



PRICEWATERHOUSECOOPERS 

CCE Journal

Cryptographic Centre of Excellence

Issue 6

beTRUSTedSM

An e-security business of
PRICEWATERHOUSECOOPERS 



© 2001 PricewaterhouseCoopers. PricewaterhouseCoopers refers to the individual member firms of the worldwide PricewaterhouseCoopers organisation. All rights reserved.



TRUE SECURITY
IS THE FREEDOM TO
LIVE DANGEROUSLY.

The Internet is now secure for big business. Using digital certificates and unprecedented standards, we can provide the security and privacy services your business needs to thrive. To find out how we put it all together, visit beTRUSTed.com

beTRUSTedSM

An e-security business of PricewaterhouseCoopers.

PRICewaterhouseCOOPERS 

Join us. Together we can change the world.

March 2002

CCE Journal

Cryptographic Centre of Excellence

The PricewaterhouseCoopers Cryptographic Centre of Excellence (CCE) was formed by the firm's Global Risk Management Solutions practice to unite members with unique expertise in cryptography and cryptographic services from around the globe. By establishing relationships with academic institutions, leading security vendors, cryptographic research organisations and leading cryptographers, we are in a truly unique position to offer our global clients the best solutions for their cryptographic security needs.

Our Global Risk Management Solutions (GRMS) practice is devoted exclusively to the critical business issues of security, privacy and compliance, operational effectiveness and management assurance. Through proven methodologies, best-of-breed tools and best practice services, GRMS helps organisations build and maintain a secure and high-performance business infrastructure. With more than 6,000 professionals located around the globe, GRMS represents the world's largest risk management practice.

The CCE was instrumental in the creation of beTRUSTed, an e-security business of PricewaterhouseCoopers. beTRUSTed (www.betrusted.com) delivers the most complete custom-designed infrastructure solutions to the Global 2000. beTRUSTed combines world-class Public Key Infrastructure expertise and Certification Authority services. Along with consulting and integration from PricewaterhouseCoopers, beTRUSTed rapidly advances an organisation's ability to conduct sensitive, high-value communications and transactions in a networked environment.

Contact Information

Dr. Alastair MacWillson

Partner in PricewaterhouseCoopers London and joint CEO of beTRUSTed within the PricewaterhouseCoopers Global Risk Management Solutions practice.

alastair.macwillson@uk.pwcglobal.com

Geoffrey C. Grabow CISSP, MSc IS

Americas Leader – PricewaterhouseCoopers Cryptographic Centre of Excellence.
Chief Scientist – beTRUSTed.

geoffrey.c.grabow@us.pwcglobal.com

John Velissarios M.Comp.Sci.

EMEA Leader – PricewaterhouseCoopers Cryptographic Centre of Excellence.
Strategy and Business Development - beTRUSTed.

john.velissarios@uk.pwcglobal.com

More information and previous editions of the CCE Journal can be found at:

www.pwcglobal.com/cce

The views expressed in this publication are not necessarily the views of PricewaterhouseCoopers.

To SUBSCRIBE to CipherText, our weekly e-mail cryptographic newsletter, go to <http://www.pwcglobal.com/cce>.

In this Issue

Editor's Soapbox **3**

by Geoffrey C. Grabow

Introducing Air Gap Technology **5**

by Joseph Steinberg, Director of Technical Services, Whale Communications

A Guide to Determining Return on Investment for E-Security **10**

by Derek Brink, RSA Security Inc

Electronic Signatures – A Short Summary **17**

by Marc Sel, PricewaterhouseCoopers

XML and Security **19**

by Mark O'Neill, CTO, Vordel Ltd

Smartcards – Consumer Non-Repudiation **24**

by Simon Ward, beTRUSTed

Building your appropriate Certificate-Based Trust Mechanism for Secure Communications **28**

by Kaijun Tan, PhD, Scientist, Rainbow Technologies, Inc

Upcoming Conferences **33**

Editor's Soapbox

Cross Certification vs. Certificate Stores

by Geoffrey C. Grabow CISSP

Users within a hierarchical PKI construction have the basic problem of being unable to verify the digital certificate of a user from another hierarchy. Several techniques can be used to resolve this problem, but lately cross-certification has been getting a lot of attention.

Cross-certification (fig 1) implements a mutual trust between the roots of different PKIs by signing each root's certificate with the Private Key of the other. That trust is meant to be a link between the two PKIs such that applications can navigate their way up one trust hierarchy across to the second, then down to the user with whom they wish to transact business.

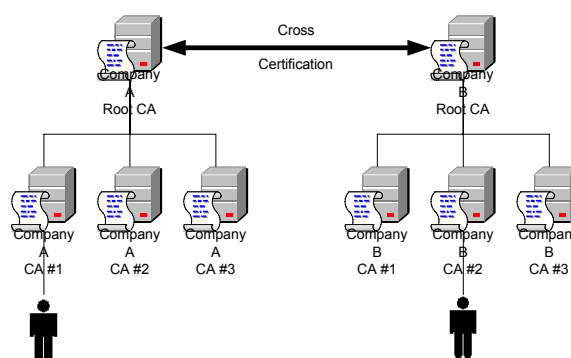


Figure 1

While this sounds good in theory, there are several problems with this trust model. Once cross-certified, the trust is absolute. There is no distinction between users in Company A from those in Company B. Users will trust employees of another company as if they were part of their own company. While this type of trust has its uses, such as when Company A merges with Company B, it is more common to need some form of granularity so it can be decided “who to trust”, “how much to trust them” and “for what transactions they can be trusted”.

Cross-certification today can only take place at the Root CA. It is more likely that there is only a portion of Company B that is to be trusted by Company A, such as when the two companies are collaborating on a project.

An alternate method for providing trust to different PKIs is to install into the application the certificates of only the CA's you wish to trust. This enables granular control over the trust model, as well as providing the ability to remove the trust linkage by simply deleting the CA's certificate from the application.

In this model (fig 2) it is inevitable that you end up with a collection of certificates and it is very convenient for applications if they are located

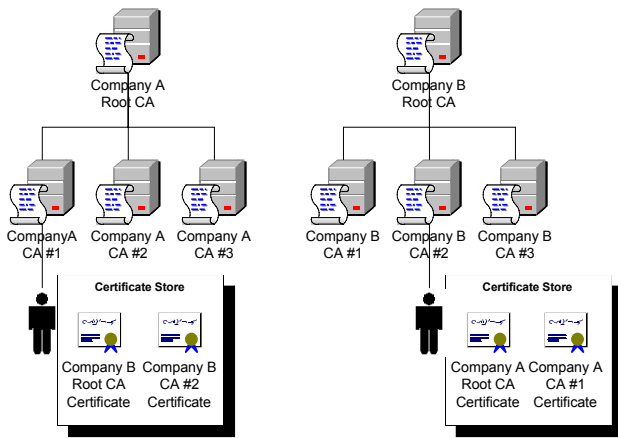


Figure 2

in a commonly accessible location. This has already been implemented in web browsers and several Operating Systems. There is a certificate store that contains all of the Root and Issuing CA certificates. If you decide that there is another company you wish to trust, you can simply add their certificate to your store (for an example of this, visit http://www.betrusted.com/products_services/ and click "CA Certificates"). If the trust relationship should change, you can just as easily remove the certificates from the store.

Even if cross-certification is used, the certificate store model will probably be required as you may be issued certificates from several different PKIs for use in different applications.

Of course the applications are the key to all of this. Few applications today understand how to negotiate the trust chain of cross-certified Root CAs, however many applications, such as email and web browsers, are already aware of the "certificate store" model. Increasingly the application to be used is a custom developed application designed specifically to address the needs of a particular business. It is these applications that must be designed with the proper understanding of the PKI trust model in which they are to perform.

Comments on this topic are welcome and may be submitted to the CCE Journal via e-mail using the contact information at the beginning of this issue.

About the author

Geoffrey is the Americas leader of the PricewaterhouseCoopers Cryptographic Centre of Excellence, co-editor of the CCE Journal and Chief Scientist of beTRUSTed.

He can be contacted via e-mail at geoffrey.c.grabow@us.pwcglobal.com

Introducing Air Gap Technology

*by Joseph Steinberg,
Director of Technical Services,
Whale Communications*

Security policies employed by corporations, organisations, and government entities often mandate that internal information-technology systems hosting proprietary and confidential information remain inaccessible to the 'outside world'. In the pre-Internet world, this seemingly obvious and straightforward requirement was implemented rapidly and inexpensively simply by ensuring that all sensitive internal systems were not connected to any outside networks. As entities updated their architectures to allow for participation in today's on-line economy, however, the aforementioned security policy has become exceedingly challenging and difficult to satisfy.

The advent of the commercialised Internet and on-line e-business forced (and continues to force) entities wishing to flourish, to design and implement methods for allowing the 'outside world' to communicate with sensitive internal systems. Retailers wishing to sell on-line, for example, need mechanisms to authenticate users, track inventory, manage customer information, and transact purchases – all of which require information from both the 'outside user' and the companies' internal systems. IT departments are faced with the dilemma of allowing communications between the outside untrusted world and sensitive internal systems and, at the same time, ensuring that the integration of external access with back-end systems does not become the breeding ground for security nightmares.

The recent expansion of the Service Provider (SP) market – encompassing application service providers, outsourced web-hosting providers, and other managed service providers – has introduced even greater significance and orders of magnitude to this conundrum. Numerous companies now host their valuable and sensitive information at SPs, and rely on these SPs to protect what may amount to the very lifeblood of their organisations. The level of information security that an SP offers to its clients may be a significant factor in a company's decision to utilise a particular SP, and even a single significant security breach at an SP could produce catastrophic consequences for that SP when seeking to develop new business. As such, while SPs must provide access for the world-at-large to a great number of both independent and interconnected systems, SPs must guarantee that the information hosted on their systems remains completely secure and free of any information 'leaks'. As SPs may host the data of competing firms, they must ensure that they shield their customers' data not only from attacks originating in the outside world,

but also against intrusion attempts emanating from the systems of other customers hosted within the same SP environment.

Historically, organisations mitigated the risk of internal systems being compromised by implementing security solutions utilising 'firewalls,' typically specialised 'gatekeeper' computers attached to both the outside world and to companies' internal networks that managed and regulated the information passed between the two sides. Firewalls were typically configured to block all inbound traffic (other than e-mail, which was handled differently) and, as such, effectively lowered the number of successful security breaches, and also provided employers with a method for controlling what types of content internal users could request from systems outside of their corporate networks.

Infrastructures advanced and more modern architectures utilised multiple firewalls, creating a so-called Demilitarised Zone (DMZ) between the internal and external networks (protected on each side by a firewall), where servers being accessed by the outside world would sit. The DMZ allowed organisations to invite large numbers of untrusted users to access systems and view marketing materials, etc, without ever compromising security by allowing external users onto the internal network.

Organisations began to feel secure and comfortable.

Unfortunately though, as time passed and business requirements for on-line commerce and other e-business activities required that systems in the DMZ access resources on the internal network, firewall technology proved insufficient to protect corporate informational assets. To allow DMZ-based systems to access internal applications and data stores, firewalls were configured to permit traffic to enter into back-end networks from the outside world, severely compromising the security offered by the firewall. Hackers learned to exploit the holes utilised for transmitting inbound data, and successfully compromised both firewalls and internal networks previously thought to have been secure. Recent high-profile security breaches at Microsoft, Egghead, the US military, and other well-established firms and government entities have underscored significant firewall deficiencies. Several well publicised incidents involving hackers pilfering credit card numbers from internal corporate systems (allegedly protected by firewalls) and then demanding financial compensation for not distributing

the stolen information en masse have clearly illustrated the potentially problematic consequences of firewall technology's failures.

Air Gap technology was invented specifically to address the security needs that firewalls do not, protecting corporate informational assets and back-end internal systems from the prying eyes and hands of outsiders in the age of e-business. Unlike firewall technology – whose effectiveness is seriously compromised once configured to permit the passing of inbound traffic – Air Gap technology succeeds in physically preventing hackers from accessing internal, sensitive networks and machines.

