Identity as a Service: Who Are You?

HotNets VII Submission # 6, 6 pages

1. INTRODUCTION

"Who are you?" — a seemingly very simple question however is not so simple in practice. This is because it often incurs another question: "How can I trust you are who you say you are?" In real life, we prove it by showing our credentials (*e.g.*, national ID, passport, or driver license). However, to prove it in the context of the Internet turns out to be extremely complicated.

The current Internet architecture hides a user's real identity by design, *i.e.*, favors *anonymity*, which is an important reason that leads to the Internet's great success. Censorship-free speech (or behavior) is a typical example. With the Internet, people can freely express their ideas and we can hear voices from every part of the society, which yields to a faster than ever democratic progress. In addition, it leads to an unprecedented thriving of ideas, free thinking, and creativity [4].

However, such an Internet design significantly complicates the way to answer the question "who are you," causing tremendous problems on a daily basis. There are a lot of examples, within which the e-mail SPAM (*e.g.*, [5]) is a typical one: without the ability to identify who sent an e-mail, we accept all e-mails sent to us, the majority of which are typically unwanted.

Although extremely complicated, it does *not* mean there are no solutions to the identity problem. Indeed, as we will show in this paper, we can well answer the question "who are you" in the Internet context by appropriately choosing the way to answer it while retaining the anonymity at the same time. There are at least three ways to answer this question and these three ways tend to cover a large majority of cases for what people really want to know when they ask this question in cyberspace:

- **Answer 1**: *"I am a trustable user."* For example, in email service, a user may actually want to know whether an email is sent from a regular user or a spammer. In P2P service, a user may want to know whether a specific peer is trustable, *i.e.*, not malicious.
- **Answer 2**: *"I am a trustable user. And my unique alias at your site is ..."* For example, for a Web site that prefers a real-name-like user system, it may not actually be interested in each user's real identity, but simply wants to find a way to ensure that each user can only have one login account for management purpose. Then this account is actually the user's unique alias at this Web site.
- **Answer 3**: *"I am a trustable user. And my real identity is ..."* For example, for a sensitive service such as online banking or VPN, the service provider may ask users to register their real identities to ensure security.

Answer 1 and *Answer 2* are crucial to solve "who are you" while retaining users' anonymity at the same time. Unfortunately, the current Internet does not support them. For *Answer 3*, the current Internet provides a good but incomplete solution, that is, digital certificate issuance. The solution is incomplete because its underlying infrastructures (*e.g.*, *X.509* [1] or *OpenPGP*-like *web of trust* [2,6]) have major limitations that impede them to evolve to the *universal* scale (*i.e.*, deployed at different places world-wide and applicable for diversified services). Without the ability to evolve to the universal scale, their effectiveness is significantly limited.

In this paper, we propose an identity solution, *IdNet Mesh*, that supports all the three typical answers. Among the three, *Answer 1* is the key. We support it by designing a common service (of the IdNet Mesh) — *identity validation*, *i.e.*, to validate whether a user is *trustable* or not. Then with slight modification of this service, the solution becomes eligible to support all the three answers.

In addition, we are designing an identity solution that is *evolvable* to the *universal* scale, an important feature upon which its effectiveness relies. At least two issues determine whether a solution is evolvable to the universal scale or not. First, the solution must be widely useful, otherwise service providers will not have strong motivation to deploy it widely. This criterion is satisfied in our case because we support all the three typical answers to "who are you," which can dramatically affect a wide range of applications such as e-mail, P2P, Web security, *etc.* Secondly, a solution must reflect a realistic trust model for entities world-wide, which is an important lesson that we learned from the limitations of existing trust solutions such as X.509 and OpenPGP-like web of trust.

The rest of this paper is organized as follows. In Section 2, we introduce our basic design of the IdNet Mesh. In Section 3, we introduce a universal trust model for the IdNet Mesh and compare it with the trust models of X.509 and web of trust. Next, in Section 4, we introduce four basic services provided by the IdNet Mesh that implement the three typical answers to the question "who are you." Finally, we conclude in Section 5.

