Computer Networks 67 (2014) 235–251

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Banking on interoperability: Secure, interoperable credential management

Glenn Benson [a], Shiu-Kai Chin [b], Sean Croston [d,1], Karthick Jayaraman [c,*,2], Susan Older [b]

[a] JP Morgan Chase & Co., 270 Park Ave FL 12, New York, NY 10017, USA[3]
[b] Dept. of EECS, Syracuse University, Syracuse, NY 13244, USA
[c] Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, USA
[d] State Street Bank and Trust Company, 1 Lincoln St, Boston, MA 02111, USA

ARTICLE INFO

Article history:
Received 27 March 2012
Received in revised form 16 March 2014
Accepted 25 March 2014
Available online 1 April 2014

Keywords:
Certificate
Authentication
Authorization
Protocols
Trust
Wholesale banking

ABSTRACT

An interoperable credential system allows users to reference a single asymmetric key pair to logon to multiple web sites and digitally sign transactions. Models that govern how keys are created, authorized, validated, and revoked are a crucial part of such a system. These models have security, scalability, and liability implications for businesses, so the requirements vary depending on the parties involved. However, the prevailing the public key infrastructure (PKI) system does not meet these diverse needs. PKI requires a certificate authority (CA) to act as a trusted third party for the parties in a transaction. For example, PKI features a receiver key validation model that requires the receiver of the transaction to communicate with a CA to validate the sender's key used to sign a transaction. These aspects conflict with liability concerns and interoperability goals of businesses doing high-value transactions such as wholesale banking. This paper presents Partner Key Management (PKM) as a mechanism which sufficiently addresses security and liability concerns of businesses performing high-value online transactions, and uses wholesale banking as the motivating example. PKM does not rely on a trusted third party, and features several flexible revocation models to accommodate diverse regulations. PKM is not merely a proposal. Rather, the financial industry has implemented the technology in some of its wholesale banking sites thereby securing millions of dollars of transactions every day. Finally, this paper justifies the security of PKM and its flexible revocation models; and illustrates the justification with proofs through formal logic.

1. Introduction

Imagine a vision for Internet security where users reference a single asymmetric key pair to routinely login to multiple web sites and digitally sign transactions. Imagine if the security technology were strong enough to be permissible by banks, insurance companies, health care, government agencies, and most other business domains. Perhaps, some may argue that PKI technology already realizes this vision today; however, theory differs from practice from the perspective of interoperability.

For example, suppose an insurance company were to issue a certificate to a user, but mistakenly identifies that user incorrectly. Further suppose that the user were a medical doctor authorized to prescribe medication; and the insurance company inadvertently issues the certificate to an adversary who prescribes medication for nefarious

* Corresponding author. Tel.: +1 315 395 9182.
E-mail addresses: glenn.benson@jpmchase.com (G. Benson), skchin@syr.edu (S.-K. Chin), sbcroston@statestreet.com (S. Croston), karjay@microsoft.com (K. Jayaraman), sbolder@syr.edu (S. Older).

[1] The author did this work during the time he worked for JP Morgan Chase & Co.

[2] The author did this work while he was graduate student at Syracuse University.

[3] The authors of this paper are solely responsible for the content thereof. The paper does not reflect the views or the official policy of JPMorgan Chase & Co.

http://dx.doi.org/10.1016/j.comnet.2014.03.024
1389-1286/

purposes resulting in injury or death. Which organization is at fault? The legitimate doctor is a victim as opposed to a perpetrator because the doctor may have been unaware of the insurance company's mistake. Because of this scenario, the insurance company normally chooses to opt-out of interoperability, thereby avoiding the possibility of many potential lawsuits. That is, the insurance company may potentially distribute certificates to its own users, but does not encourage or participate in certificate sharing across web sites not owned by the bank. Unfortunately, by choosing to opt-out, the Internet cannot realize its interoperability vision. The result is the situation in which we find ourselves today where the Internet exhibits insufficient security; and the users complain about a proliferation of passwords that they cannot handle.

Alternatively, one may potentially consider Single Sign-on technology as a solution for interoperability. However, while Single Sign-on effectively provides federated identity, Single Sign-on fails to meet the digital signature requirement. Consider the situation in which a bank executes a payment moving millions of dollars to a beneficiary. Subsequently, the user denies the payment and requests a refund. In order to help adjudicate the dispute, a digital signature's non-repudiation capability may prove beneficial. By analyzing the signature, a judge can determine whether the transaction's signer had possession of the required asymmetric private key; and the judge can identify whether the transaction amount or beneficiary may have been tampered.

By providing key management technology that interoperates between any web site, even those that handle million dollar payments, everyone benefits by amortizing the cost of security over multiple sites. Suppose each user were to obtain a physical security credential that locks an asymmetric private key. While the private key cannot leak off the credential, the credential has the computing capability to perform asymmetric cryptographic operations. In the absence of interoperability, security credentials have limited practicality because a user requires a separate credential for each web site. However, if all of the user's web sites offered interoperable security, then the user would only need a single credential to login and sign transactions everywhere. If a user could use the same credential for many different sites, then the user may be more willing to procure improved security credential hardware. For example, one user may choose to lock his or her key pair in an encrypted file on a smart phone. Another user may lock the key pair on a secured, cryptographically enhanced USB token. A third user may lock the key pair on a cryptographically enhanced token that only unlocks after providing a thumbprint. Credential vendors could continually improve by offering technology at different price points and levels of security. Ultimately, everyone wins: the Internet becomes simpler because each user gets a single credential; the Internet web sites raise their security because digital signature technology becomes commonplace; and credential technology upgrades because users voluntarily choose to upgrade to better technology.

This paper provides a key management solution that realizes the interoperable Internet security vision by directly addressing the liability concern without sacrificing security or interoperability. For clarity, this paper makes three simplifying assumptions. First, the paper bypasses the potential privacy problem by assuming that each user has a single key pair. In practice, a user may potentially wish to create multiple virtual identities each represented by a key pair, but this paper simplifies by assuming only one identity per user. Second, this paper describes digital signatures, but does not detail login events. In practice, a login event is a simple extension of a digital signature that requires a user to sign a random number chosen by the web site. Third, this paper narrows the domain to wholesale banking by describing a technology that allows a user to employ a single key pair to sign transactions at multiple banks. Wholesale banking is a microcosm of the greater Internet security problem because it focuses upon the liability concern. If Bank-A and Bank-B were to each allow a user to authorize a multi-hundred million dollar payment with the same certificate, then one may intuitively extend the security technology beyond banking to other areas such as healthcare, tax payments, or other domains.

The technology described in this paper is not merely a proposal. Rather, the financial services industry has implemented the technology thereby securing millions of dollars of transactions every day. We call this technology Partner Key Management (PKM). Since wholesale banking permits transactions of ultra-high value, we believe that a demonstration within the wholesale banking domain validates an extension to many other business domain. The prevailing security solution is traditional Public Key Infrastructure (PKI) [1], but PKI is an ill-fit for interoperable wholesale banking due to insufficiencies in the liability model. The first insufficiency relates the Certificate Authority (CA) – a benevolent party which is a fundamental building block of a PKI. In a high-risk environment, practically no CA has the financial resources required to accept a liability burden associated with multi-million dollar payments. For example, suppose a CA issues a certificate; and a multi-million dollar fraudulent transaction were executed with that certificate. If fault were somehow conferred upon the CA, then few (if any) CAs in the industry today would be willing or able to make their customers whole by reimbursing the lost funds. This paper explains that in a PKI, one needs to trust both the CA and the parties in the corporation authorized to direct the CA to execute actions such as create or revoke certificates. In contrast, in PKM, we need to trust the same parties in the corporation, but we can simply eliminate the CA. PKM shifts trust toward bilateral agreements.

In addition, when one further considers interoperability in a high-risk environment, then PKI's Registration Authority (RA) also tends to fail its liability requirements. In an interoperable environment, all participating parties should accept digital signatures executed using certificates authorized by all RAs. Suppose a fraudulent hundred-million dollar transaction were identified; and an RA were found to have issued a certificate to an adversarial party. Since no RA wishes to subscribe to an unlimited liability model, no RA agrees to make all harmed parties whole.

J.P. Morgan operates a PKM service which directly connects customer payment engines to the bank servers via a file-based communication channel. Customers may

authorize bulks of individual payments in a single file. Originally, J.P. Morgan operated a PKI, thereby requiring its customers to subscribe to certificates issued by them. However, multiple customers had their own certificates on their file servers; and the customers requested interoperability by asking J.P. Morgan to accept the customers certificates. J.P. Morgan responded by creating their first version of a flexible certificate management system which we now call PKM. Over the last few years, several banks improved their certificate management system into its current form as a full-scale PKM model serving thousands of customers throughout the world.

Subsequently, many customers contacted the banks seeking to expand the concept of interoperability beyond file-based transmissions into browser-based channels. SWIFT, a member-owned cooperative of financial institutions that facilitates inter-bank transactions, deliberated upon competing models including both PKI and PKM and ultimately picked a model called "anonymous certificates" based upon PKM.[3]

The key contributions of this paper can be summarized as follows:

1. *Partner Key Management (PKM)*: A new key management protocol for handling interoperable credentials for high-value business transactions. The technical problem is to associate certificates between users and their banks while preserving interoperability and security, and avoiding inter-bank liability.
2. *Flexible revocation models*: PKM features several alternative models for revoking credentials. Businesses may choose a revocation model that is compatible to their business processes and liability requirements. The technical problem is to allow each bank to implement its own governance model without the burden of participating in overly restrictive inter-bank governance standards. Wholesale banking provides an excellent test domain because banks tend to be relatively uncompromising in their governance. Some banks are not willing to sacrifice customers due to burdensome security; other banks are not willing to sacrifice security in order to interoperate with less secure banks; and still other banks operate in countries which have unique, incompatible regulatory requirements.
3. *Partner Key Policy Statement (PKPS)*: PKPS is a policy language for describing the properties and constraints of a digitally signed transaction document. These properties and constraints are mutually agreed between the parties executing the transactions. In PKM, a signer always signs a combination of a statement and a PKPS, and a relying party only accepts a signed statement that includes an acceptable PKPS. In essence, a PKPS is the means to ensure that signed statements are only interpreted in the context determined by the signer.
4. *Formal analysis*: We formally compare PKI and PKM using an access-control logic, and demonstrate that the underlying trust assumptions of both systems are equivalent. More specifically, our analysis shows that PKM operates as securely as a PKI, but without any need for Certificate Revocation Lists, OCSP responders, or other similar revocation infrastructures.