In this article I will explain what Air Gap technology is, how it works, why it offers superior protection of corporate informational assets than do firewalls alone, and what its limitations are.

What is Air Gap technology and how does it work?

In general terms, Air Gap technology offers total physical and logical disconnection of two or more networks, while simultaneously allowing information to be transferred securely between the two. This seeming contradiction has in fact been achieved manually for many years. Historically, Air Gap technology was first implemented as 'sneakernet' – in situations where security considerations mandated that two computers be disconnected, but information needed to be shared between the two. Using sneakernet a user would record data from one machine onto a magnetic disk or tape, walk over to the other machine, insert the disk or tape, and load the data onto the other network. In the era of nightly batch processing this solution provided sufficient communications capabilities while maintaining a comfortable level of security, and successfully shielded internal systems from hackers.

With the dawn of e-business, however, sneakernet was no longer a viable option. Real-time customer interaction, necessary for electronic commerce and other e-business activities, could not be achieved over sneakernet. As such an automated, high-speed, practical method of achieving communication between disconnected networks became necessary.

Air Gap technology delivers a similar degree of 'security through disconnection' as does the traditional sneakernet solution, albeit with automated data transfer and at a much greater speed. It also incorporates automated secure

content inspection which, in the sneakernet paradigm, would likely require even further manual intervention.

An Air Gap solution, by definition, maintains a physical disconnection (the so-called 'Air Gap') between the trusted internal network and the external world, while enabling selective secure data transfer between them. Implementations typically utilise some form of memory-containing switching device that toggles between networks – allowing only one network at a time to read or write to its memory. Data may be transmitted into the device by one network, and then after the device disconnects from that network and connects to the second, read by the second network. The process of receiving data, switching between networks, and transmitting the data out must be achieved efficiently and at a high-speed, to provide the seamless, transparent, high bandwidth, and low-latency transmission required in real-time environments.

In human terms, the Air Gap paradigm resembles the secure-transaction systems utilised at the drive-through windows of many banks; the customer and teller are physically separated by a window of bulletproof glass, yet are able to transact business. For example, the customer can give his withdrawal slip to the cashier by placing it in a metal drawer, which can be opened by only one of them at a time. The cashier can retrieve it, check that all materials needed to process the transaction are present, proper, and complete, process the transaction, and securely return the stamped withdrawal slip and currency in a similar manner.

Inherent in Air Gap technology is the lack of network and transport layer protocols in communications across the Air Gap between the two networks. If protocol-containing packets were simply shuttled across the Air Gap by the switching device, attacks could simply be shuttled into the internal network, and sensitive information shuttled out. In an Air Gap solution, however, network connectivity ends on both ends of the Air Gap; the switching device strips all headers and wrappers and transmits only the raw payload of data to the other network. In such a fashion, Air Gap technology mitigates exploits based on protocol weaknesses. In fact, because no network information ever traverses the Air Gap, an external user would have no way of

'fingerprinting or enumerating,' that is, mapping the addresses and contents of the internal network. Also, assuming that the switching device is a solid-state device and not a computer, operating-system weaknesses – long a favourite system-entry mechanism of hackers – cannot be exploited either.

Environments utilising Air Gap technology in their infrastructures also benefit from the ability to properly perform thorough application-level security checks by inspecting and analysing all inbound traffic on trusted machines within their internal networks. Content inspection of this nature yields many benefits including preventing the spread of computer viruses and terminating various types of web-server attacks based on malformed URLs and inappropriate data types. As data enters into the secure trusted network after traversing the Air Gap, it can be quarantined until it has successfully been subjected to rigorous security checks performed by servers which are securely protected by the Air Gap, and only after undergoing stringent inspection, is the data then forwarded on to the internal network. In the typical firewall paradigm, application-layer inspection (if performed at all) would likely be done in the DMZ where machines serving the external world sit, and where security-enforcement mechanisms could be compromised.

Also, because Air Gap technology breaks open data packets and ignores protocol headers during traversal across the Air Gap, and only later reconstitutes network traffic on the other side, Air Gap technology can allow an organisation to utilise different protocols for external access than internal access without modifying any internal servers. For example, an organisation may offer information over its intranet via insecure http without demanding credentials for access, whereas it would like to offer the same information over the Internet (for employees travelling, business partners, etc) but only over secure https and only when a user provides appropriate credentials. Air Gap technology makes implementing such a solution trivial. Since the http application-level protocols are broken before transmission of the internal information across the Air Gap to the outside world, the http data can simply be repackaged as https, all the while retaining the company's private key on the internal trusted side of the Air Gap, and maintaining total network disconnection.

An Air Gap solution, by definition, maintains a physical disconnection (the so-called 'Air Gap') between the trusted internal network and the external world, while enabling selective secure data transfer between them.

This decomposition and reconstitution of network packets in an Air Gap solution provides additional benefits as well. For instance, the process of breaking and rebuilding packets guarantees that the only method by which external data can travel on the internal network is within packets generated by a trusted server on the internal side of the Air Gap. Also, the target destination for the internal packets need not be based on the user-specified address in the original external packets as would be the case in firewall implementations. Air Gap solutions can be configured to route all inbound data to specific pre-defined machines, and completely ignore all destination information contained in the original network transmission. Air Gap technology can thereby eliminate issues of hackers utilising various exploits to force data to reach inappropriate target machines.

Proposed alternatives to Air Gap technology have included various methods of utilising non-standard protocols for transmission across connected networks. Although, at first glance, these solutions seem to make the task of hacking into the internal network difficult, without the hardware disconnect provided by an Air Gap solution, the infrastructures created by implementing these proposals may generate the same risk inherent in all other 'security by obscurity' solutions, that is, if the proprietary protocol were ever to be exposed, the solution could be seriously compromised and grave security concerns could potentially arise. Air Gap technology, on the other hand, is designed to ensure that even if a hacker knows exactly what solution has been implemented, he/she still cannot compromise the protected network.

Air Gap solutions account for the fact that machines on the outside of the Air Gap are not trusted. As such all configuration, management, data inspection, and encryption are performed on the trusted side of the Air Gap. This is a major benefit over traditional firewall architectures that require DMZ-based facilities for performing these necessarily internal functions. As large numbers of unknown users directly access the DMZ, the DMZ must be considered untrusted, and security functions should obviously not be performed and enforced by systems that are themselves vulnerable to compromise and manipulation.

Another essential element in an Air Gap solution is the ability to physically limit the flow of data to one direction. A company may wish to distribute specific information housed on its internal systems, without allowing any external access to those systems. Another

organisation may desire to allow information to be submitted to its internal systems from the outside world, without anyone external to the company being able to extract anything from the internal network. Air Gap solutions should provide methods for implementing such guaranteed-one-way transfers across the Air Gap.

Obviously, as integral components in corporate IT infrastructures, Air Gap solutions must be able to sustain physical component breakdowns without themselves failing, and as such, should be designed to operate in high-availability, redundant formats.

Why is Air Gap technology superior to firewalls in protecting externally-accessible internal systems?

Air Gap technology offers numerous advantages over firewalls in protecting externally-accessible sensitive internal systems from hostile inbound traffic. As alluded to earlier, firewalls were intended to guard against attacks by blocking external access to internal resources and, as such, they perform admirably. However, they suffer from numerous inherent weaknesses when utilised to selectively permit traffic into a network. Air Gap technology successfully addresses those shortcomings, and provides proper protection for corporate informational assets and systems. Some examples of firewall deficiencies that Air Gap technology addresses are:

1. Firewalls are typically computers and, therefore, run operating systems, which are notorious for having flaws that provide for security exploits. An Air Gap solution utilises a 'dumb' switching device with no operating system.
2. Firewall machines are simultaneously connected to both trusted and untrusted networks providing for a direct path from the outside world to sensitive systems in the event that the firewall were ever compromised. Network traffic – with headers, wrappers, etc – flows from the outside world to the internal systems. Air Gap solutions ensure that two networks are never connected, that at no point in time is there a direct path from a machine on one side to a machine on the other, and that no network traffic ever flows between the two sides of the Air Gap.
3. Firewall-solution architectures typically require that servers in the DMZ store sensitive information – such as user names, password, corporate private keys, etc. This information belongs on internal systems

where there is a greater level of security. Air Gap solutions allow all sensitive information to be stored on the internal side of the Air Gap, where it is properly protected.

4. Firewalls that provide content inspection perform this function on a machine connected to the outside, untrusted network where, in the event of a compromised firewall, inspection policies and engines can be altered, disabled, or removed. Air Gap solutions assume that all of the machines on the untrusted side are insecure and, therefore, all content inspection is performed on machines on the internal side of the Air Gap, where the security-enforcement mechanisms are adequately shielded from outside attacks.
5. Firewalls are multi-function products utilised for numerous security functions, and as such, often lack the great degree of granularity offered by Air Gap solutions designed specifically to protect sensitive systems from inbound data flows.
6. Properly configuring a firewall can be a complex and arduous task and, as a result, misconfigurations, which allow for security breaches, are not uncommon. Air Gap solutions are quite simple to configure, and human errors leading to negative security consequences are less likely to occur.

Limitations of Air Gap technology

Two important observations about the limitations of Air Gap technology are as follows:

Firstly, although Air Gap technology quite successfully protects sensitive corporate systems from the hostile behaviours of unknown external hackers, it is not normally intended to shield traffic generated by internal users accessing external systems (ie, problems created by outbound traffic). For that reason, in an enterprise-level architecture, Air Gap technology should typically be used in conjunction with a firewall – with the Air Gap system protecting back-end applications and data stores from inbound attacks, and the firewall managing internal users' traffic to the Internet. Also, firewalls used in conjunction with Air Gap technology can help prevent denial-of-service attacks intended to overwhelm the system managing the Air Gap. Firewalls were originally designed to be used for this purpose, and with relatively simple configurations, provide an adequate solution for this need.

Secondly, it is essential that one realise that for an Air Gap solution to successfully protect internal systems, the internal networks must themselves be trustworthy and secure. As employees, other insiders, and remote-access/dial-up connections pose potentially significant security risks, it is recommended that sensitive internal applications be maintained on a separate network from the general user population. This is true whether or not Air Gap technology is utilised in the organisation.

Conclusion

Today's business needs demand a solid infrastructure for sharing large volumes of information at high speeds. At the same time, security considerations emanating from the mechanisms utilised to achieve adequate communications are growing increasingly paramount. Air Gap technology allows organisations to have the best of both worlds, by providing a high-throughput channel from the outside world to internal systems, while simultaneously removing all external network connections to sensitive internal systems. Air Gap technology allows organisations to communicate effectively while drastically reducing their risk of a security breach.

About the author

Joseph Steinberg is Director of Technical Services at Whale Communications.

He can be reached at joseph@whale-com.com

A Guide to Determining Return on Investment for E-Security

by Derek Brink, RSA Security Inc

This paper is not about technology; it's about time and money. That is, organisations often ask for help with not only the technology case, but also the business case, for their investments in Public Key Infrastructure. In other words, what is the Return on Investment ('ROI') for PKI?

This is not always an easy question to answer – PKI is an e-security *infrastructure*, after all, and the ROI for infrastructure of any kind can be extremely difficult to quantify. Some don't try, and have implemented based more or less on a leap of faith. At some point, however, we can observe that the ROI for infrastructure often becomes *unnecessary* to quantify, because the capabilities it enables are both mission-critical and well understood. For example, when is the last time any large business required a return on investment analysis to determine whether or not it should invest in enabling infrastructure such as telephones, facsimile machines, or e-mail? This paper is developed from the present perspective that ROI for PKI is somewhere between too difficult and not necessary, somewhere between a leap of faith and a matter of course.

The objectives of this paper are to provide a reasonably fine-grained framework for the 'Return' component of the PKI ROI equation, to advance the level of practical detail in discussions about the business case for PKI, and to generate specific ideas for PKI ROI analysis. It is a non-objective – nor is it possible, given the innumerable e-business processes that can potentially leverage PKI as their e-security foundation – to provide a single set of formulas or templates into which one can simply plug numbers and compute 'the answer'.