2. BASIC DESIGN

2.1 Local Identity Infrastructure — IdNet

We first introduce *IdNet*, the basic building block of our solution. An *IdNet* is a local identity infrastructure administered by a single authority (a bank, a university, or a service provider, *etc*). It provides a common service to public — *identity validation*, *i.e.*, to validate whether a user is a registered user of the IdNet.

Each user can register a unique identity at an IdNet that she trusts most by providing her real identity (real name, national ID number, driver license number, or passport number, *etc*). We call this IdNet the user's *home IdNet*[1]. After registration, the home IdNet issues the user an *Internet passport*, with which the user can access services that require identity validation. During the validation, the user being validated generates a temporary electronic identity using her Internet passport, and the user who validates uses the IdNet service to verify whether the temporary electronic identity is valid, *i.e.*, is associated with a registered user in the IdNet.

Each IdNet consists of two basic components: *IdNet authority* and *IdNet agents*. The IdNet authority is the authority that administers the IdNet. It maintains a central database that stores identity information for each registered user (including information about her Internet passport and real identity). IdNet agents are designed to provide *high scalability* for the identity validation service via large scale replication. Each IdNet agent replicates a hashed copy of Internet passport data (excluding the real identity information) from the central database. We use the hashed copy instead of the original version of user data to ensure *security*. Each agent stores a different hashed copy.

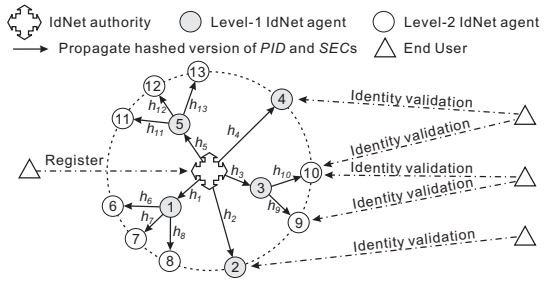

Figure 1: An IdNet with a two-level hierarchy

Internet passport. The IdNet issues each registered user a unique 160-bit *permanent identity* (PID). Along with the PID, the IdNet also issues the user a number of *secret codes* (SEC), each corresponding to one day. Both the PID and SECs are stored in an *Internet passport*, which is a small and cheap hardware that can be plugged into a computer (*e.g.*, via a USB interface). In addition, by embedding an internal clock, the Internet passport enforces a mechanism that the stored SEC for a specific day can be accessed only on that day (*i.e.*, impossible to access in advance). This implementation allows a proper balance between security and cost. The Internet passport expires after a certain period of time (*e.g.*, three years), after which the user should renew it by updating SECs via the IdNet authority.

A *key note*: To simplify presentation, currently we simply treat the Internet passport as such a hardware. Indeed, the Internet passport in *its very essence* is just an abstract device that can store PID and SECs, and can keep them secret. The actual implementation of the Internet passport can have many variants that can provide diversified security and cost options. The actual choice of which type of Internet passport to use will depend on the service context by balancing between security and cost. A software simulated Internet passport is the cheapest one, but most insecure. Yet it could be used for certain insensitive services. Solutions that use secure hardware or biometric techniques (*e.g.*, thumbprint or retina scans) to unlock the PID and SECs could provide higher security, but necessarily at higher cost. They could be used for certain sensitive services where high security outweighs the corresponding cost.

Identity Validation. To make the identity validation service both *scalable* and *secure*, the IdNet authority propagates hashed copies of users' PID and SECs to IdNet agents. The propagation structure is a tree-like hierarchy as exemplified in Figure 1. The root node of the tree is the IdNet authority. The rest nodes are IdNet agents. Each edge in the tree indicates a distinct propagation and each propagation uses a different hash function. The IdNet authority first propagates hashed copies to level-1 IdNet agents, which in turn can propagate hashed copies to level-2 IdNet agents, and so on. For example, the IdNet authority first propagates a hashed copy to agent 1 using hash function h_1. Then agent 1 propagates a hashed copy to agent 6 using hash function h_6. As a result, the hashed copy of PID and SEC being stored at agent 6 becomes $h_6h_1(PID)$ and $h_6h_1(SEC)$.