The remainder of this paper is organized as follows. Section 2 introduces the core concepts underlying PKM: the credential-registration process, the credential-validation protocols, and the Partner Key Policy Statement (PKPS) that supports bilateral agreements between corporations and banks. In Section 3, we describe in more detail the XML structures and the protocols that support these core PKM concepts. In Section 4, we use a formal logic to demonstrate an equivalence and means of comparison between the security afforded by a PKI and PKM. We give an overview of related technologies in Section 5 and conclude in Section 6.

2. Partner key management concepts

In this section, we introduce the core concepts of Partner Key Management (PKM). We defer to Section 3 a description of how these core concepts can be implemented and used in practice.

2.1. Credential registration

PKM includes a three-step credential-management process, which is illustrated in Fig. 1. In the remainder, the paper uses the term corporation to refer to a wholesale bank's customer.[4] In Step 1, the user (e.g., the corporate cash manager) obtains a credential, from either the corporation itself or a third-party credential provider. In Step 2, the user contacts one of its banks with a request to register the credential (i.e., to establish an association between the userid and the registered credential). In Step 3, the bank determines whether or not to register the submitted credential. The basis for this determination depends upon the individual bank's policy: typically, a security administrator at the corporation contacts the bank for verification that the credential in fact should be registered. This paper assumes the credential is a certificate. In PKM, the bank extracts the thumbprint from the certificate (message digest of the certificate), and builds a registration database that maps each userid onto its registered thumbprint. The distinguished name (DN) contained in the certificate plays no significant role in PKM. After the third step, the bank registers the employee's credential. The employee should repeat the credential registration process with each bank with which his corporation needs to work.

Although a corporation's employee may register his credentials with any bank with which the corporation conducts business, the credential cannot be used until bank approves the registration in Step 3 after consulting the corporation. If an employee chooses to abuse his or her privilege by registering a credential at an unapproved bank, then the corporate security administrator should not approve the unauthorized registration in Step 3. The multi-bank registration process realizes the goal of credential interoperability, because the user may employ the

[3] http://www.swift.com/products/3skey.

[4] Customers of wholesale banks are typically large corporations or governments.

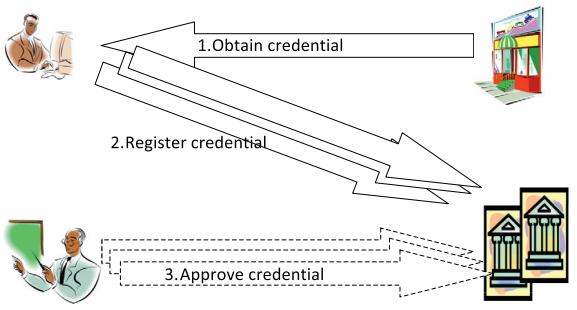

Fig. 1. Credential registration in PKM.

same credential with multiple institutions. However, the protocol does not provide the unneeded identity interoperability, because no requirement exists forcing all the banks to recognize a single userid or impose a universally recognized distinguished name.

Under PKM, a financial institution may employ an appropriate manual bootstrap procedure to register the administrative credentials used in Step 3. PKM itself does not require any of the three steps to be secured in any particular way. Rather, each bank has the freedom to impose its own security requirements and protocol without restriction of an interbank standard or governance model. Therefore, no financial institution suffers due to another financial institution's respective security shortcomings, because each financial institution's registration process has no dependency upon any other financial institution.

PKM imposes few constraints upon the type of credential, and a corporation may use one of the following: certificates issued by a certificate authority, self-signed certificates, or one-time passwords.

This paper focuses on credentials that participate in digital-signature processes in wholesale banking. However, PKM may also register credentials that do not contribute to digital signatures or that are used outside the financial services sector, such as a restriction of the IP address from which a user may connect or a registration of a SIM card for a mobile phone. At the conclusion of the credential registration process, the credential *speaks for* the user. That is, when the bank receives a transaction signed by the credential, the bank understands that the user authorizes the transaction's execution. PKM also assumes an analogous un-registration process. An authorized representative of the corporation may instruct the bank to stop accepting a previously registered credential.

PKM directly addresses the liability and credential-administration business requirements. Because no bank relies upon another bank or other entity's registration process, the inter-bank liability issue evaporates. As for credential administration, the inherent limitations that affect a benevolent trusted party do not apply to banks. The banks have the freedom and autonomy to implement any authorization process that they choose, and each bank may require as many or as few Step 3 authorizing parties as it wants. Since each bank needs to closely interact with its own customers anyway, in order to manage user privileges, the additional administration burden imposed by PKM may be minimal.

2.2. Partner key policy statements

Banks participating in the PKM model publish one or more XML [8] documents called the Partner Key Policy Statement (PKPS), which comply to the WS-Policy [22] XML schema. A PKPS defines how a corporation and a bank agree to work together, as governed by their mutually agreed security operating rules. The corporation and the bank may impose any conditions that can be expressed using PKPS.

Fig. 2 contains a flowchart detailing the steps for validating a PKPS. A user's transaction request comprises a signed transaction document and a signed PKPS. The bank compares the incoming PKPS against its PKPS repository to find a match. If the bank does not find a matching PKPS, then it rejects the transaction returns an error to the user. Through an offline process, the bank and the user must correct their misunderstanding before the bank may consider any signed transactions for further processing. If the bank finds a matching PKPS, then the bank checks if the thumbprints of the certificates used to sign the transaction match the thumbprints registered for the respective signatories. Then, the bank cryptographically validates the signatures on the PKPS. If the signature validation succeeds, then the bank processes the transaction further including transaction signature validation. In effect, a bank considers a PKPS validated only if the bank finds a matching PKPS pre-registered for the signatories' corporate.

A given PKPS specifies a collection of policies to which the user and the banks must agree. The PKPS may include any of the following specific policies:

1. *Credential media:* The definition of the credential media may mandate a particular FIPS-140-2 [16] level of protection.
2. *Credential provider:* This item contains the list of credential providers to which the corporation and the bank mutually subscribe. Example providers are third party trusted providers, self-signed certificates, or the corporation's or the bank's own provider.
3. *Revocation:* The revocation definition describes the type of permissible credential revocation mechanism, such as a certificate revocation list (CRL) or an online certificate status protocol (OCSP) [20]. The revocation definition also describes the party responsible for enforcing credential revocation and any specific usage practice. Section 2.3 presents details.
4. *Timestamp:* The timestamp definition defines timestamp rules and the timestamp provider, if any. The timestamp definition may specify either a real-time threshold value (i.e., a limit on how long past

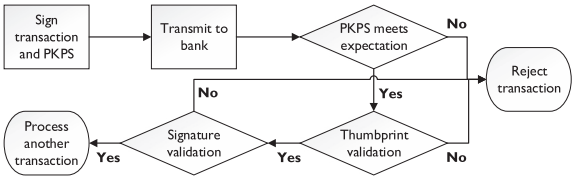

Fig. 2. Signature processing in PKM.

the timestamp a signature can be validated) or a real-time constraint.

5. *Signature policy:* The PKPS can specify the number of signatures required for a specific type of transaction, as well as the roles of signatories. For example, a signature policy may require both an individual signature and a corporate "system" signature to be present.
6. *Credential technology:* The credential-technology section specifies the standards and agreements that must be used, such as X.509 certificates [14] or PGP certificates [9]. PKPS additionally opens the possibility of technology advancements by allowing banks and customers to agree to use technologies that have not yet been submitted for global standardization. For example, some J.P. Morgan wholesale bank customers use the Portable Security Transaction Protocol (PSTP) to sign their transactions [4].

Fig. 3 illustrates possible scenarios where one or more entities recognize a single PKPS. It also shows a natural progression that industries may take in terms of supporting credential interoperability. In Scenario (a), a single bank defines its own, unique PKPS. The bank informs its customers that it rejects all incoming signatures that contain either no PKPS or a PKPS that differs from its expectation. The bank's customers benefit from a limited form of credential interoperability: they can use their credential at each bank that handles PKM, even if different banks do not recognize the same PKPS. any or all of the items covered by a PKPS. In reaction to market pressure, a group of banks may decide to band together, harmonize their differences, and agree to recognize a common PKPS, as in Scenario (b) of Fig. 3. We call this group of banks an "island of interoperability," because interoperable governance exists only within the island. Scenario (c) reflects a larger, nationwide island of interoperability, in which national governments (such as Korea and Brazil) mandate a credential governance model across all wholesale banks serving their nation. Proceeding further, like-minded nations such as the Nordic region may band together to form a very large island, as in Scenario (d). The global interoperable governance of Scenario (e) is unlikely in the near future, but we may consider it as a distant, albeit elusive possibility. PKM allows each industry to progress toward Scenario (e) at its own rate. Market pressures—as opposed to governmental fiat—dictate the relative speed at which the banks must work toward improved interoperability.

Brazil, Korea, and the Nordic nations are all examples of nations or regions that have large-scale interoperable PKIs today. If these PKI regions were to upgrade to PKM, then they could potentially extend their reach beyond the current boundaries while addressing their own inherent deficiencies in liability handling or user administration. Alternatively, a bank that adopts PKM could work in any of these regions by accepting the region-specific certificates through the PKM process.