Financial Returns: the 'R' in ROI

As PKI becomes more widely deployed, and as more hands-on experience makes the total cost of ownership for PKI more accurately understood, we can turn our attention to the topic that generates the most enthusiasm in the corner offices: the *financial returns* made possible from PKI-enabled business processes.

What financial returns does public key infrastructure really provide? Here, we provide a general framework for unlocking the financial returns that are made possible by implementing PKI-enabled applications. In considering this framework, the following simple, step-by-step approach should be kept in mind:

- **Focus on the Business Process**
It's worth repeating that PKI is an e-security

infrastructure, and infrastructure in the absence of a specific business process returns nothing. For example, if we invest in telephones, facsimile machines, and e-mail systems but never place a call, transmit a document, or send a message, what have we gained? Moreover, returns from e-security infrastructure are generally difficult, if not impossible, to separate from the returns from the business processes themselves. The primary focus – once it has been determined that authentication, data privacy, data integrity, digital signature, or other e-security capabilities provided by PKI are important business requirements – should therefore be on the financial returns from the successful implementation of a particular (security-enabled) business process. This approach also accommodates the reality that financial returns are typically application-specific, company-specific, industry-specific, and so on.

- **Establish Appropriate Metrics**
With a proper focus on security-enabled business process, the next step is to establish the appropriate *metrics* for determining potential financial returns. The metrics chosen will logically be a function of not only the particular business process under analysis (ie, is it an internal process? A customer-facing process? A partner-facing process?), but also the specific business objectives we have in mind (ie, are we aiming to increase revenues? Lower costs? Improve efficiency?). A subsequent section ('Metrics') discusses this topic in more detail.
- **Establish a Baseline for the Current State**
Having established an appropriate set of metrics, the next step is to use them to establish a baseline for the business process under analysis, based on the way things are today. This is the 'business as usual' scenario.
- **Compare to the Desired Future State**
The same metrics can then be used to compute the financial impact of implementing a new or improved business process that meets the specific business objectives we have in mind. This is the 'business as a result of' scenario, ie, the desired future state that will result from the successful implementation of a new or improved PKI-enabled business process.

If this straightforward approach sounds familiar, it should come as no surprise – it's a time-honoured method for establishing value, a process we've all gone through (consciously or otherwise) countless times before. We can step

back and observe that PKI is not uniquely complex or difficult to analyse in this regard – on the contrary, this approach for computing financial returns for PKI-enabled applications is the same one used for virtually any other significant investment. All we need, given the relatively early stage of PKI market development, is a general framework to help organise the attack and jump-start a detailed discussion of potential financial returns.

The first, critical step is to frame the ROI discussion in the context of the key e-security enablers for a particular e-business process/application. The next step is to establish an appropriate set of metrics for determining potential financial returns.

Metrics

The most appropriate metrics are a function of both the business process under analysis and one or more specific business objectives. Table 1 lists a number of potential metrics for certain example business objectives, and provides examples of 'impact statements' in the form of questions that set up a comparison of the current state with the desired future state in terms of one or more specific metrics. Quantifying the answers to these questions is the key to unlocking the financial returns made possible by PKI-enabled applications.

Based on a number of case examples, we observe that quantifiable financial returns made possible by PKI-enabled applications tend to fall into one of the following four high-level categories: *Revenues*, *Costs*, *Compliance* and *Risks*. The remaining sections of this paper explore these four categories in more detail, and include several examples of metrics that lead to quantifiable financial results.

Revenues

Business processes that generate new or increased revenue streams create perhaps the most compelling justifications for investments in enabling infrastructure such as PKI. Because revenue enhancements are generally more strategic than tactical in nature, however, they can also be somewhat more difficult to quantify.

Based on metrics such as those found in Table 1, we can reasonably quantify any number of incremental revenue streams for PKI-enabled applications. For example, suppose two-thirds of our on-line customers currently end up abandoning transactions that require them to print, sign and mail paper documents rather than allow them to complete the entire transaction on-line. What would it mean in

Business Process	Example Business Objectives	Potential Metrics	Example Impact Statements (The Key to Unlocking Financial Returns)
Customer-Facing	Maximize on-line revenues from existing customers	<ul style="list-style-type: none"> • % of revenue generated on-line • % of existing customers doing business on-line • % of customer wallet spent on-line • % dropoff rate • Repeat business rates • % of up-sell, cross-sell conversions • Lifetime revenue per customer 	“Two-thirds of our on-line customers don't complete transactions that require them to print, sign and mail paper documents. What would the financial impact be if we could reduce this dropoff rate to one-third by using digital signatures to complete the entire transaction on-line, as well as eliminate the cost of paper, printing, postage, and processing?”
	Minimize costs of finding and acquiring new customers	<ul style="list-style-type: none"> • % of new customers acquired on-line • Cost of new customer acquisition • Brand perception, brand awareness 	“What would the financial impact be if we could leverage 50% of all established on-line account relationships with Line of Business #1 to create an on-line account relationship with Line of Business #2?”
	Maximize customer satisfaction; reduce help desk and support costs	<ul style="list-style-type: none"> • # of incorrect order incidents • Service levels used • # of service/help desk requests • % of service/help desk requests resolved on-line 	“What would the financial impact be if authorized customers could resolve 80% of help desk calls directly, on-line, rather than by live agents over a toll-free number?”
Internal	Increase responsiveness to changing market conditions	<ul style="list-style-type: none"> • Order cycle/delivery time • Product time-to-market • Product time-to-change 	“What would the financial impact be if we could reduce our process cycle time from X days to Y hours, while preserving the integrity and authenticity of documents and transactions?”
	Reduce costs, improve productivity	<ul style="list-style-type: none"> • Cost of materials • Cost of services • Productivity per employee • # of service/help desk requests • % of service/help desk requests resolved on-line 	“What would the financial impact be if we could improve employee productivity and eliminate help desk calls caused by password resets, by using PKI-based authentication with our Virtual Private Network or with our Reduced Sign-On initiative?”
Partner-Facing	Tighten degree of system integration with strategic Partners	<ul style="list-style-type: none"> • % of production goods procured on-line • % of maintenance/repairs/operating supplies procured on-line 	“What would the financial impact be if we could shorten delivery times and reduce inventory, by enabling authorised users to procure 80% of all maintenance, repairs and operating supplies through a Web browser, mobile phone, or wireless personal digital assistant?”
	Reduce partnership costs, Improve partner reliability	<ul style="list-style-type: none"> • Comparative prices • Cost/uptime of partner connections • Cost/rate of partner repairs, replacements, returns • Cost, time commitment scorecard 	“What would the financial impact be if we could provide authorised strategic partners with increased access to sensitive information, without compromising security or giving up control?”

Table 1

terms of incremental revenue if we could substantially reduce this drop-off rate, say to only one-third, by using digital signatures to complete the transaction immediately while simultaneously minimising the risk of subsequent repudiation? For many document-intensive industries (including financial services, insurance, healthcare, etc) this would have an enormous impact on revenues – not to mention the potential for reducing the related costs associated with paper, printing, postage, and processing of traditional paper forms.

Other possibilities for quantifiable revenue-based financial returns include cross-selling or up-selling opportunities with established customers, an increased number of transactions per customer, higher rates of repeat business, etc. Important but less quantifiable examples in this category might include competitive advantage, strategic positioning, corporate brand/image, etc. A transactional Financial Services example is provided below.

Example: On-Line Brokerage Transactions

Organisation: On-line brokerage firm servicing self-directed individual investors.

Application: Instant Account Opening – on-line, paperless process to open an on-line brokerage account and fund the account electronically using the ACH mechanism.

Business Benefits: Time to open a new account reduced from 3-10 days to less than 3 minutes, a critical factor in accelerating revenues from new account growth and from converting prospects more quickly to active traders – an impact of tens of millions of dollars. Cost avoidance compared to manual account processing and help desk calls related to new account openings; cost reductions from reduced mailing and storage costs – an impact of over \$2m.

Benefits of PKI: Account activity acknowledged and authorised by electronic signatures; reduced risk from stronger user authentication; higher integrity of stored customer data.

Costs

Reductions in cost are perhaps the most reliable driver of financial returns for PKI-enabled applications – ie, although cost reductions are generally more tactical than strategic in nature, they are also generally the easiest returns to quantify (hence their popularity). Cost-based financial returns are typically expressed as some combination of the following:

- **Cost Savings:** ie, the new or improved business process is less expensive; we can spend fewer dollars than we did before.
- **Cost Avoidance:** ie, the new or improved business process scales to higher levels; we can avoid spending as many additional dollars in support of new capabilities or expanded scale.
- **Efficiency:** ie, the new or improved business process saves time; we can increase the velocity at which we conduct e-business.
- **Effectiveness:** ie, the new or improved business process increases productivity; we can do more or different things with the resources we already have.

While it is impossible to generalise about the best sources for cost-based financial returns, at present there are three areas that seem to be particularly fruitful: help desk costs, telecommunications costs, and costs associated with the processing of electronic forms/ electronic records.

The numbers in Table 2 illustrate why so many companies target the help desk as a rich and easy source of cost-based financial returns – end-users can usually experience faster, more convenient service at a reduction in cost of up to two orders of magnitude. Common PKI-enabled applications that can obtain substantial leverage from reductions in help desk costs include corporate Intranets, Reduced Sign-On initiatives, Virtual Private Networks, and one-to-many Extranets. A Secure Extranet case study example is provided below.

Type of Customer Service	Average Cost/Transaction
Agent (Phone-based)	\$5.00
Agent (Web chat)	\$2.50
Agent (E-mail)	\$2.25
E-mail (Auto Reply)	\$0.75
Web (Self-Service)	\$0.05

Table 2: Example Cost Reduction Target – Help Desk

Example: Secure Extranet

Organisation: Mutual funds, trust and investment services company.

Application: Secure Extranet for 5,000 independent financial advisors. 7 x 24, self-service access to high-value financial and client information.

Business Benefits: Annual cost savings of approximately 40% compared to phone-based, agent-based system. Largest driver for cost savings is 3x reduction in toll calls and direct agent assistance compared to previous process.

Benefits of PKI: Privacy and integrity of data; authentication of users; user accountability to data; customised content; reduced risk of data loss / theft; centralised control of trust policies and parameters.

Telecommunications costs also represent low-hanging fruit for cost-based financial returns, and are often used in particular to justify investments in Virtual Private Networks. Many organisations implementing VPN technology overlook authentication as a critical e-security requirement, however, on the mistaken assumption that an encrypted communications channel has fully addressed the problem of secure remote communications. Replacing a VPN's weak password-based authentication with stronger authentication technology such as PKI not only improves overall security (by more strongly establishing who's on the other end of your VPN), but also takes aim at a major source of help desk costs (according to some studies, up to 60% of help desk costs are related to lost or forgotten passwords).

A third area ripe for harvesting cost-based financial returns has to do with the cost of processing paper forms, documents and business records. This is most relevant in document-intensive industries such as financial services, insurance, and healthcare, where enormous financial returns are possible from cost reductions in the 'Four P's' of paper, printing, postage, and processing.

The cost of manual document processing is very high: the average paper document is copied 9-11 times at a cost of approximately \$18 and filed at a cost of approximately \$20, plus the additional cost of storage, electronic media, physical plant, postage and other distribution. And mistakes are expensive: the

cost of finding and retrieving misfiled paper documents is approximately \$120. Of course there are other business benefits to electronic forms processing in addition to lower cost, including wider, easier access; better quality; higher data integrity; the ability to avoid cost by containing growth in headcount; etc.

As an illustration of the magnitude of financial returns of this type, Table 3 compares the average distribution cost of Internet-based channels with that of traditional channels for term life insurance, bill payment, and banking, respectively. An Electronic Mortgage case study example is also provided below.

	Traditional Distribution	Internet-based Distribution
Term Life Insurance	\$5.50	\$2.75
Bill Payment	\$2.75	\$0.75
Banking	\$1.08	\$0.13

Table 3: Example Cost Reduction Target – The Four P's of Forms/Document Processing

Example: Electronic Mortgage Transaction

Organisation: Home mortgage services.

Application: On-line mortgage transaction.

Business Benefits: 30-45 day cycle time reduced to 5 hours. Reduced risk of mishandled documents, errors and omissions. Reduction in administrative staff, training costs. Improved customer service. Savings of approximately 20% in total loan lifecycle costs compared to previous process.