The identity validation service is provided at the *edge-most* IdNet agents (*i.e.*, leaf nodes of the tree). For each edge-most agent, the IdNet authority issues it a pair of public key and private key. The IdNet announces the public key and the hash function sequence (*e.g.*, $h_6h_1(\cdot)$ for agent 6) associated with each edge-most agent to the public. To do this, it propagates to all agents a list that consists of each agent's valid public key and hash function

[1]It is possible for a user to have multiple home IdNets, but the number is limited (and small). This is because the same user is not likely to bear direct (and strong) trust with too many IdNets such that she is willing to expose her real identity to them. Likewise, an IdNet is not likely to bear strong trust with a user (such that it issues her the Internet passport) for which it has little control.

sequence. Each entry of the list is signed by the IdNet authority. Users can download this list from a randomly selected[2] agent and should keep this list up-to-date[3].

The identity validation process can be formulated by Equations (1)-(4). Below we introduce the main idea of this process.

$$HPID_i = H_i(PID), \quad HSEC_i = H_i(SEC) \tag{1}$$
$$TID = f(HPID_i, nonce, context, PubKey_i) \tag{2}$$
$$HPID_i = g(TID, nonce, context, PriKey_i) \tag{3}$$
$$VC = V(TID, HSEC_i) \tag{4}$$

H_i – the hash function sequence of agent i, equivalent to a composite hash function. f – a function to generate TID from $HPID_i$. g – a function to resolve $HPID_i$ from TID. $PubKey_i$, $PriKey_i$ – the public, private key pair of agent i. V – a cryptographic hash function. H_i, V are implemented based on SHA-1. f and g are implemented using a combination of RSA and SHA-1.

First, the user chooses a suitable agent (denoted by i) and computes her hashed PID and SEC (denoted by $HPID_i$ and $HSEC_i$) that are stored at agent i by using agent i's hash function sequence (using Equation (1)). Then she generates a temporary identity TID based on $HPID_i$, public key of the agent, a nonce, and a 160-bit service context[4] (using Equation (2)). Along with the TID, the user also generates a verification code VC by hashing TID and $HSEC_i$ (using Equation (4)).

Next, the TID and VC are sent to agent i (either directly from the end user or indirectly via relays). From the TID, the agent resolves the user's $HPID_i$ (using Equation (3)), which in turn helps to retrieve the user's $HSEC_i$ (by searching in its database). Then the agent verifies the VC by regenerating it the same way as the user does (using Equation (4)).

Anonymity and Trustability. During the identity validation, the user always keeps her PID secret and what others can see is just the TID. Equation (2) ensures that others are unable to distinguish whether two TIDs observed at two different times or places are associated with the same user, thereby unable to infer the user's identity. In this way, the solution retains each user's *anonymity*.

Under a "legitimate law enforcement reason" admitted by a user's home IdNet (*e.g.*, accusations of child pornography), a user's real identity can be resolved by the *IdNet authority* based on the TID, VC, and the agent used. To do this, the IdNet authority first resolves the user's hashed PID at the agent from the TID (using Equation (3)). Then it resolves the user's real identity by looking up in a table that maps all users' original PID to their hashed PID at the agent. This resolvability serves as the basis for a registered user's *trustability*.

[2]The random selection is designed to effectively mitigate the effect of a compromised agent.

[3]For example, the end user's software can check certain version numbers of the list once every day. If a newer version is detected, it updates the list incrementally.

[4]By exploiting SHA-1, this 160-bit field can encode a service context in form of an arbitrarily long binary string.

2.2 Universal Identity Infrastructure — IdNet Mesh

A universal identity infrastructure can be formed by gradually merging IdNets, and we therefore term this universal infrastructure as *IdNet Mesh*. For example, several IdNets can merge together to form a small IdNet Mesh; later on, several small IdNet Meshes can merge together to form a more universal IdNet Mesh.