2.3. Example revocation models

One aspect of a PKPS that merits special attention is revocation. We present four example revocation models, which are illustrated in Fig. 4.

1. *Receiver validation:* The receiver-validation model is typically used in a PKI. First, Alice submits a signed transaction to the bank. Upon receipt, the bank validates the certificate employed in the signature against a CRL or OCSP responder managed by the certificate provider.
2. *Sender validation without evidence:* Alice submits signed transactions to the bank, but the bank performs no revocation check other than looking to see if Alice's credential has been registered but not unregistered.
3. *Sender validation with evidence:* Alice submits her certificate to an OCSP responder, and obtains a response signed by the OCSP responder. Alice signs both the transaction and the OCSP response, which she then submits to the bank. The bank validates both Alice's signature and the OCSP responder's signature. If the bank finds no error, then the bank accepts the transaction.
4. *Sender validation with cosign:* A signer's signature must have an accompanying cosignature. Alice first signs a transaction then routes the signed transaction to a central corporate facility for a cosignature. The central corporate facility validates Alice's identity and ensures that her credentials are current and valid before executing the cosignature.

Each bank has the opportunity to allow any of the example models or to build its own revocation model.

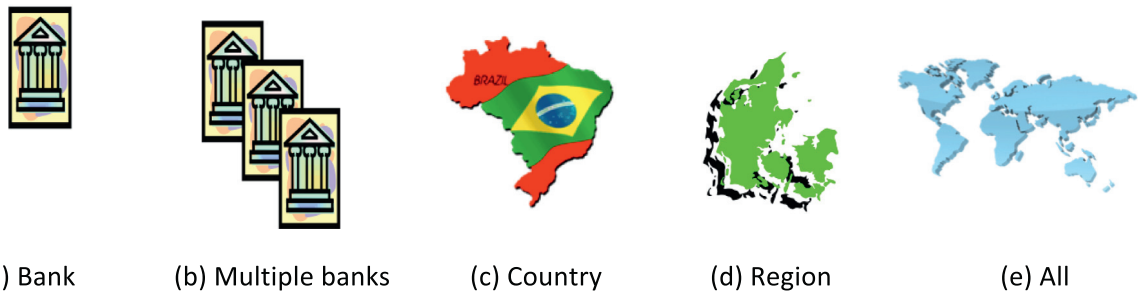

Fig. 3. Islands of interoperability.

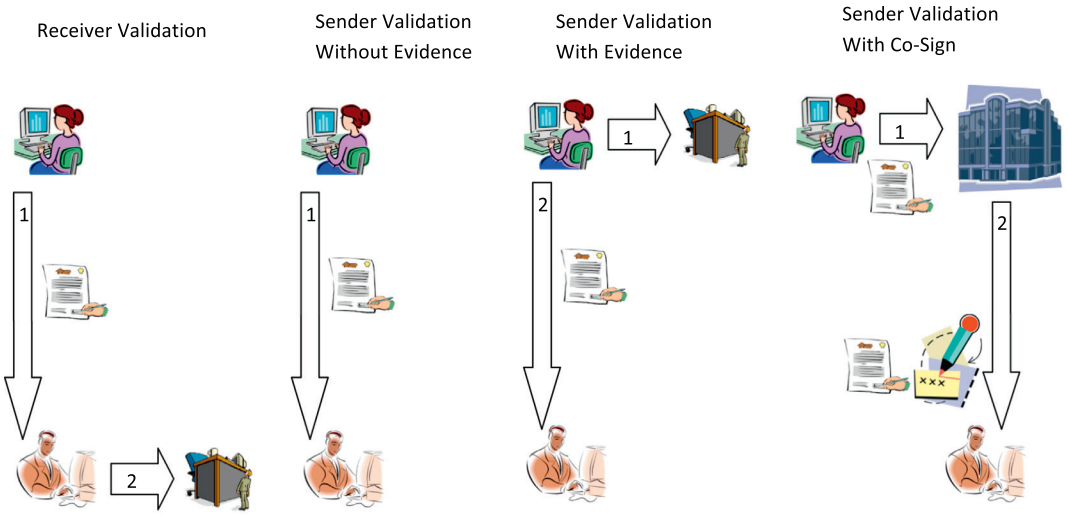

Fig. 4. Validation models.

PKM permits governmental autonomy: multiple banks can all accept the same credential from Alice while simultaneously establishing their own governance rules (e.g., required PKPS structures or revocation models).

Receiver validation is the most common revocation type in the industry today, because most PKIs support it. However, receiver validation is not a good technique to address the agile-marketplace business requirement. If each bank must connect to every CA in order to validate the certificate of every signed transaction, then the availability of a bank is no better than the availability of the CAs. Certificate Revocation Lists (CRLs) may be better than OCSP because the bank can shield itself from minor network disruptions through caching. Nevertheless, if a single bank must connect to many different infrastructures, then the bank cannot provide an adequate Service Level Agreement. Customers might be unsympathetic if a bank were to blame its unexpected downtime upon a CA's lack of service. Furthermore, the cost of connecting to each infrastructure may significantly impede the global marketplace. A connection requires not only development cost, but periodic testing, disaster-recovery planning, audits, and maintenance. These costs would discourage banks from accepting customer requests to use the customer's chosen infrastructure, even if that infrastructure were well-behaved.

In contrast, all three sender-validation models can optimize agility. Each corporation needs to build an on-line connection to either one or zero infrastructures, depending upon the variant of the model. The banks do not need to connect to any infrastructures. In fact, many banks use a manual sender validation method to register the correct keys from partners. In the manual method, a partner sends its correct keys and the evidence of their validity (signed with plain signatures) using courier services or fax to the bank to register the keys. The PKM protocol provides a digital alternative for achieving the same intent. Furthermore, the security of sender validation is as good as receiver validation – see Section 4 for details.

3. Partner key management technology

The wholesale-banking business operates through a network of contractual agreements between corporations, banks, and other financial institutions. Figs. 5 and 6 together illustrate the difference between the network of bilateral contractual agreements that typifies wholesale banking and the hierarchical agreements offered by PKI's benevolent trusted-party model. In this section, we describe the technologies that allow PKM to support this network of bilateral contractual agreements. We begin by introducing the signing mechanisms. We then describe how signatures are used in the four primary revocation models. Finally, we present the overall structure and components of the Partner Key Policy Statement (PKPS).

3.1. Signature binding of PKPS

A PKPS is an XML document that has no inherent protection against unauthorized modification and has no concept of ownership. Consequently, every PKPS should reside within the context of at least one digital signature. The format of the digital signature is outside of the scope of the PKPS specification; however, because the PKPS syntax is typically XML, one would expect the most applicable signature format to be XMLDSIG [3]. Although the PKPS does not constrain whether its associated signatures use the XMLDSIG detached or non-detached formats, we suspect that the detached format may be best for most use cases.

Fig. 7 illustrates an XMLDSIG signature that covers a PKPS. In accordance to the XMLDSIG standard, the reference contains a digest of the referenced document (which in this case is a PKPS), and the signature value covers all the references. Thus, if an adversary were to attempt to modify a PKPS, then either the reference's digest would fail to validate or the SignatureValue computed over a substituted digest would fail. The KeyInfo is an optional XMLDSIG element that provides the keys needed to validate the signature.

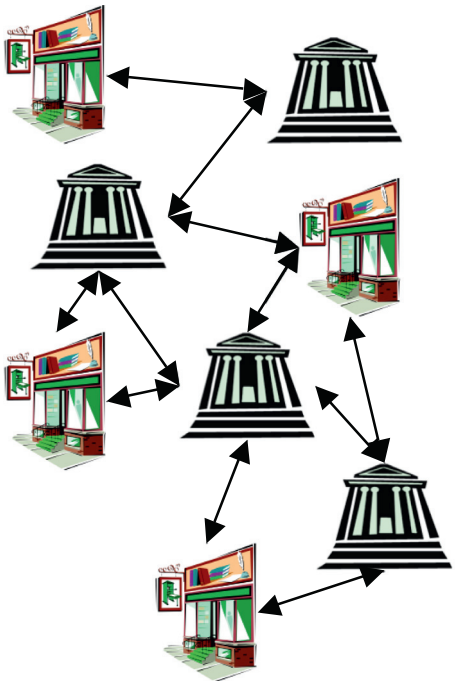

Fig. 5. Bilateral banking agreements.

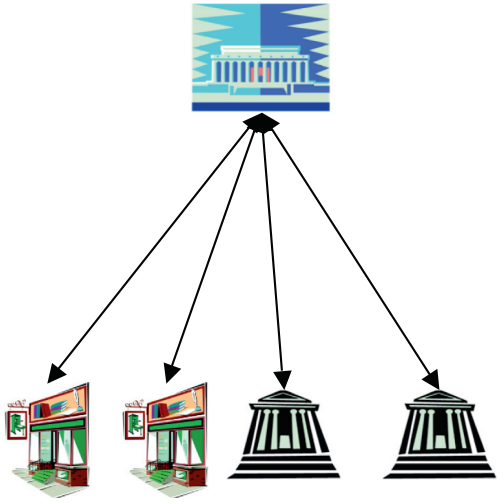

Fig. 6. Hierarchical agreements

3.2. Revocation models

We introduced four revocation models in Section 2.3. We now explain how signatures and signing mechanisms support these different models.

3.2.1. Receiver validation

Fig. 8 illustrates the data structure for the receiver-validation model. The signature on the left of the data structure is the XMLDSIG executed by the user: the user identifies the transaction document that he or she wishes

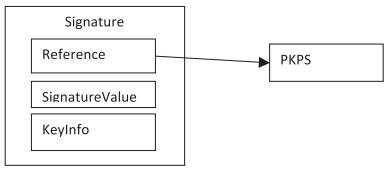

Fig. 7. XMLDSIG signature binding.

to sign and then executes the signature over both the transaction document and the PKPS. The signature on the right illustrates the CA's signature: the CA signs the PKPS and the user's certificate, but not the transaction-level document. Consequently, the CA does not need to wait to see the transaction document and can pre compute its signature. In comparison, the receiver-validation data structure is similar to the data structure that one would use in a standard PKI with X.509 certificates. The advantage is that XML relieves the X.509 certificate of the burden of handling certificate extensions. Instead, the data structure may place the information that one would normally find in a certificate extension into the PKPS (perhaps leveraging an extension of the PKPS schema when necessary). The advantage is that the PKPS uses a more modern XML format, as opposed to the X.509 extension's use of the antiquated ASN.1 syntax.

