in and are able to use without falling victim to fraud. The solution needs to cover the security of all aspects and parties involved. Any single weak link can lead to a complete loss of trust in the entire system. It also needs to provide a level of confidence and usability substantially similar to what exists in the *brick and mortar* world. The requirement to make a judgment call on content needs to be taken out of the hands of the user and dealt with by the user-agent instead, with strong, trustworthy, and easily understood indicators presented to the user. Finally, we need to make sure that identities are properly and thoroughly vetted, and that the status of identities are actively tracked and updated. If there is no trust in identity, then the entire foundation for online transactions will be in doubt.