High Trust Merging. The first way of merging is to simply merge the central databases of the two IdNet service providers. This is applicable for cases that the two IdNet service providers have strong trusts with each other (*e.g.*, one company have bought the other or two companies merge together thereby forming a new company under a single administration). In addition to the central databases, they also merge the two lists of IdNet agents and propagate the new list to public via infrastructures of both IdNets.

Low Trust Merging. The second way of merging is for more general cases where the two IdNet service providers bear little trusts with each other but simply have a motivation to reuse each other's infrastructure. For such cases, they can merge by propagating to each other's central database a hashed version of users' PID and SECs (real identities and other private information is never propagated beyond a home IdNet). In addition, each IdNet authority endorses the digital certificate of each other such that users of one IdNet can now recognize the digital signature of the other IdNet authority, thereby trusting agents of the other IdNet. Each IdNet propagates this endorsement entry to its users via its *own* infrastructure.

From the perspective of each IdNet authority, the other IdNet authority works essentially the same as one of its level-1 IdNet agents. This reduces risks of the low trust merging to the minimum level. A system fault or a compromised agent that occurs in the other IdNet will not cause security threats on an IdNet's own infrastructure.

A Big Picture of Merging. Figure 2 exemplifies a big picture of merging in which seven IdNets belonging to two countries merge together and form a large IdNet Mesh. IdNet A (or IdNet Mesh A) results from high trust merging of two previously separate IdNets, thereby becoming equivalent to a single IdNet now. IdNet B is similar, resulting from high trust merging of three IdNets. IdNet C merges with both IdNet A and B via low trust merging, thereby forming peering relationship with them. There are also two pairs of IdNets across countries (A and E, B and F) forming peering rela-

tionships via low trust merging; high trust merging might be rare between IdNets of two countries for security or other reasons.

The merging between IdNet D and IdNet A is a special case of low trust merging, in which D propagates its hashed user data to A while A does not do the same to D. Indeed, this indicates a customer-provider relationship between them. We can imagine D as a special IdNet that only has IdNet authority but no agents (*e.g.*, a university that has real identity information of all its students, staff, and faculty). D establishes a customer-provider service contract with A and propagates to A the hashed version of PID and SECs for each user. In this way, D can use A's infrastructure to provide wide-area identity validation service for its users.

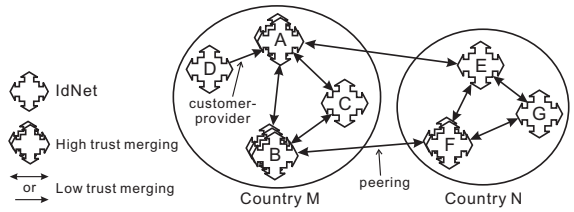

Figure 2: A big picture of merging

Identity Forwarding. In the above scenario, D might also ask A to further relay its hashed user data to B, C and E if A's service agreements with B, C, and E allow this. In this way, D can also use B, C, and E's infrastructures such that D's identity validation service becomes more widely available (even across the country). We call such a relay *identity forwarding*.

Indeed, identity forwarding could be an important approach to facilitate merging among IdNets. Although an IdNet may choose to directly merge with another IdNet instead of having a third IdNets provide identity forwarding for it, using identity forwarding can usually cost much less (*e.g.*, it could be much more costly for C to establish a direct merging contract with the foreign IdNet E instead of having A forward for it). Identity forwarding is a simple version[5] of "policy based routing" problem which has mature solutions [3].

3. UNIVERSAL TRUST MODEL

3.1 The IdNet-Mesh Trust Model

The IdNet Mesh provides a service to validate whether a specific user is a *trustable* user — a registered user of a *trusted IdNet*. In this section, we explain the solution model for an underlying but fundamental question: *How can we trust an IdNet that we previously do not know?*, *i.e.*, the trust model of the IdNet Mesh.