3.2.2. Sender validation without evidence

The sender-validation-without-evidence model requires only the left half of Fig. 8: the user signs both the PKPS and the transaction document. No CA signature is required in this model.

3.2.3. Sender validation with evidence

The sender-validation-with-evidence model adds additional signature coverage to the receiver-validation model. The bold arrow in Fig. 9 highlights the sole conceptual difference between the receiver-validation and sender-validation-with-evidence models: the transaction is signed by both the corporate user and the OCSP responder. The user first signs both the PKPS and the transaction document, as shown on the left of Fig. 9. The user then sends both her certificate and a message digest of the transaction to the OCSP responder. If the OCSP responder considers the certificate to be currently valid, then the OCSP responder signs the message digest and send this signature to the user. The user may add this signature to the data structure and then submit the entire structure to the intended receiver. In order to optimize operations, the CA may elect to use different keys to sign the certificates and OCSP responses. In this case, the data structure of Fig. 9 would be more complex, but it would serve the same purpose.

3.2.4. Sender validation with cosign

The sender-validation-with-cosign data structure appears in Fig. 10. The three-step signature process starts with a user who signs a transaction document and then sends the signature to a centralized automated validator in the corporate data center. In the second step, the automated validator consults human-resource records or other

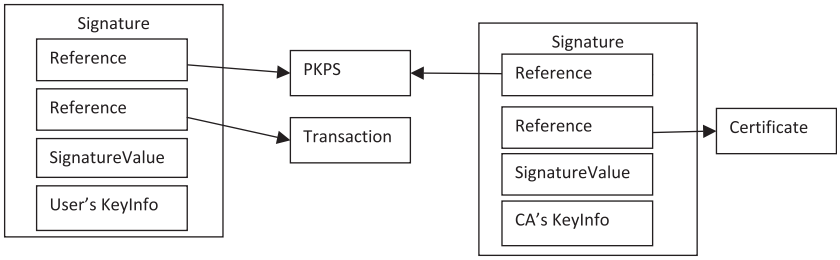

Fig. 8. Signature structure for receiver validation.

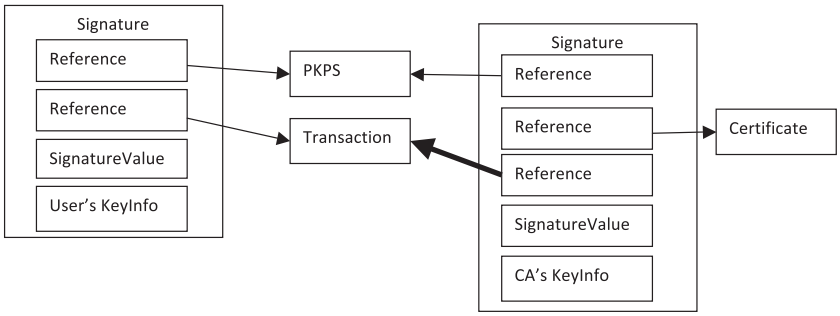

Fig. 9. Signature structure for sender validation with evidence.

facilities to verify the validity of the user's credential. If the credential is valid, then the automated validator countersigns to indicate the current validity of the user's credential and forwards it to the bank. In effect, the corporation asserts a limited scope acknowledgment: the corporation agrees that the user signed the transaction with a valid key. However, the corporation does not necessarily view or acknowledge the transaction details. In the third step, the bank validates both signatures. The user's signature indicates an agreement to the transaction document; the automated validator's signature indicates an agreement to the current validity of the user's credential.

The purpose of this revocation model is to replace trust in a benevolent third party with contract law. If the corporation were to lie by signing a transaction inappropriately, then the corporation would be in breach of contract. Furthermore, a corporate lie would be against the corporation's best interest, because it would permit signatures using certificates that should no longer be valid.

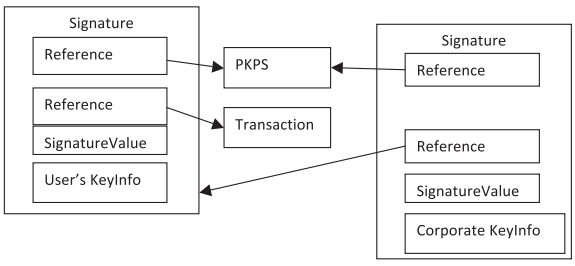

Fig. 10. Signature structure for sender validation with cosign.

4. Formal analysis with examples

This section provides a formal comparison between the PKI and PKM trust models. We use an access-control logic to highlight the underlying trust assumptions and the operations required to validate the transaction-signing keys in both the PKI and PKM models. Section 4.1 serves as a brief primer on the access-control logic that we use for our analysis. We introduce a small scenario in Section 4.2 that serves as the basis for our analysis. Section 4.3 provides a high-level comparison of PKI and the four PKM revocation models. Sections 4.4–4.6 formally express the PKI and PKM models with respect to this scenario.

4.1. Access-control logic

To reason formally about the PKI and PKM models, we use the access-control logic described in [13] and previously used to reason about retail payment systems [12]. Section 4.1.1 describes the syntax, Section 4.1.2 describes the semantics, and Section 4.1.3 describes the inference rules, and Section 4.1.4 describes how to use the logic to express important concepts in our scenario, such as statements, certificates, jurisdictions, and delegation.

4.1.1. Syntax

4.1.1.1. Principal expressions. Let P and Q range over a collection of principal expressions. Let A range over a countable set of simple principal names. The abstract syntax of principal expressions is:

$$P ::= A \mid P\&Q \mid P \mid Q$$

The principal $P\&Q$ ("P in conjunction with Q") is an abstract principal making exactly those statements made by both P and Q; $P \mid Q$ ("P quoting Q") is an abstract principal corresponding to principal P quoting principal Q.

4.1.1.2. Access control statements. The abstract syntax of statements (ranged over by φ) is defined as follows, where P and Q range over principal expressions and p ranges over a countable set of *propositional variables*:

$$\varphi ::= p \;/\; \neg\varphi \;/\; \varphi_1 \wedge \varphi_2 \;/\; \varphi_1 \vee \varphi_2 \;/\; \varphi_1 \Rightarrow \varphi_2 \;/\; \varphi_1 \equiv \varphi_2 \;/\; P \Rightarrow Q \;/\; P\,\mathsf{says}\,\varphi \;/\; P\,\mathsf{controls}\,\varphi \;/\; P\ \mathsf{reps}\ Q\ \mathsf{on}\ \varphi$$

Informally, a formula $P \Rightarrow Q$ (pronounced "P speaks for Q") indicates that *every* statement made by P can also be viewed as a statement from Q. A formula $P\mathsf{controls}\varphi$ is syntactic sugar for the implication $(P\mathsf{says}\varphi) \Rightarrow \varphi$: in effect, P is a trusted authority with respect to the statement φ. P reps Q on φ denotes that P is Q's delegate on φ; it is syntactic sugar for $(P\mathsf{says}(Q\mathsf{says}\varphi)) \Rightarrow Q\mathsf{says}\varphi$. Notice that the definition of P reps Q on φ is a special case of controls and in effect asserts that P is a trusted authority with respect to Q saying φ.

4.1.2. Semantics

Kripke structures define the semantics of formulas.

Definition 1. A *Kripke structure* $\mathscr{M}$ is a three-tuple $\langle W, I, J\rangle$, where:

- W is a nonempty set, whose elements are called *worlds*.
- $I : \textbf{\textit{PropVar}} \to \mathscr{P}(W)$ is an *interpretation* function that maps each propositional variable p to a set of worlds.
- $J : \textbf{\textit{PName}} \to \mathscr{P}(W \times W)$ is a function that maps each principal name A to a relation on worlds (i.e., a subset of $W \times W$).

We extend J to work over arbitrary *principal expressions* using set union and relational composition as follows:

$$J(P\&Q) = J(P) \cup J(Q)$$
$$J(P \mid Q) = J(P) \circ J(Q),$$

where

$$J(P) \circ J(Q) = \{(w_1, w_2) \mid \exists w'.(w_1, w') \in J(P) \text{and} (w', w_2) \in J(Q)\}$$

Definition 2. Each Kripke structure $\mathscr{M} = \langle W, I, J\rangle$ gives rise to a function

$$\mathscr{E}_{\mathscr{M}}[\![-]\!] : \textbf{\textit{Form}} \to \mathscr{P}(W),$$

where $\mathscr{E}_{\mathscr{M}}[\![\varphi]\!]$ is the set of worlds in which φ is considered true. $\mathscr{E}_{\mathscr{M}}[\![\varphi]\!]$ is defined inductively on the structure of φ, as shown in Fig. 11.

Note that, in the definition of $\mathscr{E}_{\mathscr{M}}[\![P\mathsf{says}\varphi]\!]$, $J(P)(w)$ is simply the image of world w under the relation $J(P)$.