Benefits of PKI: Provable chain of evidence as to the authenticity of documents; authorisation to access documents based on user authentication.

Compliance

By compliance, we mean some business process that we are required to implement, or some e-security requirement that we are obligated to meet. Compliance generally refers to things about which we have very little choice, ie, things we must do in order to stay in business as we know it. In some cases, compliance may be related to cost avoidance (eg, avoid a fine); in others, it may be related to protecting an existing revenue stream. In any event, compliance-based business cases tend to

be somewhat binary: above a certain threshold, we just do it. As it relates to e-security infrastructure, compliance-based arguments tend to come from one of the following four categories: Regulatory, Partner, Customer, and Competitive.

- Regulatory compliance: where failure to implement could mean fines, loss of revenues, jail terms, etc, eg HIPAA regulations for the US healthcare industry, the Gramm-Leach-Bliley bill for the US financial services industry, etc.
- Partner compliance: where failure to implement could mean losing our ability to participate with a key partner or group of partners, eg a segment of the financial industry moving to the Identrus model for cross-certification.
- Customer compliance: where failure to implement could mean the loss of a business relationship with a key account, eg 'all General Motors suppliers who wish to have their contracts renewed must implement technology X by a certain date.'
- Competitive compliance: where failure to implement could mean the loss of competitive advantage and likely revenue loss, eg 'our competitors are eating our lunch!'

Compliance-based business cases tend to be made not so much on the basis of precisely quantified financial returns, but on the basis of 'the cost of doing business' or as a means to avoid 'what will happen if we don't implement.'

Risks

Until only recently, risk-based arguments were probably the most frequently used approach to justify investments in e-security infrastructure. Marketing campaigns and business cases alike were commonly based on arguments of fear, uncertainty and doubt. Selling security through fear can be reasonably effective, up to a point – for example, the big bad wolf certainly sold fairy tales in volume for the Brothers Grimm – but it also tends to marginalise e-security as an operating expense, subject to being trimmed at the first round of budget cuts. Today, happily, there is beginning to be significantly less emphasis on FUD and more on the systematic management of risk.

Risk is an inescapable fact of e-business, and there are only four things we can do about it: accept it; ignore it (which is the same as

accepting it); assign it to someone else; or mitigate it. Investments in e-security infrastructure that are made with prevention in mind are usually not all that visible (unless there's a problem), which tends to make risk-based justifications the least glamorous of the four categories in our model.

It seems obvious, but risk mitigation investments should be focused on things that are worth protecting, such as high-value information and high-value transactions. For examples of 'high-value' information, consider the following:

- Information that generates revenue, either directly or indirectly: eg information, programs, services, etc.
- Information essential to the smooth running of the company: eg operational information, administrative information, etc.
- Information pertaining to future revenue streams: eg research, new product plans, marketing plans, customer databases, etc.
- Information that must be protected by law: eg personnel records, student records, patient records, etc.

Once high-value information has been identified, we can then make a reasonable attempt to quantify the impact of various security-related risk scenarios, using the familiar 'impact statement' approach. For example:

- Productivity loss: eg what would the financial impact be if a security breach caused a sustained disruption of internal processes and communications? If we lost the ability to communicate with customers? (Keep in mind that 99.5% uptime still translates to 3.6 hours of downtime per month.)
- Monetary loss: eg what would the financial impact be if there were a security-related corruption of our accounting system which led to delays in shipping and billing? If there were a diversion of funds? What would be the expense of recovery and emergency response?
- Indirect loss: eg what would the financial impact be if a security breach caused the loss of potential sales? The loss of competitive advantage? The impact of negative publicity? The loss of goodwill and trust? (Indirect losses are among the most difficult to quantify but also among the most compelling in the risk-mitigation category, especially for businesses built on the fundamental foundation of 'trust'.)

- Legal exposure: eg What would the financial impact be due to failure to meet contractual milestones? Due to failure to meet statutory regulations for the privacy of data? Due to illegal user or intruder activity on company systems? (Your corporate counsel can potentially be an excellent source of justification for PKI-enabled business process.)

The answers to these risk-oriented impact statements can be difficult to quantify, but the financial implications can be extraordinary. And the risks themselves are very real – it seems that not a month goes by without a highly publicised security breach, and undoubtedly the vast majority of security breaches go unpublicised. The annual FBI/Computer Security Institute survey on computer crime and security shows that over 80% of respondents now answer ‘yes’ or ‘don’t know’ (which is probably the same as ‘yes’) to the question ‘have you experienced some kind of unauthorised use of your computer systems in the previous year’; unauthorised access by insiders is twice as frequent as unauthorised access by outsiders, and growing; and the Internet has rapidly replaced internal systems and remote dial-up as the most frequent point of attack.

Financial Returns: Summary

The most important points for developing meaningful financial returns for PKI-enabled applications are to focus on the business process, establish appropriate metrics, and look for all relevant returns in the following high-level categories: Revenues, Costs, Compliance, and Risks.

As we have seen in the example metrics and impact statements provided in Table 1, by properly framing the ROI discussion in the context of the key e-security enablers for a particular e-business process, we can very quickly begin to quantify financial returns using a straightforward, widely accepted approach. In general, we believe that the benefits from PKI-enabled applications significantly outweigh the costs of PKI implementation. Yes, Virginia, there is a strong ROI for PKI.

As we said at the beginning, this is not about technology; it’s about time and money. To put things in perspective, consider the parallels between current thinking about e-security infrastructure and the thoughts about various quality initiatives in manufacturing (Just-In-Time manufacturing, Total Quality Management programs, etc) in the 1980s. A common

business issue for pragmatic, non-technical executives at that time was the ‘Cost of Quality’, as in “Sure, these quality programs sound great, but how much will they really cost, and will there really be a return on my investment?” Then a provocatively titled little book – *Quality is Free* – helped business people to better understand and quantify the financial effects of poor quality: scrap, rework, longer cycle times, product returns, poor word of mouth, higher customer support costs, etc. So the phrase “Quality is Free” was really a concise, provocative summation of the concept that the cost of implementing quality programs was significantly less than the financial returns made possible by producing high quality products in the first place.

And so it is with e-security: the total cost of ownership for implementing enabling e-security infrastructure such as PKI is significantly less than the financial returns made possible by PKI-enabled applications. In other words, “e-Security is Free”. Plus ça change, plus c’est la même chose.

About the author

Derek E. Brink is Director of Product Marketing at RSA Security. He also directs the RSA Security Customer Advisory Council and is currently the Chair of the PKI Forum Executive Board.

He can be reached by e-mail at dbrink@rsasecurity.com

Electronic Signatures – A Short Summary

by *Marc Sel,*
PricewaterhouseCoopers

In 1977 Rivest, Shamir and Adleman made their discovery public that a simple mathematical function could actually be used to construct a practical system for (what they called at that time) digital signatures. Their system was based on Integer Factoring.

Such a system would use data, the RSA algorithm, a private key and a public key. The private key is used to create the signature, and the public key is used to verify the signature (refer to Technical Note #1 below for the distinction between signatures with appendix and with message recovery). A hash function is often used to be able to sign a short representation of the data, rather than the full-length original data. As such, the creation and standardisation of hash functions is fundamental to digital signatures.

Over the years, other digital signature systems were created besides RSA, such as those based on Discrete Logarithm (eg the Digital Signature Algorithm) and on Elliptic Curves.

Originally, the RSA company defined basic standards for encrypting and signing in their PKCS (Public Key Cryptographic Standards) series. Their PKCS #7 document became a de-facto standard for cryptographic messages and it is still in use today.

However, over the years a number of standardisation initiatives led to a wide range of standards, including amongst the most influential ones:

- **ISO/IEC 9796 (1991):** This specifies a digital signature mechanism based on the RSA public key technique and a specifically designed redundancy function;
- **ISO/IEC 9796-2 (1997):** This specifies digital signature mechanisms with partial message recovery that are also based on the RSA technique but make use of a hash-function.
- **ISO/IEC CD 9796-4:** Discrete logarithm based mechanisms.

Other standards include ISO/IEC FCD 14888-1, -2 and -3, as well as ISO/IEC WD 15946-2 (ECC). Also the ANS (American National Standards body) issues a number of digital signature standards, mainly for use by the Financial Services industry.

Underlying hash functions include MD2, MD4, MD5, SHA, SHA-1, RIPEMD, and RIPEMD-160.

In Europe, bodies such as the CEN (Comité Européen de Normalisation) work on taking over existing standards into a European context.

This includes the EESSI (European Electronic Signature Standardisation Initiative) project.

For telephony, ETSI (European Telecommunications Standards Institute) established a number of standards, including those for securing GSM (Groupe Spécial Mobile) and DECT (Digital European Cordless Telephone) systems.

In an Internet context, it is important to mention the RFC (Request for Comment) documents, which reflect the Internet adagio of 'rough consensus and running code'. As such, they do not establish standards, but such RFC's sometimes spread already well-known algorithms to a wider audience, or they do present a new solution. RFC documents often represent a somewhat American view to problems and solutions, which is not necessarily shared by European experts. However, in an Internet-centric society, they obviously cannot be ignored.

Today in Europe, we have the European Directive 1999/93/EC defining the Community framework for electronic signatures. In this context, we prefer to use the term 'electronic signatures' (as opposed to 'digital' signatures) to indicate that various electronic technologies need to be considered (biometrics, smart pens, ...).

The Directive specifies 'advanced electronic signatures' that are created by 'secure signature-creation devices'. Such signatures are verified on the basis of public keys residing in 'qualified certificates' provided by 'qualified certification providers'. Such signatures will satisfy legal requirements in the same manner as hand-written signatures do. It is clear that a PKI-based solution can meet the requirements for an 'advanced electronic signature' as laid down in the Directive.

The Directive does not detail which technical standards are required. However, in the context of the CEN, work is done on the EESSI. Here a set of technical standards is proposed to form the foundation with regard to underlying algorithms and data formats for use in Europe.

This set is deliberately kept fairly rich, in order to allow systems to be built which meet stringent security requirements for a number of specific cases. Signature verification techniques used for road pricing are subject to totally different constraints as those in the context of eg Internet banking. However, as a consequence, the designer (and to a certain extent the user) of the system should be aware of all possibilities at his disposal.

The commonly uttered phrase 'the good thing about standards is that there are so many to choose from' certainly applies to electronic signatures. For systems to be secure, well-performing and inter-operable, the selection of appropriate signature standards is critical.

Technical Note # 1

Essentially, there are two classes of signatures, 'with appendix' or 'with message recovery'.

'With appendix' refers to a separate signature file, created by the algorithm when providing the private key and the data as input. However, the original data is first transformed into a short hash value, which is encrypted rather than the full-length data. The encrypted hash functions as a signature and is sent as appendix to the original message. The relying party will use the original data, the signature file, the algorithm and the public key to perform verification.

'With message recovery' refers to the fact that the full data is formatted and encrypted, and successful recovery of the original data (complemented by some redundant information) is considered as verification of the signature.

Signatures 'with appendix' are more common.

Technical Note # 2

For those interested, the original publication of the RSA algorithm was in the article 'A method for obtaining digital signatures and public key cryptosystems', Communications of the ACM, 21, (1978), 120-126.

An excellent source of information on cryptography and digital signatures is the 'Handbook of Applied Cryptography' by Menezes, van Oorschot and Vanstone (1997 – CRC Press LLC).

Further information with regard to electronic signatures and cryptography can be found at the website of PricewaterhouseCoopers' Cryptographic Centre of Excellence (www.pwcglobal.com/cce) or at the website of our Trusted Third Party, www.betrusted.com

About the author

Marc Sel is a director of PricewaterhouseCoopers in Brussels, Belgium.