Mutual Initial Trust. The initial trust between a user and her home IdNet is established in a mutual way. The user trusts this IdNet most, therefore she selects it as her home IdNet. The home IdNet trusts the user, therefore it issues the user the Internet passport. This mutual initial trust serves as the starting point of the entire trust model.

Trustee Area. Figure 3 depicts the entire trust model of our design. First, we define the *trustee area* of an IdNet. The *trustee area* of an IdNet is the area that consists of all IdNets that trust this IdNet. For example, the trustee area of IdNet A in Figure 3 consists of IdNets C, D, E, and G. These IdNets trust A by allowing A to propagate its hashed user data to their databases. The propagation is performed either through direct propagation (via high trust or low trust merging, *e.g.*, $A \rightarrow C$ and $A \rightarrow D$) or through identity forwarding (*e.g.*, $A \rightarrow C \rightarrow G$ and $A \rightarrow D \rightarrow E$). The propagation structure can be represented by a spanning tree rooted at A to all other IdNets in the trustee area, *i.e.*, there is a unique propagation route from A to each IdNet.

Trust Area. Secondly, we define the *trust area* of an IdNet. The *trust area* of an IdNet is the area that consists of all IdNets that this IdNet trusts. It is quite different from the trustee area. The trust area is completely defined by each IdNet itself while the trustee area is decided by other IdNets' will. The IdNet *explicitly* expresses its trust by endorsing the digital certificates of other IdNets. In Figure 3, the IdNet B explicitly trusts IdNet C, D, F, and G, thereby defining its trust area.

Note that there is a key difference between our trust model and other models that use *implicit trust* as well (*e.g.*, the *web of trust* approach [6]). The implicit trust assumes a transitive trust among entities, *i.e.*, if entity 1 trusts entity 2 and entity 2 trusts entity 3, then entity 1 is regarded to trust entity 3 as well. Although the implicit trust facilitates trust propagation, it significantly complicates trust revocation, thereby causing uncertainty of trusts (which is a fundamental problem of such trust solutions).

By contrast, our model uses explicit trusts to ensure certainty. Nevertheless, implicit trusts can be used as external mechanisms for the IdNet to establish its explicit trusts. In addition, a trust area is defined on a *per service context* base in our model. An IdNet can explicitly announce different trust areas for different services (*e.g.*, one for email, one for P2P, and one for VPN, *etc*).

Validation Area. Next, we define *validation area*, which is associated with a pair of IdNets. Referring to Figure 3, the validation area *of* A *for* B is the overlapped area between A's trustee area and B's trust area.[6] This area consists of all IdNets through which B's users can validate identities of A's users. B's users admit the identity validation results because these IdNets are within B's trust

[5]It is simple because only "static and unique routes" are used, *i.e.*, no dynamic routing, no parallel routes.

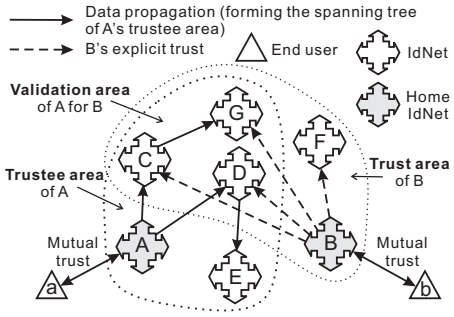

Figure 3: The trust model of IdNet Mesh

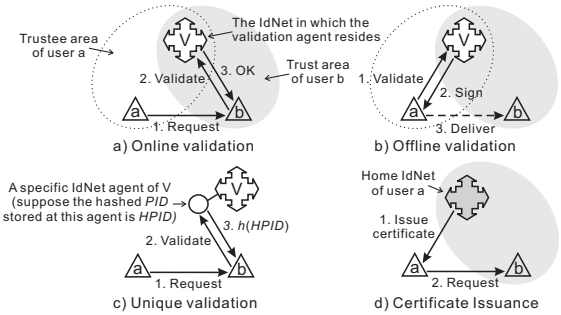

Figure 4: Four basic services of IdNet Mesh

area. The identity validation for A's users can be performed because these IdNets have imported the hashed version of A's user data.