4.1.3. Inference rules

In practice, relying on the Kripke semantics alone to reason about policies and behavior is inconvenient. Instead, structure inference rules are used to manipulate formulas in the logic. All logical rules must be sound to maintain consistency.

$$\begin{aligned}
\mathscr{E}_{\mathscr{M}}[\![p]\!] &= I(p)\\
\mathscr{E}_{\mathscr{M}}[\![\neg\varphi]\!] &= W - \mathscr{E}_{\mathscr{M}}[\![\varphi]\!]\\
\mathscr{E}_{\mathscr{M}}[\![\varphi_1 \wedge \varphi_2]\!] &= \mathscr{E}_{\mathscr{M}}[\![\varphi_1]\!] \cap \mathscr{E}_{\mathscr{M}}[\![\varphi_2]\!]\\
\mathscr{E}_{\mathscr{M}}[\![\varphi_1 \vee \varphi_2]\!] &= \mathscr{E}_{\mathscr{M}}[\![\varphi_1]\!] \cup \mathscr{E}_{\mathscr{M}}[\![\varphi_2]\!]\\
\mathscr{E}_{\mathscr{M}}[\![\varphi_1 \supset \varphi_2]\!] &= (W - \mathscr{E}_{\mathscr{M}}[\![\varphi_1]\!]) \cup \mathscr{E}_{\mathscr{M}}[\![\varphi_2]\!]\\
\mathscr{E}_{\mathscr{M}}[\![\varphi_1 \equiv \varphi_2]\!] &= \mathscr{E}_{\mathscr{M}}[\![\varphi_1 \supset \varphi_2]\!] \cap \mathscr{E}_{\mathscr{M}}[\![\varphi_2 \supset \varphi_1]\!]\\
\mathscr{E}_{\mathscr{M}}[\![P \Rightarrow Q]\!] &= \begin{cases} W, & \text{if } J(Q) \subseteq J(P)\\ \emptyset, & \text{otherwise}\end{cases}\\
\mathscr{E}_{\mathscr{M}}[\![P\ \mathsf{says}\ \varphi]\!] &= \{w \mid J(P)(w) \subseteq \mathscr{E}_{\mathscr{M}}[\![\varphi]\!]\}\\
\mathscr{E}_{\mathscr{M}}[\![P\ \mathsf{controls}\ \varphi]\!] &= \mathscr{E}_{\mathscr{M}}[\![(P\ \mathsf{says}\ \varphi) \supset \varphi]\!]\\
\mathscr{E}_{\mathscr{M}}[\![P\ \ \mathsf{reps}\ Q\ \mathsf{on}\ \varphi]\!] &= \mathscr{E}_{\mathscr{M}}[\![P \mid Q\ \mathsf{says}\ \varphi \supset Q\ \mathsf{says}\ \varphi]\!]
\end{aligned}$$

Fig. 11. Evaluation semantics, with $\mathscr{M} = \langle W, I, J\rangle$.

$$\textit{Quoting (1)}\ \frac{P \mid Q\ \mathsf{says}\ \varphi}{P\ \mathsf{says}\ Q\ \mathsf{says}\ \varphi} \qquad \textit{Quoting (2)}\ \frac{P\ \mathsf{says}\ Q\ \mathsf{says}\ \varphi}{P \mid Q\ \mathsf{says}\ \varphi}$$

$$\textit{Controls}\ \frac{P\ \mathsf{controls}\ \varphi \quad P\ \mathsf{says}\ \varphi}{\varphi} \qquad \textit{Derived Speaks For}\ \frac{P \Rightarrow Q \quad P\ \mathsf{says}\ \varphi}{Q\ \mathsf{says}\ \varphi}$$

$$\textit{Reps}\ \frac{Q\ \mathsf{controls}\ \varphi \quad P\ \ \mathsf{reps}\ Q\ \mathsf{on}\ \varphi \quad P \mid Q\ \mathsf{says}\ \varphi}{\varphi}$$

$$\textit{Rep Says}\ \frac{P\ \ \mathsf{reps}\ Q\ \mathsf{on}\ \varphi \quad P \mid Q\ \mathsf{says}\ \varphi}{Q\ \mathsf{says}\ \varphi}$$

Fig. 12. Derived rules used in this paper.

Definition 3. A rule of form $\frac{H_1 \cdots H_n}{C}$ is sound if *for all Kripke structures* $\mathscr{M} = \langle W, I, J\rangle$, *if* $\mathscr{E}_{\mathscr{M}}[\![H_i]\!] = W$ *for each* $i \in \{1, \ldots, n\}$, *then* $\mathscr{E}_{\mathscr{M}}[\![C]\!] = W$.

The rules in Figs. 13 and 12 are all sound. If sound rules are used throughout, then the conclusions derived using the inference rules are sound, too.

4.1.4. Expressing statements in logic

With the definition of the syntax and semantics of access-control logic, we provide an introduction to expressing actual payment instructions and the PKPS in logic.

4.1.4.1. Statements and certificates. Principals make statements, including requests; such statements are expressed using the says operator. For example, if Alice wants to issue a payment transaction denoted by Φ_T, then Alice's request is stated as

$$\textit{Alice}\ \mathsf{says}\Phi_T.$$

A certificate is a signed statement. For example, in a PKI, a certificate authority signs a statement associating a cryptographic key with a principal by associating a distinguished name and a public key under the auspices of the certificate authority's signature. The receiver of the certificate in a PKI must ascertain whether the public key contained in the certificate is currently active. For example, suppose that Alice obtains a certificate from *CA*. Alice's key certificate can be formally expressed as

$$\textit{CA}\ \mathsf{says}\ (\langle K_A, \textit{Active}\rangle \Rightarrow K_A \Rightarrow \textit{Alice}),$$

where $\langle K_A, \textit{Active}\rangle$ is a proposition that reflects the status of the key K_A. Informally, CA says if K_A is active, then K_A speaks for Alice; and the receiver requires an extra step

$$\textit{Taut}\ \frac{}{\varphi}\ \text{if } \varphi \text{ is an instance of a prop-logic tautology}$$

$$\textit{Modus Ponens}\ \frac{\varphi \quad \varphi \supset \varphi'}{\varphi'} \qquad \textit{Says}\ \frac{\varphi}{P\ \mathsf{says}\ \varphi}$$

$$\textit{MP Says}\ \frac{}{(P\ \mathsf{says}\ (\varphi \supset \varphi')) \supset (P\ \mathsf{says}\ \varphi \supset P\ \mathsf{says}\ \varphi')}$$

$$\textit{Speaks For}\ \frac{}{P \Rightarrow Q \supset (P\ \mathsf{says}\ \varphi \supset Q\ \mathsf{says}\ \varphi)}$$

$$\textit{Quoting}\ \frac{}{P \mid Q\ \mathsf{says}\ \varphi \equiv P\ \mathsf{says}\ Q\ \mathsf{says}\ \varphi}$$

$$\textit{\&Says}\ \frac{}{P\&Q\ \mathsf{says}\ \varphi \equiv P\ \mathsf{says}\ \varphi \wedge Q\ \mathsf{says}\ \varphi}$$

$$\textit{Idempotency of} \Rightarrow\ \frac{}{P \Rightarrow P} \qquad \textit{Monotonicity of} \mid\ \frac{P' \Rightarrow P \quad Q' \Rightarrow Q}{P' \mid Q' \Rightarrow P \mid Q}$$

$$\textit{Associativity of} \mid\ \frac{P \mid (Q \mid R)\ \mathsf{says}\ \varphi}{(P \mid Q) \mid R\ \mathsf{says}\ \varphi}$$

$$P\ \mathsf{controls}\ \varphi \stackrel{\text{def}}{=} (P\ \mathsf{says}\ \varphi) \supset \varphi$$

$$P\ \mathsf{reps}\ Q\ \mathsf{on}\ \varphi \stackrel{\text{def}}{=} P \mid Q\ \mathsf{says}\ \varphi \supset Q\ \mathsf{says}\ \varphi$$

Fig. 13. Core inference rules.

beyond certificate validation to determine whether K_A is active.

4.1.4.2. Authority and jurisdiction. Jurisdiction statements identify who or what has authority, specific privileges, powers, or rights. In the logic, jurisdiction statements are typically expressed via the controls operator. For example, if a bank believes in the authority of *CA* to issue a certificate, then we write

$$CA\ \mathsf{controls}\ (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice).$$

If *CA* has authority to issue a certificate and subsequently issues that certificate, then the *Controls* inference rule in Fig. 12 allows us to infer the validity of the certificate:

$$\frac{CA\ \mathsf{controls}(\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice) \quad CA\ \mathsf{says}(\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice)}{\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice}.$$

Furthermore, if the bank can verify that $\langle K_A, Active\rangle$ is true (for example, by receiving an OCSP response), then the bank can conclude that $K_A \Rightarrow Alice$ using the following derived inference rule:

$$\frac{\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice \qquad \langle K_A, Active\rangle}{K_A \Rightarrow Alice}.$$

4.1.4.3. Proxies and delegates. In an electronic transaction, the cryptographic key used to sign the transaction serves as a proxy for the principal who submits the transaction to the bank. For example, suppose that Alice uses her key K_A to issue a transaction to the bank. If the bank trusts that the key K_A belongs to Alice (i.e., $K_A \Rightarrow Alice$), then the bank can attribute all statements made using K_A to Alice. Using the *Derived Speaks For* rule in Fig. 12, the bank can deduce that the transaction signed by K_A came from Alice:

$$\frac{K_A \Rightarrow Alice \qquad K_A\ \mathsf{says}\ \Phi_T}{Alice\ \mathsf{says}\ \Phi_T}.$$

In some situations, a principal may be trusted only on specific statements. This notion of constrained delegation is described using the reps operator. For example, if K_A is trusted to be Alice's delegate on the statement Φ_T, then we can write

$$K_A\ \mathsf{reps}\ Alice\ \mathsf{on}\ \Phi_T.$$

The semantics of reps ensures that, if we recognize K_A as Alice's delegate, then we are in effect saying that K_A is trusted on Alice to issue transaction Φ_T. If K_A says Alice says Φ_T, then we write

$$K_A \mid Alice\ \mathsf{says}\ \Phi_T$$

We can then use the *Rep Says* rule from Fig. 12 to conclude that Alice has made the request:

$$\frac{K_A\ \mathsf{reps}\ Alice\ \mathsf{on}\ \Phi_T \quad K_A \mid Alice\ \mathsf{says}\ \Phi_T}{Alice\ \mathsf{says} \Phi_T}.$$

4.2. Sample scenario

To illustrate the similarities and differences among PKI and the PKM revocation models, we introduce a sample scenario where a corporation C sends a transaction Φ_T to a financial institution F. Alice, Bob, and Doug are employees of C and are assigned public keys denoted by K_A, K_B, and K_D, respectively. Alice, Bob, and Doug hold respective roles R_1, R_2, and R_3 that are assigned by C and recognized by F. A benevolent third-party certificate authority *CA* is also used in some instances. In the case of PKM, C and F agree on a PKPS. Fig. 14 summarizes the notation. C and F agree to impose a signature policy that requires F to reject any incoming transaction that does not have signatures from three distinct people acting in the roles R_1, R_2, and R_3 respectively. In the PKM model, C and F can enforce such signature policies using the signature policy section of the PKPS. Fig. 15) describes an XML excerpt of the PKPS that describes the signature policy. In contrast, when using PKI, C and F have to use other proprietary methods for enforcing the policy.

