[Home] [Current Edition] [Compendium] [Forum] [Web Archive]
[Email Archive] [Guestbook] [Subscribe] [Advertising Rates]

Hettinga's Best of the Month

From Contributing Editor Bob Hettinga
Email: rah@shipwright.com
URL: The e$ Home Page

"... however it may deserve respect for its usefulness and antiquity, [predicting the end of the world] has not been found agreeable to experience." -- Edward Gibbon, 'Decline and Fall of the Roman Empire'

From: Lynn.Wheeler@firstdata.com
To: e-payments@lists.Commerce.Net
Date: Mon, 5 Jan 1998 18:51:55 -0800
Subject: x9.59 electronic payments and account-authority digital signatures
Sender: e-payments-owner@firstdata.com
Precedence: bulk

X9 is working on x9.59 electronic payments in the x9a10 working group. In support of x9.59, I've been working an account-authority digital signature model .... the attached is marginally technical.

Three digital signature models are described; the original "offline" model and two newer "online" models. It is expected that the two "online" models become the prevailing modes of operation for online financially-related and/or electronic commerce transactions.

Digital Signature Model 1:

The traditional PKI infrastructure talks about issuing certificates that are signed by a certificate authority which attest to the validity of the public key, preferably checking the validity of the private key, possibly some identity information of the entity that the certificate is issued to.

The associated PKI-use model has an entity "digitally signing" a document with their private key and "pushing" the transaction/document, the digital signature and a copy of their digital certificate to another party. The receiving party presumably will validate the authenticity of the digital-signature and the originator's public key via the contents of the associated digital certificate. Originally the contents of the digital certificate was assumed to be sufficient such that digital signature validation could be performed without any additional electronic transmissions. As the methodology matured, it became apparent that more and more complex verification mechanisms were needed, if nothing else various status could have changed between the time that the certificate was originally manufactured and the current moment. Certificate revokation lists (CRLs) were one such development in an attempt to partially address the issue of current real-time certificate status in the offline verification model.

Digital Signature Model 2 (or account-based PKI):

This is a proposed implementation for the X9.59 framework. An account-holder registers their public-key (and verification of their private-key use) with the account authority. In a transaction the account-holder digitally signs a transaction and pushes the transaction, the digital signature, and the account number. Eventually the transaction arrives at the account authority and the digital signature is verified using the public key registered in the account record. The account authority maintains the status of the holder's public key as part of the overall account management process. The transaction therefore requires neither a certificate nor some complex status methodology (like CRLs) since the account authority maintains current validity status as part of account management.

This is effectively the X9.59 check and credit-card models where the receiving entity/business forwards the payment instruction to the account issuing institution. The payer's digital signature is forwarded by the receiving businness to the issuing institution; the issuing institution authenticates the digital signature using the registered public-key in the account record. No signed certificate attesting to the validity of the public key is required since the public-key is on file in the account record.

An account-authority performs the majority of the same functions performed by a certificate authority, but the processing costs are absorbed by the standard business process ... not by charging for the issuing of a certificate. It is possible that an account-authority might also wish to become a certificate authority since it potentially could be undertaken at less then 5% additional business costs.

Digital Signature Model 3 (positive authentication):

This is actually a slight variation on #2, although it bears some superficial resemblance to #1. The initial designs for positive authentication PKI, used the credit-card authorization model to replace CRLs. However they kept the rest of the infrastructure; the originator's certificate was still pushed around with the transaction. The receiver validated the CA's signature on the certificate, then sent off a certificate status request and validated the CA's signature a second time on the status response (and then validated the original digital signature).

Model #3 looks very much like model #2 in that the originator's certificate is not pushed around with the transaction. However, rather than sending the digital signature in the authorization request, just the certificate identifier (account number) is sent to the CA. The CA signs a status response that includes information regarding the real-time validity of the account along with a copy of the account's public key. In effect the real time status response becomes a mini-certificate. The entity that will act on the transaction now only has to verify the CA's signature on the status response (i.e. mini-certificate, it doesn't also have to verify the CA's signature on a certificate manufactured at some point in the past). It then uses the public-key returned in the status response to validate the originator's digital signature.

Superficially this resembles digital certificate model #1 but the actual operation is much more like model #2. Including the account's public-key in the real-time status response creates, in effect, a mini-certificate. It also eliminates a redundant/superfluous validation of the CA's digital signature (on both the manufactured digital certificate and the real-time status responses).