He can be reached by e-mail at marc.sel@be.pwcglobal.com

XML and Security

by Mark O'Neill, CTO, Vordel Ltd

As XML becomes the de-facto format for businesses to communicate over the Internet, so the need for security comes to the fore. Digital security has always been about the compromise between convenience and peace-of-mind. This holds true for XML also. The proposed advantages of XML for digital commerce – the opening-up of internal systems to trading partners via commonly agreed standards – are also concerns for security. These security concerns are now being addressed by a number of industry initiatives. This article describes a selection of these initiatives – the W3C's recent XML Signature specification and its relationship to SOAP, The OASIS SAML (Security Assertions Markup Language) initiative, and XKMS (XML Key Management Specification). Together, these initiatives are setting in place the infrastructure that will allow XML to travel safely between enterprises.

A bit of history

A common thread in the debate about XML and security has focused on whether to put the security layer within the XML document or not. Some of the early non-XML B2B integration frameworks, such as OBI (Open Buying on the Internet) which began in June 1997, incorporated X.509 and digital certificates and digital signatures at field-level into their document sets. Then the early XML-based B2B integration frameworks such as Open Trading Protocol (OTP) followed suit with security-specific tags. At this point opinion shifted, and it was thought best not to mix the XML data payload up with security and authentication information. As the HTTP POST protocol became the commonly accepted method of transmitting XML, it was felt that SSL should be used since it comes 'for free' with HTTP. XML can be transmitted just as well over an SSL connection as over a plain HTTP connection, albeit somewhat more slowly. The first SOAP (Simple Object Access Protocol) draft (1999) avoided the authentication question, deferring it to later drafts, and suggested the use of SSL. However, although SSL handles authentication, it does not address digital signatures. The W3C (World-Wide Web Consortium) then became involved, setting up the XML Signature Working Group to produce the XML Digital Signature Specification (XML-DSIG). XML-DSIG is an important standard because it supports the digital signing of *any digital content*, not just XML. Thus the debate has come full circle; the signature is now once again part of the XML document, except that now the signature format is a common standard that can be archived and interpreted by any piece of standards-compliant software.

XML Signature

The XML Signature standard describes a set of XML elements and attributes that are used to store information about the hashing and encryption algorithms used to generate a digital signature, as well as, of course, the signature itself. In addition, the public key that is used to verify the signature can be incorporated within the <Signature> block, or alternatively the address of the public key directory that includes the public key can be included.

The discussion relating to the design of the XML Signature standard threw up a number of interesting questions. Some of these touch on philosophical issues, and get to the core of the concepts behind structured data and its representation on-screen. The XML Signature standard mandates that only what is 'seen' should be signed. The word 'seen' is in inverted commas because the user may perceive the information in another media rather than the visual media, for example through sound. It is important to secure the actual data that was presented to the person. This means that if XML is being rendered on-screen using a style-sheet, then the visual representation of the data must be signed, since this is what the user actually sees. It has been suggested that the components used to render the XML should also be signed – the XML Signature specification says that the data must be signed along with 'whatever filters, style sheets, client profile or other information that affects its presentation'. These items may include the browser itself, even video drivers or font packs, or ultimately the operating system itself. The important point is that the user's decision to sign is based on the visual representation of the XML data, not the underlying XML itself.

The Identrus PKI group – a consortium of banks that issue digital certificates signed by the Identrus root certificate authority (CA) – requires that users be presented with a bitmapped image of the document that is to be signed. This bitmap is not useful for subsequent data processing but instead serves as a record of what the user saw. The signing software must ensure that the document that the user views is not being obscured by another application in the foreground. Identrus makes use of an XML format called CSC (Certificate Status Check) in order to authenticate users.

Another interesting aspect of XML Signature is that the document itself must be protected so that no changes happen to it in transit that could invalidate the signature. To understand why this is important, it's necessary to understand

what a *hash* is. A hash is a value produced by a one-way mathematical function run on a piece of data. If someone else runs the same hash function onto the data, they obtain the same hash value. This is how signatures work – this hash value is encrypted with the private key of the signer, and then anyone with access to their public key can decrypt the original hash, compute a new hash based on the data they have received, and make sure that the two hash values are the same. XML presents a number of problems for hashing, however. An XML document may contain some white space between tags, for example, and this white space may be lost when a DOM or SAX processes the XML. Similarly, the order in which tags or attributes occur in an XML document may be changed when it's loaded into a DOM or SAX processor. The problem with this scenario is that when the application computes a hash of the document, the white space or the tag-order having changed, then the hash will not match the original hash and so the signature will not compute. In addition, certain differences between file formats on different operating systems can cause XML documents to subtly change as they are sent between disparate machines. These issues are to be solved by *XML Canonicalisation*. XML Canonicalisation defines a standard way to normalise XML information between operating systems. So-called *canonical XML* is intended to be platform-neutral.

An example of an XML Signature is shown opposite in Figure 1. The SignatureMethod tag tells us that a combination of RSA (for public key encryption) and SHA-1 (for hashing) was used to create this signature. The X.509 certificate that is used to verify the signature is included with the signature itself. This signature is appended to the document which it signs.

PKI – binding a key to a person

The XML Signature standard specifies XML digital signature processing rules and syntax that prove that a document was signed using a certain private key, and then a Public Key Infrastructure (PKI) binds that key to a user's identity. Note that there are two clauses in the previous sentence. Digital signature algorithms provide the mathematical proof of a transaction. However, unless the private key is linked to a person or organisation, that proof is just a mathematical proof. PKI is used to link the transaction to the person, making use of publicly available directories to store the public keys that are used to check the digital signatures and referencing a security policy document to

```

<?xml version="1.0" ?>
<Signature Id="Vordel"
  xmlns="http://www.w3.org/2000/CR-xml-c14n-
  20001026">
<SignedInfo>
<CanonicalizationMethod Algorithm="None" />
<SignatureMethod
  Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
  sha1"/>
<Reference
  URI="http://www.w3.org/TR/2000/CR-xml-c14n-
  20001026">
<Transforms>
<Transform
  Algorithm="http://www.w3.org/2000/CR-xml-c14n-
  20001026" />
</Transforms>
<DigestMethod
  Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
<DigestValue>qyd5dHCHsQ1GXw0j6hk6PZtF8vE=</Digest
  Value>
</Reference>
</SignedInfo>
<SignatureValue>
NrZOJ7rEyIPmLs/CoK2gQJ32EWwkTnAkhuzUMrjs/+WwJ
dJ+3XoP
</SignatureValue>
<KeyInfo>
<X509Data><X509IssuerSerial><X509IssuerName>
c=IE, o=Vordel, ou=TS, cn=VordelCA,
mail=info@vordel.com
</X509IssuerName>
<X509SerialNumber>970241782</X509SerialNumber>
</X509IssuerSerial>
<X509SubjectName>
c=IE, o=Vordel, ou=Dev, cn=Mark,
email=mark@vordel.com,
telephoneNumber=3531215333
</X509SubjectName>
<X509Certificate>
-----BEGIN CERTIFICATE-----
MIICTCCAcGgAwIBAgIEOdS29jANBgYkCgYEAz2emzUvz
nx9/j
eFNc5NUImceS9x9QSP63cxkwlGAQYS3OkOFShmeF6xvt
8ra2Ui
wS0xO1FYXQu7mRIAKQe9zhQaIP63NlsqfuRjLNkRFkHstf
ZtTIE
SzAe5LosLGVgeU8ocT+8f6zu3LkcgqfWJhxq79YScl9OixBY
D6jA
IIDC4IHgEOyDCLhKajGZ2eAnepx4Mk+fSPmGvN7uDuUlk
/OujQ
OwlOG4qYzrd4d4Vax/QV6GXn2UpT894h0giEBxZczY4xIk
CsdXI
GF+PGIfcq1WdPYMG+Nvz661QMrTxGYiG8Aaws2R8+29
mw6jkY
TzcpItNw95FQoM1MpeMfUZcm/Ja7Fon2Qfp9oeGTINE+
Q==mk
l213blkrh[-qwDFSgfEWtet=2tewgfsm,qWuGrt
-----END CERTIFICATE-----
</X509Certificate>
</X509Data>
</KeyInfo>
<Object>
<SignatureProperties>
<SignatureProperty Id="TimeStamp" Target="#Vordel">
<timestamp><date>2001Apr02</date>
<time>10:08:04</time></timestamp>
</SignatureProperty>
</SignatureProperties>
<DirectoryIPAddress />
</Object>
</Signature>

```

Figure 1: XML Signature Example

enforce identity checks on applicants for digital certificates. Implementing a PKI can be a notoriously difficult and expensive undertaking, so many organisations rely on global PKI services such as Verisign or PricewaterhouseCoopers' beTRUSTed.

As we have seen, PKI brings a lot of value to XML. But, conversely, the world of PKI is beginning to become XML-enabled, with the arrival on the scene of XKMS. XKMS is proposed by Verisign, Microsoft, and webMethods, and has been submitted as a W3C note. It comprises two parts – the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS). X-KISS allows a client application to delegate part or all of the tasks required to process an XML Signature to a Trust Service. This is useful for developers who do not want to implement the signature checking themselves, or who want to delegate this functionality to an Application Service Provider that may be optimised for signature checking (eg through hardware acceleration). X-KRSS is an XML-based replacement for existing PKI file formats that are used when a user applies for a digital certificate. XML brings the same advantages to PKI as it brings to other industries – open standards, platform independence, and human-readability. XKMS looks likely to take off, not least because Microsoft is bundling it into its .NET initiative.

Web Services – component-based computing takes to the Web

The long-standing drive towards component-based computing in IT architectures is now moving to the web. Components that are physically located on different computers can run together as one solution, using technologies such as SOAP (for enveloping XML on the wire), UDDI (for publishing information about available services), and DSML (for accessing directories over the web), over frameworks such as .NET, Jini, or E-Speak. A simple example of a web service is a stock quotation object that can be instantiated over the Internet by an application that requires such a tax-calculation feature. By tying together web services, 'business webs' – dynamic collections of businesses – can be spawned on a massive scale. An example of a business web is a retail store that uses UDDI to publish its on-line catalogue, then the catalogue can call a company's shopping cart and a third company's credit card transaction service. Development tools such as Microsoft's Visual Studio.NET and Bowstreet's jUDDI allow

developers to link together web services to create business webs, often without any need for programming.

SOAP is firmly established as the enveloping protocol of choice for web services. Until recently, SOAP did not address the requirement for security. But in January 2001, Microsoft and IBM proposed in a W3C note the integration of XML Signatures into the SOAP 1.1 Envelope via a new <SOAP-SEC:Signature> header entry. The various web services frameworks – .Net, Jini, and E-Speak – will most likely use XML Signature enabled SOAP messages. E-Speak is something of a special case because it was the first fully operational web services design – initially announced by Hewlett Packard back in 1999 – and has recently been updated to comply with the SOAP specification. Certificate-based security is included in E-Speak in the form of fine-grained, rule-based security that uses attribute certificates. It remains to be seen if SOAP-level security will supplant this.

The advent of web services opens up some important questions for security. If it is so easy to string web services together to create a business web, then what is to stop a hacker from exploiting this? What is needed is a way of certifying web services – otherwise a web agent that searches for services has no way of knowing what services to trust. Centralised, trusted, UDDI directories are one way of answering this security question. However, it remains to be seen how well this option will scale. The other option would be to use a certification system similar to Microsoft's AuthentiCode – where the onus is on the vendor to register and sign their service. This has the advantage of retaining the peer-to-peer nature of the Internet, but still depends on the existence of a service to check credentials. As we have seen, XKMS fits the bill as a protocol to deliver this.

And what about firewalls?

One very special reason why XML-specific security is important is that web services typically use the web ports, thereby bypassing firewall restrictions. An example of this trend is SOAP, which earlier in its lineage used to travel over port 135 (the RPC Endpoint Mapper port), a port that is typically blocked by firewalls for security reasons. Now SOAP uses the web ports and so avoids firewalls. Other examples are the

new XML interface on Microsoft SQL Server 2000, or the XSQL feature that allows Oracle 8i to conveniently read in a stream of XML. For an IT Manager it is an appealing prospect to Internet-enable an application by opening it over web ports via an XML interface. Quite often the fact that an application is blocked by a firewall appears to users as if the application 'just plain doesn't work'. Users typically do not understand that a protocol is being blocked by the firewall *for a reason*. This problem held up the spread of CORBA, even resulting in some CORBA vendors resorting to writing their own firewalls. PKI rollouts, too, have been affected by this problem which results in essential LDAP directory lookups (which use port 389) being blocked – hence the need for DSML (Directory Services Markup Language, pronounced 'dismal') to provide an XML-based directory lookup over the web ports.