4.2.1. Transaction request

Alice, Bob, and Doug sign a transaction Φ_T using their respective keys K_A, K_B, and K_D and asserting their respec-

Function	Notation
Corporation	C
Financial Institution	F
Alice, employee of C	A
Bob, employee of C	B
Doug, employee of C	D
Alice's role	R_1
Bob's role	R_2
Doug's role	R_3
Alice's key	K_A
Bob's key	K_B
Doug's key	K_D
Certificate authority	*CA*
Mutually agreed PKPS	Φ_{PKPS}

Fig. 14. Notation.

```
<SignaturePolicy>
 <Policy>
  <ExactlyOne>
   <Roles>
    <Role> R1 </Role>
    <Role> R2 </Role>
    <Role> R3 </Role>
   <Roles>
  </ExactlyOne>
 </Policy>
</SignaturePolicy>
```

Fig. 15. Signature policy.

tive roles R_1, R_2, and R_3. The transaction request can be formally stated as follows: (See Table 1).

$$(K_A \mid R_1)\&(K_B \mid R_2)\&(K_D \mid R_3) \text{ says } \Phi_T.$$

4.2.2. Signed PKPS

Additionally, Alice asserts her role R_1 to sign a PKPS document that she includes in the transaction. When F receives the transaction and matches it to Φ_{PKPS} in its repository, the signed PKPS can be formally stated as follows:

$$(K_A \mid R_1) \text{ says } \Phi_{PKPS}.$$

4.2.3. PKPS validation

The inference rule governing the validation of PKPS can be stated as follows:

$$\frac{\begin{array}{c} R_1 \text{ controls } \Phi_{PKPS} \\ Alice \text{ reps } R_1 \text{ on } \Phi_{PKPS} \\ K_A \Rightarrow Alice \\ K_A \mid R_1 \text{ says } \Phi_{PKPS} \end{array}}{\Phi_{PKPS}}$$

The first line in the inference rule states that R_1 is the authorized role for signing the PKPS, and the second line in the inference rule states that Alice is authorized to this role. Both these statements are established through a prior and independent administrative transaction between C and F. The third line is a result of validating K_A, Alice's key. The fourth line is the signed PKPS received. All these four statements, together, help F validate the PKPS.

F will act on the transaction request if it is able to conclude Φ_T. After validating the PKPS, F processes the transaction further. The remainder of this section formally describes these steps. Sections 4.4 and 4.5 provide a case analysis that describes the validation steps when the Φ_{PKPS} specifies each of the PKI and PKM trust models. Finally, Section 4.6 describes the other portions of the PKPS and the inference rule for validating the transaction.

4.3. Comparison of PKI and PKM models

From the bank's perspective, the answers to the following three questions characterize the underlying trust assumptions and operations of both the PKI and PKM key-validation methods:

1. *Who decides when a certificate should be valid?*
   An aspect of commonality between the PKI and the PKM is the authority who determines a key's current validity. In wholesale banking, a corporation's authorized administrators or the key owner determine the key's current status. If the corporation uses a certificate authority such as *CA*, then the corporation's administrators or the key owner instruct *CA* on the key's current status.[5]
2. *Who has authority to quote the corporation on current status of certificate?*
   Both the PKI and PKM need to understand the current validity of a certificate during the validation sequence. However, they may differ in their technical means of discovery. Customarily, PKI employs the receiver-validation model; however, no technical prohibition stops the PKI from adopting sender-validation-with-evidence. In neither of these models do the bank and the corporation directly communicate to discover the certificate's current status. Instead, the corporation communicates the certificate's current status to the bank using the certificate authority. In contrast, PKM's sender-validation-without-evidence and sender-validation-with-cosign models do not use a certificate authority because the corporation directly transmits current status of its certificates to each of the banks without relying upon a middleman.
3. *Who issues the credential?*
   The concept of issuing credentials is very important in PKI, because of the need to secure a trusted distinguished name. If the credential issuer is not trustworthy, then the issuer could potentially provide a certificate marked with a particular distinguished name to the wrong party. In contrast, PKM ignores the distinguished name; and in some models certificate issuance and revocation have a relationship.

The next two sections illustrate the trust assumptions and operations in each of the five trust models in detail.

4.4. Public key infrastructure

This subsection describes the validation process in the case that C and F choose PKI. There are three core trust assumptions in the PKI model, plus a reliance on the receipt of an appropriate certificate:

1. The CA is the authority for issuing the credentials to all the employees of C:

   $$CA \text{ controls } (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice).$$

2. The corporation is the authority for determining the current status of the keys:

   $$C \text{ controls } \langle K_A, Active\rangle.$$

3. *CA* is a delegate of the corporation C for communicating the status of the keys to the financial institution:

[5] For expository purposes, in our subsequent analysis, we focus on the corporation's authority and ignore the user's authority. Accounting for the user's authority requires only small changes to the relevant statements of jurisdiction.

Table 1
Comparison of PKI and PKM from a bank's perspective.

	PKI	PKM-RV	PKM-SVE	PKM-SVNE	PKM-SVCS
Who decides when a certificate should be valid?	C	C	C	C	C
Who has authority to quote C for status?	CA	CA	CA	N/A	N/A
Who issues the credential?	CA	CA	CA	C	C

PKI: Public key infrastructure.
PKM: Partner key management.
RV: Receiver validation.
SVE: Sender validation with evidence.
SVNE: Sender validation without evidence.
SVCS: Sender validation with cosign.
C: Corporation.
N/A: Not applicable.
CA: Certificate authority.

$$CA \text{ reps } C \text{ on } \langle K_A, Active\rangle.$$

Typically, the *CA* maintains an OCSP responder or a CRL to communicate the status of the keys to relying parties such as F. When the *CA* relays a statement from *C* that K_A is active, we write

$$CA \mid C \text{ says } \langle K_A, Active\rangle$$

4. The *CA*-issued certificate asserts that, if the key is active, then the key K_A is associated with Alice:

$$CA \text{ says } (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice).$$

The inference rule for key validation (i.e., concluding $K_A \Rightarrow Alice$) under the PKI model can be formally stated as follows:

$$\frac{\begin{array}{c} CA \text{ controls } (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice) \\ C \text{ controls } \langle K_A, Active\rangle \\ CA \text{ reps } C \text{ on } \langle K_A, Active\rangle \\ CA \text{ says } (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice) \\ CA \mid C \text{ says } \langle K_A, Active\rangle \end{array}}{K_A \Rightarrow Alice}.$$

This rule states that, for the financial institution to conclude that the key K_A speaks for Alice, it must rely on the four trust assumptions and also receive a message from *CA* on *C*'s behalf.

4.5. Partner key management

In this Subsection, we describe the PKM validation process under each of the four revocation models. All four revocation models under PKM share variations of the following two trust assumptions:

1. Credentials are issued either by *C* or *CA*, depending upon the revocation model. The authority for issuing credentials to C's employees can be expressed in the following general form:

$$X \text{ controls } (\langle K_A, Active\rangle \Rightarrow K_A \Rightarrow Alice).$$

In the receiver-validation and sender-validation with-evidence models, *X* is instantiated with *CA*; in the sender-validation without-evidence and sender-validation with-cosign models, *X* is instantiated with *C*.

2. In all cases, the corporation C is the authority for determining the key's status:

$$C \text{ controls } \langle K_A, Active\rangle.$$

In addition to these two trust assumptions, the receiver-validation and sender-validation-with-evidence models require an additional trust assumption:

$$CA \text{ reps } C \text{ on } \langle K_A, Active\rangle.$$

That is, the financial institution must recognize *CA* as a trusted delegate of C with regards to whether the key K_A is active.

The four revocation models vary in how C communicates the status of the keys to F. In the remainder of this section, we describe the details of each of the four revocation models for our scenario.

4.5.1. Receiver validation

Suppose that *CA* uses the OCSP protocol to communicate the status of the keys. Fig. 16 shows the inference rule for this model. The trust assumptions underlying PKI and the PKM receiver-validation model are equivalent. In both models, *CA* issues the keys, and the corporation is the authority on the key's current status and communicates the status using *CA*.