The biggest operational difference between #2 and #3 is that the account authority verifies the originator's digital signature in #2 and in #3 it just returns the value of the account's public key for the requester to validate a digital signature. If the requester can send the document or even the secure hash of the document to the account authority along with a copy of the digital signature, then the account authority can verify the digital signature. If not, the request just identifies the account and the mini-certificate is returned allowing the requester to validate the digital signature.

The positive authentication model presents a number of revenue opportunities for the CA to charge for various levels of detail returned in real-time status responses (and/or approval levels associated with the transaction).

Conclusion

Digital signature model #1 was originally developed to allow "offline" verification of a digital signature. A manufactured certificate was pushed along with the signed document and the digital signature could be verified using just the contents of the certificate that was passed along with the document.

Offline signature verification using a certificate manufactured several months in the past (and by implication relying on status that was several months stale) turned out to be inadequate for various kinds of transactions. This has led to the definition of more complicated processes in the certificate-push model in an attempt to provide more timely status and verification.

There has also been the implicit assumption that only the certificate authority is performing registration services for digital signature processes. As the concept of digital signatures have become more acceptable, it has also becoming apparent that existing business processes (already performing account registration functions) can be simply extended to add public-key registration.

Revisiting the PKI basic architecture, it became apparent that there were several optimizations possible if it was recognized there were significant numbers of online PKI operations (compared to earlier models that started out assuming offline PKI and later tried to graft online features afterwards).

The offline validation and certificate push model is still valid for some types of transactions and shouldn't be precluded. However, real online validation (models #2 and #3) can eliminate some number of superfluous operations.

It should be noted that the "offline" validation is different than the "offline" purchasing referred to in X9.59. X9.59 assumes that the purchuser/payer can be offline and transmits an order and payment instructions via methods like email (not requiring real-time, online interaction with the business). In the validation process, there is an issue whether the business is also offline at the point that it approves the transaction. If the business is offline, then it needs a payer's certificate to validate (and authorize) the payer's transaction. If the business is online, then either model #2 or model #3 is used (and it is not necessary for the consumer to push the certificate with the transaction). Furthermore, in the case of model #2, either the business can perform its own PKI registration function and/or it can rely on a financial account infrastructure to have implemented a PKI registration function.

It is expected that digital signature models #2 and #3 become the prevalent modes of operation for at least financial transactions.

Denial of service attack addenda

There is a hypothetical case (that can be made for certificate pushing in the online world) which is associated with anonymous denial of service attacks. The existing Internet infrastructure provides significant opportunities for electronic terrorists to anonymously (and/or under assumed identity) launch denial of service attacks (flooding a web site with enormous number of bogus requests). These are undertaken with the assumption that it is nearly impossible to trace the source of the attack.

One of the techniques for dealing with denial of service attacks is to recognize and eliminate bogus requests as soon as possible. If a certificate is pushed with a request then some preliminary screening of requests can be performed during initial processing and possibly eliminate some number of bogus transactions.

The downside is that public key operations are extremely expensive; preliminary screening of a request using the certificate (and still doing the online validation later) could be more expensive than allowing bogus transactions through and recognizing them via the standard mechanism.

Most of these are simple band-aid solutions. The real problem is that existing Internet backbone operation makes it simple to impersonate a network address. As a result it is usually very difficult to trace back to the originator of an electronic attack.

From: Lynn.Wheeler@firstdata.com
To: dwightarthur@mindspring.com
cc: apearson@nscc.com, dcsb@ai.mit.edu
Date: Sun, 11 Jan 1998 10:51:22 -0800
Subject: Re: variations on your account-authority model
Sender: bounce-dcsb@ai.mit.edu

Precedence: bulk
Reply-To: Lynn.Wheeler@firstdata.com

There are several differences between a Certification Authority Digital Signature model (CADS) and an Account Authority Digital Signature model (AADS).

Model two or AADS has the account-holder registering a public key with their financial institution almost identical to the way a public key is registered in CADS ... but no certificate is issued; the public key just goes into the database. While the technical processing looks like CADS ... the AADS business process just mimicks signing a signature card when opening an account.

From a financial business standpoint, the AADS is really only an electronic version of a signature. There is no reliance on a certificate by the financial business process ... AADS strictly emulates the existing business process but with a digital signature. There is no worry about any liability associated with somebody attaching meaning to a certificate/credential since there are none.

In a CADS scenerio, with a certificate playing some role in a financial transaction authentication process, the AADS model eliminates extraneous computational intensive cryptographic operations associated with validating the certificate (and/or any certificates in a CA-hierarchy). The X9.59 payment object requires a single digital signature and a single digital signature verification.