The XML-based Internet does away with the possibility of denying network traffic based on specifying TCP/IP port numbers. Next-generation firewalls must be capable of dipping into XML streams travelling over web ports to check their payloads, much like today's email virus checkers dip into email data streams on mail servers. In the case of XML signatures this authentication can be done locally or by sending the signature block to an XKMS Trust Service. However if the XML stream is encrypted then a traditional firewall is of limited use, because it simply cannot read the data. SOAP partially gets around this problem by allowing the SOAPAction method name in the HTTP header to travel in the clear so that a firewall can route the document. But this has the disadvantage of giving away information about which web service is being accessed.

The SOAP specification includes the SOAP-specific M-POST command which enables SOAP-compliant programs to add header information to the HTTP protocol to allow fine-grained, rule-based filtering and handling of SOAP messages by firewalls and proxy servers. Of course, this relies on proxies and firewalls being configured to recognise M-POST.

S2ML and AuthXML – two become one?

In November 2000 two separate initiatives were announced to develop an XML standard for transporting security information between

If it is so easy to string web services together to create a business web, then what is to stop a hacker from exploiting this?

on-line commerce systems. The two initiatives are S2ML (Security Services Markup Language), led by Netegrity, and AuthXML, led by Securant Technologies. The goal of both initiatives is to implement Single Sign-On, one of the holy grails of computing, between on-line trading environments. This service is needed because on-line commerce typically involves more than one web site or web service, and these may need to share information about a user. S2ML or AuthXML would facilitate partners and affiliates to link their exchanges together to share 'entitlement' information, for example credit limits and 'gold card' type profiles. Also, both protocols would eliminate the need for users to repeatedly enter registration information onto multiple websites. Participants in S2ML include webMethods, Sun, VeriSign, and Jamcracker. In addition, the ebXML working group has endorsed S2ML. Participants in AuthXML include Check Point, Novell, and Valicert. Some vendors, like some Florida voters, signed up to support both competing initiatives.

In view of the fact that S2ML and AuthXML address the same requirements but are not interoperable, OASIS (part of the Open Group) set up a Technical Committee for XML-Based Security Services to merge the two initiatives into a single standard. It was felt that a single standard would be a more favourable outcome for the industry than two competing initiatives. After all, by definition a 'standard' should be something that everyone uses. The OASIS

initiative to merge AuthXML and S2ML is still at the early stages, having started in December 2000, but is gathering momentum. The initiative has been christened SAML – Security Assertions Markup Language.

How this all fits together

The initiatives described in this article fit into various parts of the following four layers (see Figure 2 below). It is expected that XML Signature will be incorporated into many of the B2B integration frameworks, via the proposed SOAP XML-DSIG header extensions.

The next few months should be interesting for both the XML and security worlds, because they are coming together in interesting ways. XKMS is bringing the XML message of common standards to the digital security industry, notorious for its fragmented standards. Similarly, initiatives such as XML Signature and OASIS SAML are bringing the vital level of trust to business-to-business trading on the Internet. These events should lay the secure foundations for the much-anticipated growth of business webs.

About the author

As Chief Technical Officer at Vordel, Mark oversees the development of Vordel's technical strategy and product development in the areas of XML and public key cryptography.

He can be contacted via e-mail at mark.oneill@vordel.com

	Secure transports: SSL and/or HTTPS	Messaging protocol: S/MIME and JMS	Enveloping Formats: SOAP, etc.	B2B Application Protocol: ebXML, BizTalk, etc.
XML Signature (W3C & IETF)	Independent of transport protocol	Independent of messaging protocol	Proposal for inclusion in SOAP header	BizTalk Framework 2.0 includes XML Signature support
XKMS (submitted to W3C)	Independent of transport protocol	Independent of messaging protocol	Uses SOAP for enveloping	Proposed support in .NET
SAML (OASIS)	Independent of transport protocol	Independent of messaging protocol	Bound to SOAP as XML Protocol (XP) is not available yet	N/A

Figure 2 – XML Security standards vs. B2B network layers

Smartcards – Consumer Non-Repudiation

by Simon Ward, beTRUSTed

In Dr Kim Wagner's article about smartcards (CCE Journal, October 2001) the security concerns related to smartcards and the interface device are raised. This article discusses these issues in more depth and describes a possible consumer solution. (NOTE: Dr Wagner briefly mentioned phones and PDAs in his article.)

Problem

As explained in the previous article, smartcards provide a very secure way of storing and protecting the private keys critical to PKI. However, security issues still arise due to the way these keys are accessed. One common scenario is to have a smartcard reader plugged into a PC and then use software on that PC to sign documents and transactions. For example, a user types their e-mail, chooses to digitally sign it and then presses send. At this point the software prompts the user to insert the smartcard and enter the PIN. A hash of the message is sent to the smartcard and the digital signature is returned. The software then formats the data according to S/MIME and sends it to the recipient. This normally works fine, and it gives the recipient some assurance of whom the message came from and that the message has not been altered or corrupted in transit. However, there are some potential issues with this scenario.

The user sees the message on the screen (because they have just typed it), but how can they be sure that the message which is hashed and sent to the smartcard is the same message that they saw on the screen? Usually the PIN is entered via the PC keyboard: how does the user know that rogue software on the PC is not capturing this PIN?

If rogue software captures the smartcard PIN then it could conceivably unlock the smartcard without the user being aware if the smartcard is left in the reader. The rogue software could then generate whatever signatures it likes or pass the PIN to an attacker that would attempt to steal the smartcard. However, the most serious issue is that of the data being signed. If a user can argue that they did not know that they were signing or did not see the data being signed then they can argue that they did not consent to the content of the message and therefore the 'digital signature' is not a signature in a legal sense. For example, it could be argued that the data being signed was the data sent down the cable to the smartcard reader, which the user cannot of course see. Rogue software could send different data to the smartcard from the data displayed on the screen. In this way, the

software could display 'Pay \$10' to the user, but then send the equivalent of 'Pay \$1000' to the smartcard to be signed. This vulnerability has been documented several times, eg Bruce Schneier, Cryptogram November 2000 (<http://www.counterpane.com/crypto-gram-0011.html#1>).

These issues exist whenever digital signatures are used, but it does not mean that smartcards are worthless because the risks can be mitigated. For example, consider a consumer using a crypto-smartcard to digitally sign a credit card transaction at a point of sale terminal. The terminal would display the transaction information to the user prior to signing. The terminal would be from an approved supplier; have a secure design and be tamper evident in case the merchant attempted to modify it in some way. (Some countries already have the requirement for a secure PIN pad on point of sale terminals.) The customer would then be relatively confident that the terminal did not contain any rogue software, would not capture the PIN and the data displayed was in fact the data that is signed.

The case of a dedicated point of sale terminal is easier because the terminal would only ever be designed to handle card sale transactions. Compare this to a general-purpose computer workstation that would be designed to run a wide range of different programs and support a variety of connections. The proliferation of viruses, worms, Trojans, etc on modern PCs suggests that it is not inconceivable for rogue software to run covertly on a user's PC and exploit the vulnerabilities described above. This risk can be mitigated in a controlled environment. For example, in a bank where the workstation has fund transfer software, the workstation would be protected from rogue software by physical security, network security, anti-virus software, strict configuration management, etc. The users would also be trained so that they knew the significance of the security and were aware that they were committing to the transactions when they inserted their smartcard and entered their PIN to sign.

However, consider a consumer shopping on their home PC – this is probably the worst case for smartcards. Most home users do not have anywhere near the technical skill and understanding to protect their home PCs from rogue software. Therefore they might end up with fraudulent digitally signed credit card transactions and genuinely have no idea how the transactions were originated.

Solutions

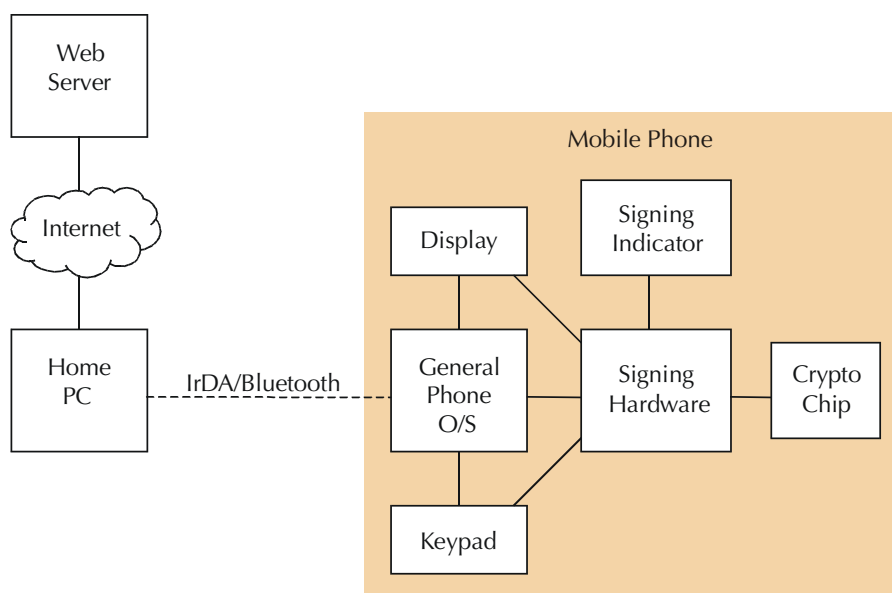
The ideal solution to the home consumer problem would be a dedicated piece of hardware that has a built in smartcard reader, display and keypad. This device would be designed to be secure in a similar way to the point-of-sale terminal. The device would always display the exact message/transaction being signed and resist attempts to collect the PINs. Such a device may sound expensive and difficult to roll out to consumer, but most people already have a very similar device – their mobile phone.

Mobile phones have displays, key pads and the ability to read smartcards (a mobile phone SIM is basically the chip part of a smartcard). Of course, a mobile phone is not a dedicated smartcard terminal, but they are more secure devices than PCs because they do not support the wide range of macros, scripts and executables that PCs do. Additional functionality would have to be added to the mobile phone to support digital signing but it would not require significant extra hardware because the existing display, keypad and IrDA could be used.

How would this work for the user? They could shop on-line using their PCs as normal. When the user is ready to pay or (commit to the purchase if on account) the PC will send the transaction to be signed to the mobile phone, eg by IrDA or Bluetooth. The phone displays the transaction to the user and prompts for confirmation. The user then enters their PIN via the phone keypad. The smartcard is unlocked and signs the transaction. The digital signature is then send back to the PC and processes according to the transactions scheme.

The diagram overleaf shows the main logical component to achieve this.

The phone has a general operating system that handles all the other things the phone does, which are not relevant here. This General Operating system has access to the display and keypad and will probably allow some form of custom programs to run to increase the phones functionality. The signing hardware (or firmware) is tightly controlled such that it only supports the API for signing. (If firmware is used then it must be securely installed and upgraded.) The signing hardware is the part of the phone that can access the crypto chip of the smartcard and can send it data to be signed. The signing hardware will be passed the transaction to be signed in human readable form. This text will then be displayed verbatim on phone display. The user then



confirms/accepts the transaction and enters the PIN so that the crypto chip can be unlocked to sign the message.

The signing hardware will need to take control of the display and keyboard during this operation so that false messages cannot be displayed and the PIN cannot be intercepted. The signing hardware will also need a 'signing indicator', such as an LED on the phone, to show the user that they are committing to a transaction. This is because software on the general phone O/S could display spoof messages that tricked the user into entering their PIN. Only the secure signing hardware can control the signing indicator, so the user would know if the message was a spoof.

Note that such a system could also be built into PDAs in a similar way. Although PDAs are rapidly going the way of PCs – many already run Windows CE.

This solution would not just be limited to home Internet purchases. The same scheme would work at the point of sale in shops or with WAP sites.