4.5.2. Sender validation with evidence

Fig. 17 shows the inference rule under the sender-validation-with-evidence model. The formalization of this model is identical to that for the receiver-validation model,

$$\frac{\begin{array}{c} CA \text{ controls } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ controls } \langle K_A, Active\rangle \\ CA \text{ reps } C \text{ on } \langle K_A, Active\rangle \\ CA \text{ says } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ CA \mid C \text{ says } \langle K_A, Active\rangle \end{array}}{K_A \Rightarrow Alice}$$

Fig. 16. Receiver validation

$$\frac{\begin{array}{c} CA \text{ controls } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ controls } \langle K_A, Active\rangle \\ CA \text{ reps } C \text{ on } \langle K_A, Active\rangle \\ CA \text{ says } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ CA \mid C \text{ says } \langle K_A, Active\rangle \end{array}}{K_A \Rightarrow Alice}$$

Fig. 17. Sender validation with evidence.

$$\frac{\begin{array}{c} C \text{ controls } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ controls } \langle K_A, Active\rangle \\ C \text{ says } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ says } \langle K_A, Active\rangle \end{array}}{K_A \Rightarrow Alice}$$

Fig. 18. Sender validation without evidence.

because the difference between the two models is purely mechanical and not logical. Using receiver validation, the bank obtains the information directly from the OCSP responder. In contrast, in the sender-validation-with-evidence model, the bank receives the same information indirectly through the corporation. Because the OCSP responder use digital signatures to sign their statements, the corporation cannot forge these statements.

4.5.3. Sender validation without evidence

The sender-validation-without-evidence model removes the benevolent third party from the interaction (and hence CA does not show up in the inference rule). Fig. 18 contains the inference rule for this model. Accordingly, the trust assumptions are similar to those for the previous models, except that there is no proxy relationship in this setting. As opposed to indirectly informing the bank using a third party, C communicates the status of the keys to F directly using the PKM protocol.

4.5.4. Sender validation with cosign

Under this model, whenever Alice signs a transaction, C is also expected to sign Alice's statement. Fig. 19 contains the inference rule for this model. In addition to the standard PKM trust assumptions, the following statements characterize this model:

1. In reference to the operating rules mutually agreed offline between the corporation and the bank, whenever C cosigns statement, C states that Alice's key is currently active without posing additional assertions concerning the validity of Φ_T:

$$\frac{\begin{array}{c} C \text{ controls } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ controls } \langle K_A, Active\rangle \\ C \text{ says } (\langle K_A, Active\rangle \supset K_A \Rightarrow Alice) \\ C \text{ says } K_A \text{ says } \Phi_T \supset C \text{ says } \langle K_A, Active\rangle \\ K_C \Rightarrow C \quad K_C \text{ says } K_A \text{ says } \Phi_T \end{array}}{K_A \Rightarrow Alice}$$

Fig. 19. Sender validation with cosign.

$$C \text{ says } K_A \text{ says } \Phi_T \Rightarrow C \text{ says } \langle K_A, Active\rangle.$$

2. F knows the public key of C, and therefore can attribute any statement signed by K_C to C:

$$K_C \Rightarrow C.$$

3. In a transaction, when C cosigns the transaction request using K_C, we write:

$$K_C \text{ says } K_A \text{ says } \Phi_T.$$

4.6. Transaction model

Under the PKM model, the presence of an appropriately validated PKPS is crucial for the transaction to be accepted. Recall that the PKPS determines the type of statements that C can make in a transaction. We will illustrate this formally using the degenerative signature policy as an example. Fig. 20 describes the inference rule. The top line establishes the C's jurisdiction for signing random number for use in an authentication event, and that a validated PKPS was found in the signed transaction document. The second line states that the validated PKPS implies that C signed a random number to be used for an authentication event. These two rules are the basis for determining that C signed a random number to be used for authentication. Thus, when C attaches a degenerative signature policy, F would only interpret and process the associated payload as signed random number.

For brevity, this section does not describe in detail the formal interpretation for aspects of the PKPS besides the revocation policy. However, their formal analysis bears similarities to the inference rules employed for the signature policy.

The inference rule for the final step of transaction validation can be stated as follows:

$$\frac{\begin{array}{c} (K_A \mid R_1)\&(K_B \mid R_2)\&(K_D \mid R_3) \text{ says } \Phi_T \\ K_A \Rightarrow Alice \quad K_B \Rightarrow Bob \quad K_D \Rightarrow Doug \\ Alice \text{ reps } R_1 \text{ on } \Phi_T \quad Bob \text{ reps } R_2 \text{ on } \Phi_T \quad Doug \text{ reps } R_3 \text{ on } \Phi_T \\ C \text{ controls } R_1\&R_2\&R_3 \text{ controls } \Phi_T \\ C \text{ says } R_1\&R_2\&R_3 \text{ controls } \Phi_T \end{array}}{\Phi_T}.$$

The first line corresponds to the initial transaction request. The second line is the result of validating keys under either the PKI or PKM model. The third line represents the role assignment. The method of assigning roles to individuals is outside the scope of PKM/PKI, but within the scope of each bank's authorization method. The fourth and fifth lines assert that the corporation has jurisdiction over which roles are necessary and sufficient for directing transactions. All these statement are necessary for F to conclude Φ_T.

In conclusion, the PKI and all four revocation models yield the same result: sound reasoning permits F to

$$\frac{\begin{array}{c} C \text{ controls } \langle AuthenticationEvent{:}random\_number\rangle \quad \phi_{PKPS} \\ \phi_{PKPS} \supset C \text{ says } \langle AuthenticationEvent{:}random\_number\rangle \end{array}}{\langle AuthenticationEvent{:}random\_number\rangle}$$

Fig. 20. Degenerative signature policy.

conclude Φ_T and thereby safely process the transaction. Furthermore, upon close inspection of the logical steps required in the demonstration, one can see that the difference in security between PKI and the four revocation models is a mere technicality: the PKI and some of PKM models require the CA as a middleman, and other PKM models do not require a middleman.

5. Related work

This section provides a brief survey of research work related to PKM.

5.1. Public key infrastructure

This paper compares and contrasts PKI and PKM models with respect to the following aspects, namely distinguished name, reliance on trusted third party, delegated administration, revocation models, and scalability.

5.1.1. Distinguished name (DN) and trusted third party

In PKM, the relying party maintains a mapping between each registered user and his certificate thumprint after successful credential registration. Whenever a user signs a message, the bank checks that the thumbprint of the signature's certificate matches the user's registered certificate thumbprint. As a result, PKM has no need for either an independent trusted third party or distinguished name (DN). This is because wholesale banking requires a liability model where no bank relies upon another bank's or benevolent third party's identity management. PKI directly contradicts this requirement because all banks need to recognize a single identity manager's (registration authority's) DN assignments. PKM avoids this fundamental issue by disposing the concept of a universally recognized DN.

5.1.2. Delegated administration

A fundamental requirement of wholesale banking is recognizing administrators who onboard and manage entitlements for corporate employees. In some cases, banks could also delegate this function to their customers. Recall that in Step 3 of the PKM credential registration process (see Fig 1, a corporate administrator should approve the a user's credential registration. This step, by blending authentication and authorization, lends itself to accommodating delegated administration without requiring new infrastructure or trust assumptions.

In contrast, PKI focusses only on authentication – identifying a user, and provisioning a certificate with an appropriate DN. Wholesale banks and their customers have to establish use their infrastructure and mutual trust to support delegated administration.

5.1.3. Delegated administration

PKM first distributes a credential to a user, and second associates the credential with an identity. A user obtains the credential from a single source, but registers the credential with each of its banks independently. PKM integrates into the wholesale bank's authentication system into its authorization system by treating a certificate as an object to which a user may have authorized access. In other words, the PKM protocol authorizes a user to use a particular certificate during the authentication and signature steps. In contrast, PKI separates authentication and authorization as much as possible, so the PKI protocols are independent of a wholesale bank's authorization system.

In contrast, PKI leverages an opposite order of operations by first establishing the identity for an individual, and second distributing a certificate uniquely minted for that individual. The specially crafted certificate contains the user's distinguished name.

5.1.4. Revocation and policy models

PKM offers many different choices for revocation models, and does not require the wholesale banking industry to agree to a single policy. Each wholesale bank may choose the most appropriate revocation model for its purposes. Similarly, each wholesale bank may enforce a different policy without breaking credential interoperability. This is an important requirement because wholesale banks are each subject to different regulations.

PKI imposes a single revocation and policy model. All users must comply with the same certificate practice statement. Moreover, PKI's receiver validation model is a poor fit for wholesale banking interoperability.

5.1.5. Scalability

PKI and PKM differ in the type of scalability they offer. The PKI architecture massively scales to handle global secure e-mail. PKI scalability allows a user to contact a single authority to revoke a certificate while expecting all other e-mail users to automatically recognize that revocation event. The level of scalability required for e-mail far surpasses the scalability required for wholesale banking. While an e-mail user may communicate with hundreds or even thousands of peers, a cash manager only works with banks for which his or her company has accounts (typically not more than ten banks). So, the cash manager can easily contact each of its banks whenever it wants a bank to stop recognizing a particular certificate. Since most wholesale banks today do not observe interoperability, PKM does not impose any new constrains upon its corporate customers that the customers do not already observe. In other words, when a corporate customer experiences compromised credentials today, the corporate customer contacts each of its banks directly.

5.2. Decentralized PKI trust models

Several alternative trust models exist for PKI [21]. The objective of these models is to enable a receiver to validate a certificate issued by CA different from his primary CA. For example, in the bridged PKI model, a central bridge CA cross certifies with several CA's, and this cross-certification enables the customers of these inter-operating CAs to validate certificates issued by any of the other inter-operating CAs. However, bridged PKIs and other similar distributed solutions do not meet the needs of interoperability because no CA or RA assumes universal liability. The bridged PKI exacerbates rather than solves the liability issue. In a traditional PKI, a CA or RA bears responsibility to all parties to

whom it issues certificates. In a bridged PKI, the CA and RA additionally assume responsibility to foreign users known indirectly through peer CAs. So, contractual liability may be tenuous. The bridged PKI additionally complicates governance. For example, unless all the PKIs agree upon common governance rules, customers may need to subscribe to a complex myriad of revocation infrastructures. In contrast, in a PKM, registration and unregistration bears little more complexity than their respective analogs in password management.

5.3. Single sign on

Single-sign on systems provide a federated identity credential such that a user who authenticates once using the credential (typically a username and password) can access several servers without needing to re-authenticate. The Security Assertion Markup Language [11] (SAML) is the most popular SSO standard. In addition, there are also novel proposals for single-sign on credential systems to balance security and privacy [6].