Also in a couple meetings with some very smart financial folks this past week, it was pointed out that the lack of certificates and Certification Authority (especially one independent of the financial house) eliminates systemic risks to financial infrastructure associated with various kinds of Certification Authority &/or certification private key failures. Compromise of individual private keys can still occur but there is no systemic compromise of the infrastructure.

NSCC relying on a certificate (and therefor the real-time integrity of the CA's infrastructure) only for registration (and not for any financial transaction approval or processing decisions) could significantly mitigate the systemic risk to the financial infrastructure (except possibly attacks involving re-keying of accounts).

To some extent X9.59 (and my work on AADS) has been:

- KISS
- exactly mimick the existing financial business processes where possible
- eliminate unnecessary computational intensive cryptograpic operations

i.e. looking at how signatures operate in the existing account-based financial business processes and simply translating that into digital signatures resulted in realizing that certificates were superfulous (in this specific instance).

X9A10 is a working group in X9A (some information at http://www.x9.org). At the moment on the web there is only a presentation by chair of X9A10 (Tom Jones) given at the July EPF meeting (check July presentations at http://www.epf.org/).

For financial transactions, my guesstimate has been the trade-off is some where around 5% participation of accounts where independent certification pilot implementations and core-processing integration cross over. AADS-model assumes core-processing integration, so it requires more upfront commitment by a financial institution (but the business costs should scale much better since it is now part of the normal operation).

I can take your CAIP invitation to the appropriate FDC business people.

From: Lynn.Wheeler@firstdata.com
To: dwightarthur@mindspring.com
cc: apearson@nscc.com, dcsb@ai.mit.edu
Date: Sun, 11 Jan 1998 12:09:31 -0800
Subject: Re: variations on your account-authority model (slight addenda)

Sender: bounce-dcsb@ai.mit.edu
Precedence: bulk
Reply-To: Lynn.Wheeler@firstdata.com

Not directly related ... you might be interested in the stuff I do for the IETF (which includes the stuff that goes into section 6.10 of STD1): http://www.garlic.com/~lynn/rfcietf.html

Another project in support of electronic commerce is I've put together both a payment & security glossary and taxonomy at:
http://www.garlic.com/~lynn/payment.html
http://www.garlic.com/~lynn/secure.html

Copies of these files (may sometimes be slightly stale) are also available at: ftp://ftp.netcom.com/pub/ly/lynn

A new group in X9; X9F5 is starting to look at the policy and security issues associated with digital signatures in financial operations.

From: Lynn.Wheeler@firstdata.com
To: dwightarthur@mindspring.com
cc: apearson@nscc.com, dcsb@ai.mit.edu
Date: Sun, 11 Jan 1998 17:58:10 -0800
Subject: Re: variations on your account-authority model (small clarification)
Sender: bounce-dcsb@ai.mit.edu
Precedence: bulk
Reply-To: Lynn.Wheeler@firstdata.com

In AADS, a public key is registered with the account-authority ... effectively the digital equivalent of the signature card; technically it is analogous to the way a public key is registered with a CADS ... but no certificate need be issued ... since it is never necessary to push a certificate along with a payment instruction; the issuing bank is able to validate a signed payment instruction with the public key registered for the account (the purpose of the certificate supposedly is to allow a digital signature to be validated w/o resort to any additional mechanism ... especially in the NPR/no-prior relationship scenerio).

In the financial scenerio ... seperating the authentication of the payment instruction (from the approval/execution) via the certificate (possibly stale &/or dependent on large number of complex network operations) creates (at least) both liability as well as a systemic risk problem ... which are unecessary.

In CADS, a pushed certificate (accompanying a transaction), supposedly relies on the certification authority signing of the certificate (indicating a valid public key for the transaction). The financial authority, executing the transaction, supposedly would use the public key in the certificate (after verifying the certificate digital signature of the certification authority) to validate the signature on the transaction.

CADS is a reasonable model for the offline & NPR operations ... possibly providing an improved sense of security for two otherwise anonymous (to each other) parties (and hopefully risk). A trusted 3rd party certification authority can improve the integrity of an otherwise somewhat random event.

In the financial account transaction operation ... where there is already a significant relationship ... inserting dependencies on a 3rd party into the process, increases the risk (rather than decreasing risk ... in contrast to the NPR scenerio)

The word/lawyer example turns out to be bit of a red herring ... if really comparing apples-to-apples ... it would be not whether or not "legal options" were available for word documents ... but instead the legal option not only had to exist ... but a legal attachment (aka certificate) had to sent with every document (whether it was needed or not).