Issues

There are three main ways in which the smartcard could be used with the phone. The phone could accept a smartcard in an external slot; a second internal SIM-like slot could be used, or the digital signing keys could be incorporated on to the same GSM SIM chip (called a SWIM).

The external slot would allow users to have many different smartcards, which would be kept in their wallet like normal credit cards.

However, this solution would make the phone bigger when manufacturers are trying to make the phones as small as possible.

The second internal slot (a dual chip phone) would allow the signing chip to be totally independent of the GSM SIM and only make the phone marginally bigger. However, it would not be as convenient for the user to change signing chips, say if they came from different banks. Customers may also need a conventional chip card to use with merchant terminals; thus this solution may be more expensive.

The SWIM would not alter the phone layout or size significantly. However, the SWIM chips would have to be issued and controlled by the network operator. Therefore the network operator would have to issue SWIM chips as routine. Anyone wanting to use the signing features of the SWIM would then have to have some kind of agreement with the network operator, but this could give the network operators additional revenue. However, there would not be the need to pay for and distribute any additional crypto chips, but the SWIM would be slightly more expensive than a normal GSM SIM.

At this point the issues become political. There are several main interest groups examining this type of technology. The WAP Forum (www.wapforum.org) is the largest group and focuses on developing the Wireless Application Protocol (WAP) to provide Internet services on mobile and PDA devices. The WAP Forum specified WML Script which includes a WML Crypto library with a `WML Crypto.signText()` function. This function is designed to provide an API to signing hardware (as described above) to sign text on phones or PDA browsing WAP sites. It

should be noted that the principles described in this document are not intrinsically tied to WAP. The process described above could be done totally without WAP. However, the majority of work done towards using mobile phones for signing has used WAP as the application environment.

The MeT initiative (www.mobiletransaction.org) was started by the major mobile phone manufacturers to establish a framework for using mobile devices to secure transactions – in essence making mobile devices a Personal Trusted Device (PTD). One aim of the MeT initiative is to ensure that this is done in a standard way across devices. This is so system developers can support a wide range of devices. It is also important that the user experience be consistent across devices. Otherwise users could learn to sign transactions with one phone and then mistakenly commit to a transaction on a new phone, if it worked differently.

The Mobey Forum (www.mobey.org) was founded by a group of banks to promote the use of mobile devices for financial transactions. The Mobey Forum has published a Preferred Payment Architecture document. This strongly favours dual chip phones. The reasons cited for this are VISA rules, which do not allow a VISA product to be distributed by a non-bank organisation, and service independence from network operator, which avoids ‘service confusion’ on the part of the consumer. The Mobey Forum have proposed ways in which SET and EMV payment protocols could be used from mobile devices.

The Global Mobile Commerce Forum (www.gmcforum.com) is a wider forum that aims to bring together all the relevant parties to debate issues and promote m-commerce.

In addition to technical and liability issues, branding is also important. Each party involved will want to promote their brand. If the chip is inside the phone then the branding will have to be done via the phone display. Solutions that allow an acceptable level of branding are likely to be preferred by most businesses. Although it is worth noting that in a survey done by the Mobey Forum, banks rate branding as 20th in the list of requirements.

Progress

SIM Toolkit applications have been around for sometime and are capable of digitally signing. However, SIM Applications Toolkit (SATs) are highly proprietary and do not offer an open solution for security. The use of SIM

Applications Toolkit has been very limited. Also, the architecture for signing with a SAT is not necessarily that of a PTD described above.

The initiatives behind the current technology are largely driven through the much-maligned WAP. WAP has not taken off as expected a year ago – mainly because it was slow, expensive and very limited graphically. WAP may improve with packet-switched technology such as GPRS and UTMS and newer, more advanced mobile devices. Many mobile network providers are planning on providing data services as a means to expand their market – this should lead to improvements in WAP.

Phones supporting the WTLS Crypto functions are available now. The Nokia 6310/6510 and the Ericsson T68 support this functionality via WAP using a SWIM chip. However, there are no known network operators currently issuing SWIM chips except in small pilots.

Conclusion

There is an overwhelming incentive to provide good transaction security – customers, merchants, banks, etc will all benefit from this. Using mobile phones as a Personal Trusted Device is one feasible solution to this. However, the standards are still at a very early stage and there are differing opinions on how the details of the security should be implemented.

In the current economic climate, the phone manufacturers and the network providers are all focusing on short-term profit rather than longer-term and higher risk ventures. Therefore the implementation of secure transactions with mobile devices is less of a priority so progress may be slow. Hopefully, some of the pilots will show how mobile phones can be successful in securing transactions, and this will motivate wider use.

About the author

Simon Ward can be contacted via e-mail at sward@betrusted.com

Building Your Appropriate Certificate-based Trust Mechanism For Secure Communications

*by Kaijun Tan, PhD, Scientist
Rainbow Technologies, Inc*

The central issue facing the Internet today can be summarized in one word: trust. A number of companies endeavor to provide services to answer the question of trust – most commonly in the form of digital certificates – which are issued to both individuals and companies in various degrees of security. Certificates represent the concept of a ‘trusted third party’ that is partly a software company, partly notary public and partly a local records office.

Digital certificates can be set up in such a way as to let you easily know whether a public key truly belongs to the purported owner. It consists of three things: a public key; certificate information which is about the user (such as name, user identity and other pertinent identification data); and a digital signature. The purpose of the digital signature on a certificate is to state that the certificate information has been attested to by a trusted third party. Digital certificates effectively thwart attempts to substitute one person’s key for another. When it’s necessary to exchange public keys with someone else, the certificates provide each party with the confidence that the public keys are authentic.

In a small group, people find it easy to manually exchange diskettes or e-mails containing each owner’s public key. But this manual distribution can only be practical when used in the scope of a small business or a compact team of people. Beyond that scope, a more systematic and comprehensive way is required to provide the security, storage and exchange mechanisms necessary so that co-workers, business partners or customers can communicate with a user over the Internet. That is why the Public Key Infrastructure (PKI) was proposed. PKI stores and manages certificates securely. It not only issues certificates and maintains the status for each certificate, but also provides a way to revoke issued certificates. The main feature or the central component of a PKI is the Certificate Authority (CA), which is a human entity – a person, group, department, company or other association. The role of the CA is analogous to a state’s department of motor vehicles in issuing drivers licenses or a department issuing passports.

Despite the de facto standard status of PKI, however, people still hesitate to use it due to complexity and cost. But there is a simple, cost-effective solution to this problem in a big percentage of cases. To understand it, however, this paper will first examine the applications of PKI.

Current PKI Survey

When talking about PKI, we generally refer to the X.509 PKI standard. Actually, there are other PKI standards. There have been some surveys of current PKIs^{1,2}.

X.509/PKIX

This is a hierarchically structured PKI, and is spanned by a tree with a Root Certificate Authority (RCA). So in this structure, the trust is centered at the root, and then transferred hierarchically to all the users in the network via Certificate Authorities (CAs). Specifically, the public key of the RCA is known to all the users in the tree, and this knowledge is used to induce confidence in the public keys of other entities via trust-paths in the tree. There is no need for certifying the public key of the RCA, since it is assumed that this key is known to all entities.

PGP³

PGP certificates are issued by individuals, not like X.509/PKIX certificates which come from a professional CA. Anyone can decide who they trust. To compensate for the fact that the issuers are not specifically protected or professional, PGP implements a security fault-tolerance mechanism called Web of Trust. Under the Web of Trust, multiple keyholders sign each certificate (binding the userID to the public key), attesting to the validity of the binding. The assumption is that these different keyholders are independent, so that even if one of them makes a bad judgment, they won't all do so. The certificate verifier sets the level of trust in the binding after demanding some number of independent signatures on a PGP certificate.

SDSI⁴/SPKI⁵

Like PGP, SDSI/SPKI advocates widely distributed issuance of certificates, rather than having them all come from a central CA hierarchy. But it has deterministic certificate chains just like those of X.509, unlike the PGP Web of Trust. Because SDSI/SPKI defines a k -of- n subject which is a list of n subjects (keys, names) together with the values k and n , such that the verifier needs k complete paths between this certificate and some eventual subject, it is the certificate issuer who decides what level of fault tolerance is required (by selecting k and n).

No PKI⁶

Carl Ellison and Bruce Schneier argued that PKI has too many risks and so it may be better not to have any PKI at all. For more details see [6].

To better evaluate PKIs, let's review their features.

X.509/PKIX is almost the emblem of PKI. Public discussions and articles are always referring to it. There are multiple vendors for this kind of PKI, including Verisign, Entrust, Baltimore and others. Users can buy certificates from them or purchase private CA software from them, but must pay a fee for each certificate issued. This seems the de facto standard for PKI applications. However, the incident in which Verisign wrongly issued a Microsoft certificate⁷ clearly demonstrated that there are problems with the authentication process used by Verisign in verifying the information submitted for digital certificates. The erroneously issued certificate will bring big losses not only for potential Microsoft customers, but for Microsoft itself. This accident raised some doubts about PKI in the public consciousness: Can we really trust a PKI vendor to issue certificates? What is their rule to authenticate the customer before issuing a certificate? There are too many uncertainty factors or risks^{6,7,8} that challenge this hierarchically structured PKI. PGP or SDSI/SPKI, in contrast to PKI's hierarchical structure, are unstructured frameworks. This lax feature makes them more intuitive and appropriate for an individual environment.

Carl Ellison and Bruce Schneier agree that the many risks in current X.509/PKIX mean that, perhaps, PKI isn't needed at all. This is unrealistic as it is like suggesting you shouldn't use the Internet since there are too many security threats. Without PKI, users won't have confidence in the authenticity of a public key. At this point, PKI has its positive benefits. These days, people are realizing that password-based authentication is not secure, and certificate-based authentication is becoming the de facto industry standard. Additionally, the popular SSL protocol in eCommerce through the Internet requires certificate-based authentication and session key exchange. Thus no PKI at all is not an option.

Intended Applications for PKI

It is important to consider the intended application or use of the PKI from the application's perspective. For ease of analysis, it is helpful to separate the possible fields of use into different business models that are commonly discussed today¹: the business-to-business model (B2B); the single business or enterprise model (B); the business-to-customer model (B2C); and the individual model (I). We will analyze whether each category has its own unique PKI, or whether a single PKI can

be the right choice for multiple categories simultaneously.

It is very natural to take X.509/PKIX into B2B, B and B2C since they are all in business or commerce environments. As for the individual model, hierarchical PKI is overly complex-PGP or SPKI or simpler solutions involving public key technology might be more appropriate¹.

Is that really the case for X.509/PKIX? Can there be simpler solutions for these business models? On the following pages we would like to do a comprehensive analysis.

Analysis for Hierarchically Structured PKI in Business Models

In a B2B model, the certificates are used between two different businesses and they are in an equal position. Both of them care about the credibility of each other's certificate, so the best way is to have a higher-level CA which can be trusted by both of them to issue certificates for them.

For a B model, in a single enterprise or business, the certificates are used right inside the enterprise. The enterprise must trust these certificates, which are held by its employees. So the two entities in the trust relationship are not in an equal position: employees belong to the enterprise and it is the enterprise that needs to trust the employees.

B2C is still the predominant model in eCommerce. People are all familiar with this scenario: go to a Web site, type in your credit card number and buy something. Clients need no certificate at all. It seems to work well in most cases. However, if you are a frequent visitor to some Web site's service, you won't like typing in a credit card number every time. Or perhaps the business wants to manage its clients in an effective way and would like to authenticate them when they ask for services. Thus the client certificates in this model are not unnecessary. It is easy for the clients to recognize the business's certificate only if it comes from some well-known PKI vendor. This is popular for today's B2C, but it is difficult to attain the required client certificates. It's impractical to require users to buy a certificate from a PKI vendor before they can access services, or require the business to buy certificates for each client before allowing them to access the services. So, while there is client authentication in the SSL standard, there is seldom client authentication in real SSL applications.

Argument for B and B2C

Based on the former analysis, we would argue: do we really need X.509 PKI/CA in the B and B2C model?