The most significant shortcoming of single-sign on is that it does not offer digital signatures. Requirements in ultra-high trust environments such as wholesale banking extend beyond authentication into audit by mandating after-the-fact evidence for each transaction to support non-repudiation. Digital signatures are essential for non-repudiation. A single-sign on system may possibly integrate digital signatures, but still it introduces the need for a trusted third party and has the same shortcomings as PKI.

5.4. Privacy-preserving credential management

Several approaches exist for provisioning and managing privacy-preserving credentials [10,7]. For example, *Idemix* is a credential system that provides accountable anonymity, i.e., a user's anonymity is preserved as long as he does not perform any inappropriate action [10]. PKM's focus is helping a bank ascertain that an authorized cash manager executed a transaction, therefore privacy and anonymity are outside the scope of PKM.

5.5. Trust negotiation

Trust negotiation [5,24,25,23,18] is an approach for establishing trust between two parties for online transactions. The negotiation typically exchanges credentials and attributes. PKM and trust negotiation have two differences. First, wholesale banks and their customers have a pre-existing trust relationship established through out-of-band contractual agreements. Second, PKM meets interoperable wholesale-banking business requirements such as attributing liability and compatibility with contractual laws, but these issues are outside the scope of trust negotiation.

5.6. Other XML standards

A PKPS is a set of constraints upon credentials. We chose WS-Policy [22] for our implementation, because it is better suited to express these constraints than the alternatives of WSPL [2], XACML [19], X.509 extensions, and P3P [15]. WS-Policy is a W3C standard for specifying web-service policies for security, quality of service, messaging, and other non-functional requirements. WSPL has similar abilities, but it is not an accepted W3C standard.

XACML is a declarative language for specifying access-control policies governing authorization. PKPS policies can be crafted like access-control policies using XACML. Because PKPS policies express a set of simple constraints for authentication, we do not need a complex authorization policy framework such as XACML. WS-Policy is a better match because it is a widely adopted standard for expressing similar constraints for web services. Moreover, expressing PKPS policies in WS-Policy makes it straightforward to verify the correctness of the policies.

X.509 extensions use ASN.1 notation for specifying constraints on certificates, but they do not enjoy universal acceptance. Moreover, XML provides a more modern format that is more readable than ASN.1. P3P [15] is language designed for expressing privacy preferences and is not suited for expressing PKPS constraints.

The key management interoperability protocol (KMIP) [17] defines a standard protocol and API for requesting, delivering, and managing the life-cycle of cryptographic keys (both asymmetric and symmetric keys). In effect, KMIP seeks to standardize the storage and use of cryptographic secrets in an enterprise through a common API. This goal, although tangential, is complementary to PKM. For example, a bank may leverage KMIP for managing the credentials registered by its customers such as keys and their thumbprints.

6. Conclusion

This paper provides the following technical contributions:

1. *Partner Key Management (PKM)*: An key management protocol that addresses interoperability and security requirements without introducing irreconcilable liability concerns. We built PKM out of necessity in the wholesale banking domain, but other business domains can also adopt PKM.
2. *Flexible revocation models*: Methods for managing keys have security, scalability, and liability implications for businesses. Thus, PKM provides a set of revocation models to cater to a diverse set of needs. These models also provide a means to dispense with Certificate Revocation Lists, OCSP responders, and other troublesome revocation infrastructures that inhibit interoperability.
3. *Formal analysis*: We use an access-control logic to formally compare the PKI and PKM revocation models, and demonstrate the security of PKM.

In comparison to PKM, we suspect that PKI will never become the universal interoperable standard. PKI needs CAs and RAs for administration, but liability concerns prohibit interoperability. Interoperable PKI wants customers to subscribe to multiple revocation infrastructures, but

the customers lack the resources for multiple connections. The Internet mandates reliability for many of its subscribers, but PKI providers do not have the resources to match the ultra-high reliability offered by highly trusted parties such as the banks.

Many banks that offer wholesale banking globally implemented PKM. This implementation is a significant achievement because it proves the validity of PKMs legal framework between corporations and a wholesale bank, and scalability of an online PKM credential management system.

References

[1] C. Adams, S. Farrell, T. Kause, T. Mononen, Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP), RFC 4210 (Proposed Standard), 2005.

[2] A.H. Anderson, An introduction to the web services policy language (wspl), in: Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Network, IEEE Computer Society, Washington, DC, USA, 2004, p. 189.

[3] M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon, XML Signature Syntax and Processing, second ed. Technical Report IETF/W3C XML Signature Working Group, 2008.

[4] G. Benson, Portable security transaction protocol, Comput. Netw. 51 (2007) 751–766.

[5] E. Bertino, E. Ferari, A.C. Squicciarini, Trust-x: A peer-to-peer framework for trust establishment, IEEE Trans. Knowl. Data Eng. 16 (2004) 827–842.

[6] A. Bhargav-Spantzel, A.C. Squicciarini, E. Bertino, Establishing and protecting digital identity in federation systems, J. Comput. Secur. 14 (2006) 269–300.

[7] S.A. Brands, Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy, MIT Press, Cambridge, MA, USA, 2000.

[8] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, F. Yergeau, Extensible Markup Language (XML) V1.0. Technical Report W3C XML Core Working Group, 2008.

[9] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, R. Thayer, OpenPGP Message Format, RFC 4880 (Proposed Standard). Updated by RFC 5581, 2007.

[10] J. Camenisch, E. Van Herreweghen, Design and implementation of the idemix anonymous credential system, in: Proceedings of the 9th ACM Conference on Computer and Communications Security CCS '02, ACM, New York, NY, USA, 2002, pp. 21–30.

[11] S. Cantor, J. Kemp, R. Philpott, E. Maler, Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, Technical Report OASIS Security Services Technical Committee, 2005.

[12] S.-K. Chin, S. Older, Reasoning about Delegation and Account Access in Retail Payment Systems, in: MMM-ACNS, 2007.

[13] S.-K. Chin, S. Older, Access Control, Security, and Trust: A Logical Approach, first ed., Chapman and Hall/CRC, 2011.

[14] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 5280 (Proposed Standard), 2008.

[15] L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall, J. Reagle, The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, Technical Report W3C Technology & Society Domain, 2002.

[16] D.L. Evans, P.J. Bond, A.L. Bement, Security Requirements for Cryptographic Modules, Technical Report FIPS PUB 140-2 National Institute of Standards and Technology – Information Technology Laboratory Gaithersburg, MD 20899-8900, 2001.

[17] R. Griffin, S. Sankuratripat, Key Management Interoperability Protocol Profiles Version 1.0, Technical Report OASIS, 2010.

[18] J. Li, N. Li, W.H. Winsborough, Automated trust negotiation using cryptographic credentials, ACM Trans. Inf. Syst. Secur. 13 (2009) 2:1–2:35.

[19] T. Moses, eXtensible Access Control Markup Language (XACML) V2.0. Technical Report OASIS Access Control TC, 2005.

[20] M. Myers, H. Tschofenig, Online Certificate Status Protocol (OCSP) Extensions to IKEv2. RFC 4806 (Proposed Standard), 2007.

[21] R. Perlman, An overview of pki trust models, IEEE Netw. 13 (1999) 38–43.

[22] A.S. Vedamuthu, D. Orchard, F. Hirsch, M. Hondo, P., Yendluri, T., Boubez, U. Yalcinalp, Web Services Policy 1.5 – Framework, Technical Report W3C Web Services Policy Working Group, 2007.

[23] M. Winslett, A.J. Lee, K.J. Perano, Trust negotiation: authorization for virtual organizations, in: Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies CSIIRW '09, ACM, New York, NY, USA, 2009, pp. 43:1–43:4.

[24] L. Xiong, ing Liu, Peertrust: supporting reputation-based trust for peer-to-peer electronic communities, IEEE Trans. Knowl. Data Eng. 16 (2004) 843–857.

[25] T. Yu, M. Winslett, K.E. Seamons, Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation, ACM Trans. Inf. Syst. Secur. 6 (2003) 1–42.

Glenn Benson is a security architect of JPMorgan Chase Treasury Services. His responsibilities include all aspects of security architecture covering hundreds of applications that cumulatively process more than USD 3 trillion daily. He received his PhD from Georgia Institute of Technology and has worked throughout his entire career in the information security industry. He has five patents, and additional patent applications.

Shiu-Kai Chin is a Professor in the Department of Electrical Engineering and Computer Science at Syracuse University. He is the Director of the Center for Information and Systems Assurance. His research includes the application of mathematical logic to the engineering of trustworthy systems. His focus is on access control and policy-based design and verification. Prof. Chin has received several awards for his teaching and scholarly service at Syracuse University. Prof. Chin received his PhD from Syracuse University.

Sean Croston is Executive Director, Senior Product Manager at J.P. Morgan Treasury Services. In this capacity, he is responsible for Channel Access Security, Infrastructure and Product Risk as well as Electronic Bank Account Management (eBAM). Mr. Croston is a frequent speaker at industry conferences on Security & Risk, Electronic Bank Account Management (eBAM) and Digital Interoperability strategy. Mr. Croston has a B.S. in Management from Bentley College.

Karthick Jayaraman is a security engineer at Microsoft. His work focusses on network security monitoring for Windows Azure, Microsoft's public cloud service. He received his PhD from Syracuse University. He has several publications in system security on topics such as web security, access control, and security policy analysis.

Susan B. Older is an Associate Professor in the Department of Electrical Engineering and Computer Science at Syracuse University. Prof. Older's areas of scholarship include programming-language semantics, logics of programs, formal methods, and information-assurance and computer-science education. Prof. Older's scholarly and teaching activities are focussed on the use of mathematics and logic to reason about complex program behavior, such as security and access control, or fairness and concurrency. Prof. Older received her PhD from Carnegie Mellon University.