3.2 IdNet Mesh vs. X.509 and Web of Trust

To understand why the IdNet Mesh's trust model is more *realistic* as a universal trust solution, here we briefly compare it with the trust models of *X.509* (a.k.a. *public key infrastructure* and *privilege management infrastructure*) and the *OpenPGP*-like *web of trust*.

The IdNet Mesh shares a flavor of the web of trust in that both exploit a *bottom-up* trust propagation process, which is realistic in terms of the trust evolution nature. On the contrary, the X.509 assumes a strict *top-down* hierarchical structure of trust which relies on a "self-signed" root that is trusted by everyone. The unreality of such a "self-signed" root impedes the X.509 from becoming a universal solution. Currently most X.509 systems stay at enterprise scale.

The IdNet Mesh has a distinct difference from the web of trust. Its trust model provides *deterministic* trusts by prohibiting the use of implicit trust. By contrast, the web of trust fundamentally depends upon the use of implicit trust to ensure effective trust propagation, which leads to *uncertainty* of trusts. Such uncertainty significantly restricts the usefulness of the web of trust. Without wide usefulness, the solution is less attractive, hence unlikely to evolve to the universal scale within moderate period of time (several to ten years).

In addition, the trust model of IdNet Mesh is more realistic to reflect the fact that a trust is always associated with a specific context, *i.e.*, *what* to trust (*e.g.*, in Figure 3, B might trust G for email service, but not for VPN service. B might trust D for all the IdNets that D trusts but excluding E). In our model, this is well addressed by giving each IdNet the full control for definition of its trust area due to the exclusive use of explicit trusts. We leave implicit trusts as external mechanisms to build confidences for an explicit trust.

[6]There is an exception, that is, when B explicitly expresses that it does *not* trust A. In such a case, the validation area of A for B is empty.

4. SERVICES

In this section, we introduce four basic services of the IdNet Mesh as shown in Figure 4: (*i*) *online validation*, (*ii*) *offline validation*, (*iii*) *unique validation*, and (*iv*) *certificate issuance*. These four services correspond to the three typical ways to answer the question "who are you" as we introduced in Section 1. Throughout this section, let's consider the same abstract context: user b asks user a the question "who are you?"

Services (*i*) and (*ii*) correspond to *Answer 1* in which a shows b that she is a trustable user. The difference between (*i*) and (*ii*) is whether there is a direct online communication between a and b. Service (*iii*) corresponds to *Answer 2* in which b not only knows that a is trustable but also knows a's unique alias at b's site. Service (*iv*) corresponds to *Answer 3* in which b also knows a's real identity.

Validation Agent. Before going to the services, we first introduce the concept of *validation agent*, which will be referred to by the first three services. Suppose that user a's home IdNet is A and user b's home IdNet is B. A *validation agent* of a for b is defined as *any* IdNet agent of *any* IdNet within the validation area of A for B.

In the first three services, the trust between a and b can be established using any validation agent. However, there should be an external mechanism for a and b to select a suitable validation agent that both of them agree upon based on their preferences. This mechanism could differ much in different service contexts. Below we will show examples about how to select a validation agent properly when introducing the services.

4.1 Online Validation

In online validation, user a sends her validation data (TID and VC, *etc*) along with the service request to user b. Then b validates a's identity via the *validation agent* by relaying a's validation data. If the validation is successful, b accepts a's service request, otherwise not.

Web Site Protection Example. For example,

b could be a Web site and a could be one of its users; b wants to use online validation to protect itself from malicious users. To select the validation agent in this case, b can provide at the Web site a candidate list of agents within its trust area. This candidate list only includes agents that B prefers (*e.g.*, close to B). Then a can select a suitable agent (within its trustee area) among this candidate list as the validation agent and indicates it in her validation data sent to b.