If we suppose X.509/PKIX is to be adopted in these two models, there are two methods for deployment:

- The enterprise or business buys private CA software from the PKI vendors and issues its own certificates. Then for each certificate it issues, a fee will be charged from this enterprise or business by the vendor. If the number of employees in this company or the clients asking for services is huge, this is an expensive solution.
- The enterprise buys certificates from a vendor CA for the employees, which still has a cost problem similar to the former case. Or the business service counts on the clients' own certificates, which involves the uncertainty problem such as the authenticity of these certificates and the trust involved. Additionally, it is not practical to require clients to get the certificate themselves before they can access the service.

Actually, since the certificates of employees or clients are used only in the enterprise or business service in a small scope, the directional trust is not equal. It is easy for the employees to trust their own enterprise, or the clients to trust the business, however, the reverse direction is different. The enterprise or business needs a convenient and practical way to trust its own employees or clients.

Current X.509 PKI vendors try to provide certificate services for their customers, with the idea that these certificates issued by them can be trusted in the whole Internet or eCommerce world. What they are actually charging for is their services as a kind of arbitrator. But for the B model or B2C model, does the enterprise or business really need this kind of arbitrator service for the employees or clients? The answer is no. What they really care about are:

- Certificate-based security services, such as signing, authentication and key exchange.
- That the certificate can be trusted in its own scope, not necessarily in a larger one.

With this analysis, we finally reach our point: for the B model or B2C model, what is really needed is not a CA, but a CI (Certificate Issuer).

Certificate Issuer and Certificate Authority

Certificate Issuer (CI) is a service which provides the basic certificate-related functions, such as issuing certificates, managing a CRL (Certificate Revocation List) for the revoked certificates and maintaining the status for these certificates. Some might want to call it a 'Minimum CA', which we would argue is not an appropriate name.

No matter whether it is minimum, maximum or middle-sized, a CA is still a CA. CA means 'certificate authority', so it must work as an arbitrator, a trust organization that provides the services that link an identity to the certificate. A common CA has four basic functions/responsibilities:

1. Issue certificates;
2. Maintain status information and issue CRLs;
3. Publish certificates and its current CRLs; and
4. Maintain archives of status information about the expired or revoked certificates that it issued.

This combination process is trusted by others in some direct or indirect way. Since it is an arbitrator, it is important to have a good process to authenticate the potential certificate holder before issuing the certificate, which is also a big risk for the whole CA service. In a general way, a customer who applies for a certificate has to provide enough documentation to convince the CA that he is the one he claims to be. A CA vendor will investigate these documents with a third-party database, even notaries.

The CI differs significantly from a CA in that it only works as a dummy service and provides some certificate-related functions. The trust relationship depends on the security policy in the enterprise or business which sets this CI. A CI differs from a CA in the following ways:

1. Simple certificate issuing. Before issuing certificates, this enterprise or business already has its own way to differentiate its employees or clients. Thus, the CI won't provide a way to authenticate the potential certificate applicants, but uses the existing authentication method in the enterprise or business. In addition, since the certificates issued by the CI will only be used in the small scope of the enterprise or business, there is no need for the publicly known identity name of the certificate holder. It is enough to be unique in this small scope.

2. Since the CRL will only be checked in the small scope of the business or enterprise, there is no need to publish it and the information in the CRL can be as simple as a list of serial numbers.
3. A certificate which holds the public key and its related private key are usually used in three ways: authentication, key exchange and signature verification. In the B2C model, they are generally only used in real-time authentication and key exchange, but not for document or message signing. In this case, the CI has no need to keep the status of the certificates since nobody will check for the validity of a signature that is signed by a private key with an expired certificate.
4. CA is a part of PKI, and PKI is an infrastructure which involves a lot of policies, procedures, deployment and maintaining. It never can become a plug-and-play solution. CI, on the other hand, is only a dummy service which counts on existing policy of enterprise or business, so it is able to be deployed as a plug-and-play solution.

In fact, writing a CI program is very simple since free software already exists, such as OpenSSL and JDK. Additionally, some hardware devices such as the CryptoSwift HSM from Rainbow Technologies (<http://www.rainbow.com>) can assist high-performance CI functionalities. Additionally, HSM protects CI root private key which is used to sign certificates. So why do you need to buy CA software or certificates in these special conditions?

Other uses for CI

We've talked primarily about the CI in the B2C and B models; actually, it can be used in broader areas, such as CableLab's DOCSIS and PacketCable, LMDS/MMDS, etc. The manufacturers of cable modems, MTAs (Multimedia Terminal Adapter) or antennas, can use their own CI to generate certificates for these devices for identification, and will find it unnecessary to buy certificates from a CA.

Summary

Although PKI and CA is a hot topic in the information technology industry, people still hesitate to use it. There are several reasons. First, the concepts of PKI or all the kinds of services included in a PKI are too complicated and not easy to understand. Secondly, buying certificates from PKI vendors is costly. Even if you buy CA software from them, you still have

to pay a fee for each certificate you generate with this CA. In this paper, we analyzed the features of B and B2C models, and found that what they really need is a Certificate Issuer, not a CA. It is easy to write a CI program with the free software available. With this in mind, you really can be freed from the burden of PKI and CA, while taking advantage of certificate-based security techniques. You set your own level of trust needed when generating certificates with your own CI for your employees or clients, and you have no need to count on a third party. Writing a CI program costs almost nothing; the same is true for generating certificates. So using PKI-based security services can be both easy and inexpensive!

References

- ¹ C. Adams, M. Burmester etc. Which PKI (Public Key Infrastructure) is the right one? CCS'00, Athens, Greece.
- ² SPKI/SDSI and the web of trust. <http://world.std.com/~cme/html/web.html>
- ³ P. R. Zimmermann. The official PGP user's guide. MIT Press, Cambridge, Massachusetts, 1995.
- ⁴ R.L.Rivest, B. Lampson. SDSI-a simple distributed security structure. <http://theory.lcs.mit.edu/~cis/sdsi.html>
- ⁵ C. Ellison, B. Frantz, etc. SPKI Certificate Theory. <http://www.isi.edu/in-notes/rfc2693.txt>
- ⁶ C. Ellison, B. Schneier. Ten risks of PKI: what you're not being told about Public Key Infrastructure. Computer Security Journal, V.XVI, N.1, 2000.
- ⁷ R. Forno, W. Feinbloom. PKI: A question of trust and value. Communications of the ACM, Vol.44, No.6, June 2001.
- ⁸ Forno, W. Feinbloom. A matter of trusting trust: why current Public Key Infrastructure are a house of cards. www.infowarrior.org/articles/2001-01.html

About the author

Kaijun Tan is Scientist for Rainbow Technologies, and a post-doctoral researcher at the Computer and Information Science University of Pennsylvania.

She may be contacted via e-mail at kaijunt@saul.cis.upenn.edu, or at her web site, <http://www.cis.upenn.edu/~kaijunt/>

Upcoming Conferences

The following list of conferences has been brought to our attention. We would welcome any additions.

April 14-15, 2002
San Francisco, CA, USA

Workshop on Privacy Enhancing Technologies

<http://www.pet2002.org/>

April 16-18, 2002
San Francisco, CA, USA

CFP '02: 12th Conference on Computers. Freedom & Privacy

<http://campus.acm.org/calendar/confpage.cfm?ConfID=2002-1536>

April 24-25, 2002
Gaithersburg, MD, USA

1st Annual PKI Research Workshop

<http://www.cs.dartmouth.edu/~pki02/>

April 28-May 2, 2002
Amsterdam, The Netherlands

EuroCrypt 2002

<http://www.ec2002.tue.nl/>

May 7-9, 2002
Cairo, Egypt

IFIP/SEC 2002: 17th International Conference on Information Security

<http://www.sec2002.eun.eg/>

May 12-15, 2002
Oakland, CA, USA

2002 IEEE Symposium on Security and Privacy

<http://www.ieee-security.org/TC/SP02/sp02index.html>

May 29-30, 2002
Vienna, VA, USA

eSecurity Conference & Expo

<http://seminars.internet.com/esec/spring02/index.html>

June 3-7, 2002
Porquerolles Island, France

**Yet Another Conference on
 Cryptography (YACC'02)**

<http://www.univ-tln.fr/~grim/YACC/>

June 17-19, 2002
San Francisco, CA, USA

NetSec 2002

<http://www.gocsi.com/netsec/02/>

July 3-5, 2002
Melbourne, Australia

**ACISP 2002 – The 7th
 Australasian Conference on
 Information Security and Privacy**

<http://www.cm.deakin.edu.au/ACISP'02/>

July 11-12, 2002
Kitakyushu, Japan

**STEG'02 – Pacific Rim Workshop
 on Digital Steganography 2002**

<http://www.know.comp.kyutech.ac.jp/STEG02/>

August 13-15, 2002
Redwood City, CA, USA

**Workshop on Cryptographic
 Hardware and Embedded System**

<http://security.ece.orst.edu/ches/>

August 15-16, 2002
St. John's, Newfoundland, Canada

**SAC 2002: Ninth Annual
 Workshop on Selected Areas in
 Cryptography**

[http://www.cs.utah.edu/flux/cipher/cfps/
 cfp-SAC2002.html](http://www.cs.utah.edu/flux/cipher/cfps/cfp-SAC2002.html)

August 18-22, 2002
Santa Barbara, CA, USA

Crypto 2002

<http://www.iacr.org/conferences/crypto2002/>

September 4-5, 2002
Aix en Provence, France

**Workshop on Trust and Privacy
 in Digital Business**

<http://www.wi-inf.uni-essen.de/~dexa02ws/>

September 5-7, 2002
Oviedo, Spain

**VII Spanish Meeting on
 Cryptology and Information
 Security**

<http://enol.etsiig.uniovi.es/viirecsi/viirecsiev.htm>

September 23-25, 2002
Essen, Germany

**ECC 2002 – The 6th Workshop on
 Elliptic Curve Cryptography**

[http://www.cacr.math.uwaterloo.ca/
 conferences/2002/ecc2002/announcement.html](http://www.cacr.math.uwaterloo.ca/conferences/2002/ecc2002/announcement.html)

September 23-26, 2002
Hampton, VA, USA

**New Security Paradigms
 Workshop 2002**

<http://www.nspw.org/current/>

October 1-3, 2002
Bristol, UK

**Infrastructure Security
 Conference 2002**

[http://www.cs.utah.edu/flux/cipher/cfps/
 cfp-InfraSec2002.html](http://www.cs.utah.edu/flux/cipher/cfps/cfp-InfraSec2002.html)

October 7-9, 2002
Noordwijkerhout, The Netherlands
5th International Workshop on
Information Hiding

<http://research.microsoft.com/ih2002/>

November 20-22, 2002
Fifth Smart Card Research and
Advanced Application
Conference (CARDIS'02)

<http://www.usenix.org/events/cardis02/>

December 1-5, 2002
Queenstown, New Zealand
AsiaCrypt 2002

<http://www.commerce.otago.ac.nz/infosci/asiacrypt/>

December 9-12, 2002
Singapore
ICICS 2002 – Fourth
International Conference on
Information and
Communications Security

<http://www.krdl.org.sg/General/conferences/icics/Homepage.htmlx>

Call for Articles

If you are interested in contributing to this publication, we invite you to submit articles containing your thoughts, ideas and concepts.

Contribution guidelines for papers being submitted to the Cryptographic Centre of Excellence Journal are:

- Topic must fall under the umbrella of cryptography, security and/or privacy;
- Articles should not be of a promotional or product marketing nature;
- All submissions will be reviewed for content and may be declined at the discretion of the editor (for example, if the tone and/or content is overtly promotional or product marketing-oriented);
- Maximum article length to be 5,000 words plus tables/graphics;
- Submissions must be original work and, where appropriate, give credit to the original author(s);
- The editor reserves the right to edit the text with the agreement of the author; and
- All submissions must be made in MS Word or .RTF format.

PricewaterhouseCoopers reserves the right to re-format for publication purposes and re-distribute as appropriate.

Authors maintain ownership of all submissions.

Completed submissions or abstracts should be submitted via e-mail to either:

geoffrey.c.grabow@us.pwcglobal.com

john.velissarios@uk.pwcglobal.com

beTRUSTedSM

An e-security business of
PricewaterhouseCoopers

www.beTRUSTed.com