4.2 Offline Validation

In offline validation, there is no direct online communication between a and b; a wants to indirectly deliver a data object to b, and b wants to validate whether an object is sent from a trustable user. To do this, a encodes the digital fingerprint (*e.g.*, using SHA-1) of the object into the 160-bit service context (as shown in Equation (2)) used to generate the TID. Then a asks the validation agent to verify her identity by sending TID, VC, and the service context. If the validation is successful, the agent returns a a digital signature that certifies the data combination of TID, VC, the service context, and the agent's own identifier. Next, a delivers the data object by embedding into the object the digital signature as well as the data combination that it certifies. Based on the embedded information, b can verify whether the object is sent from a trustable user or not (by checking the consistency among the signature, the certified data combination, and the fingerprint of the object).

E-mail Trusts Example. For example, b could be a user who wants to only read emails from trustable users (such that she can effectively counter SPAMs). Then an email user a can use the offline validation to show that she is trustable. To decide the validation agent, there could be some DNS service extension that can resolve b's mail server's preferred agents from its domain name; a therefore can select a suitable validation agent among them. For popular email service providers (*e.g.*, hotmail or gmail), even the DNS extension is not needed. The providers can simply announce the preferred agents via their Web sites.

P2P Trusts Example. Peer trusts in P2P service is another example. A user can use offline validation to show other users that her peer identifier is trustable. In this case, user a usually does not know b's trust area in advance. Therefore, for the validation agent, a might select a popular agent in her trustee area such that it has a high probability to also reside within b's trust area.

4.3 Unique Validation

Unique validation (Answer 2, Section 1) is a special version of the online validation. In this case, b is usually an enterprise user; b signs a service contract with a popular IdNet to set up the unique validation service. The IdNet assigns to b a fixed validation agent as well as a hash function h. When user a successfully passes a validation, the validation agent sends a hashed version (using h) of user a's hashed PID stored in its database (denoted by $h(HPID)$) to user b in addition to a result code that indicates the success. The $h(HPID)$ then serves as user a's quasi-unique alias at b's site. The alias is quasi-unique because each user can *at most* have a limited (and small) number of aliases.

Real-name Forum Example. For example, b could be an Internet forum Web site that prefers a real-name-like user system — each user has a relatively unique account. Therefore, when a new user a registers an account at b, b can use the unique validation to record a's $h(HPID)$ to a's account. In this way, b can effectively prevent the same user a from opening too many accounts.

4.4 Certificate Issuance

The last basic service of the IdNet Mesh is certificate issuance. In this service, user a registers to a popular IdNet and the IdNet issues a a certificate. As long as user b's trust area covers this IdNet, b can recognize a's certificate and therefore trust a. To address issues of outdated certificates (usually resulting from certificate revocation or reclamation), the IdNet can sign a's certificate verification information (*e.g.*, a version number) and propagate it to all its IdNet agents. In this way, others can verify via these agents whether a certificate is up-to-date.

5. CONCLUSION

In this paper, we proposed IdNet Mesh, a general purpose identity solution for the Internet. The solution provides a widely applicable identity validation service by supporting three typical ways to answer the question "who are you" in the Internet context. In addition, it adopts a much more realistic trust model than existing ones. Both the wide applicability and the more realistic trust model enable it to be (i) widely useful for diversified services and (ii) evolvable towards a universal solution world-wide.

6. REFERENCES

[1] ITU-T Recommendation X.509: Information technology - Open systems interconnection - The directory: Public-key and attribute certificate frameworks.

[2] RFC 4880: OpenPGP message format. Nov. 2007.

[3] M. Caesar and J. Rexford. BGP routing policies in ISP networks. *IEEE Network Magazine*, 19(6):5–11, 2005.

[4] L. Lessig. *The Future of Ideas: The Fate of the Commons in a Connected World*. Random House Inc., Oct. 2001.

[5] A. Ramachandran and N. Feamster. Understanding the network-level behavior of spammers. In *ACM SIGCOMM '06*, Pisa, Italy, Sept. 2006.

[6] W. Stallings. The PGP web of trust. *BYTE*, 20(2):161–162, Feb. 1995.