FREEICP.ORG: FREE TRUSTED CERTIFICATES BY COMBINING THE X.509 HIERARCHY AND THE PGP WEB OF TRUST THROUGH A COLLABORATIVE TRUST SCORING SYSTEM

Marco Antônio Carnut (kiko@tempest.com.br)
Tempest Security Technologies
Centro de Estudos e Sistemas Avançados do Recife - CESAR
Universidade Federal de Pernambuco – CIn/UFPE

Evandro Curvelo Hora (evandro@tempest.com.br)
Tempest Security Technologies
Centro de Estudos e Sistemas Avançados do Recife - CESAR
Universidade Federal de Pernambuco – CIn/UFPE
Universidade Federal de Sergipe – DCCE/UFS

Cristiano Lincoln Mattos (lincoln@tempest.com.br)
Tempest Security Technologies
Centro de Estudos e Sistemas Avançados do Recife - CESAR
Universidade Federal de Pernambuco – CIn/UFPE

Fabio Q. B. da Silva (fabio@cin.ufpe.br)
Universidade Federal de Pernambuco – CIn/UFPE

ABSTRACT

This paper describes a CA hierarchy that mimicks PGP's web-of-trust model using a collaborative web-based trust scoring system to provide free client digital certifcates with strong identity guarantees. Entry-Level CAs that approve temporary short-lived certificates immediately may replace traditional password-based web registration systems; identity guarantees may be added later by passing several qualification rounds in a trust manager web application. When the user exceeds the minimum qualification criteria, he is granted a Verified Identity-class certificate. The system encourages users to tie their digital IDs with their real world IDs, making them more institutionally acceptable and sometimes automatically verifiable; it is argued how this can also provide means of mitigating and managing identity disputes. Experiences gathered from implementing both a CA hierarchy and a relying party web application based on these principles are also presented.

.

1 INTRODUCTION

The hierarchical X.509 PKI [13, 12] and the PGP web of trust [25, 6, 3] have historically been presented as inherently antagonic approaches [2, 8] and extensive discussion has been published about their limitations and unsuitability for global e-commerce [24, 19, 10, 9]. Notwithstanding, these were the only ones to achieve a reasonable level of popularity, as measured by the widespread availability of implementations (PGP [27], GPG [31] and numerous email client plugins), toolkits (OpenSSL [28], Jonah [33], Cryptlib [34], etc.), web server software (Apache [30] + mod_ssl [29]) and clients (IE, Netscape, Mozilla, Opera, etc).

This paper proposes a way to take the best of both worlds, showing one possible way to endow an X.509 hierarchy with a collaborative trust system somewhat like the PGP's web of trust model, but with considerable advantages. In fact, we wanted that the two PKIs user bases could reinforce each other, making our solution also a kind of bridge-CA.

Specifically, we wanted to provide a way for individuals to be identified by means of SSL/TLS client certificates for authentication purposes in web-based applications, but without having to pay for their identities to be verified like in commercial CAs. The solution has been present in PGP since its inception: users vouch for other users' identities by signing their keys, building a distributed, collaborative web of trust [15]. X.509 was not quite designed to support this, so we built a web-based CA application that allows users to introduce each other, assigning numeric scores to the amount of certainty users grant each other – in fact, a generalization of PGP's own trust scores, but with a few novelties: a mechanism for tying their virtual identities to real world identifiers (so as to make them more "institutionally acceptable"); a way to perform many simple identity validation checks automatically; and a method to detect and manage identity disputes, either malicious or not.

We also tried to maintain a few key design principles: the whole CA infrastructure should be implementable with common open-source software (Apache, mod_ssl) and should be usable with the standard popular web browsers (IE, Nescape, etc.). Moreover, they should be made as simple as possible, up to the point of rivalling with common "email & password" web registration schemes.

The rest of this paper is organized as follows: section 2 describes the CA infrastructure at considerable length: the Entry-Level and Verified Identity family of Certificate Authorities, the Trust Manager and the trust scores, along with many experiences from the actual reference implementation. Section 3 details the combination of automatic and human-assisted identity validation procedures used by the Trust Manager to ascertain the users and hosts identities. Particular attention is given to the resilience to misbehavior with a description of the identity contention management scheme. It is also argued that these metrics adhere to good design principles proposed in the literature. We could not judge how well our system would perform without trying it in a real application; section 4 describes our initial experience in adapting an existing application to support our mixed-PKI infrastructure. Section 5 presents conclusions and future work directions.

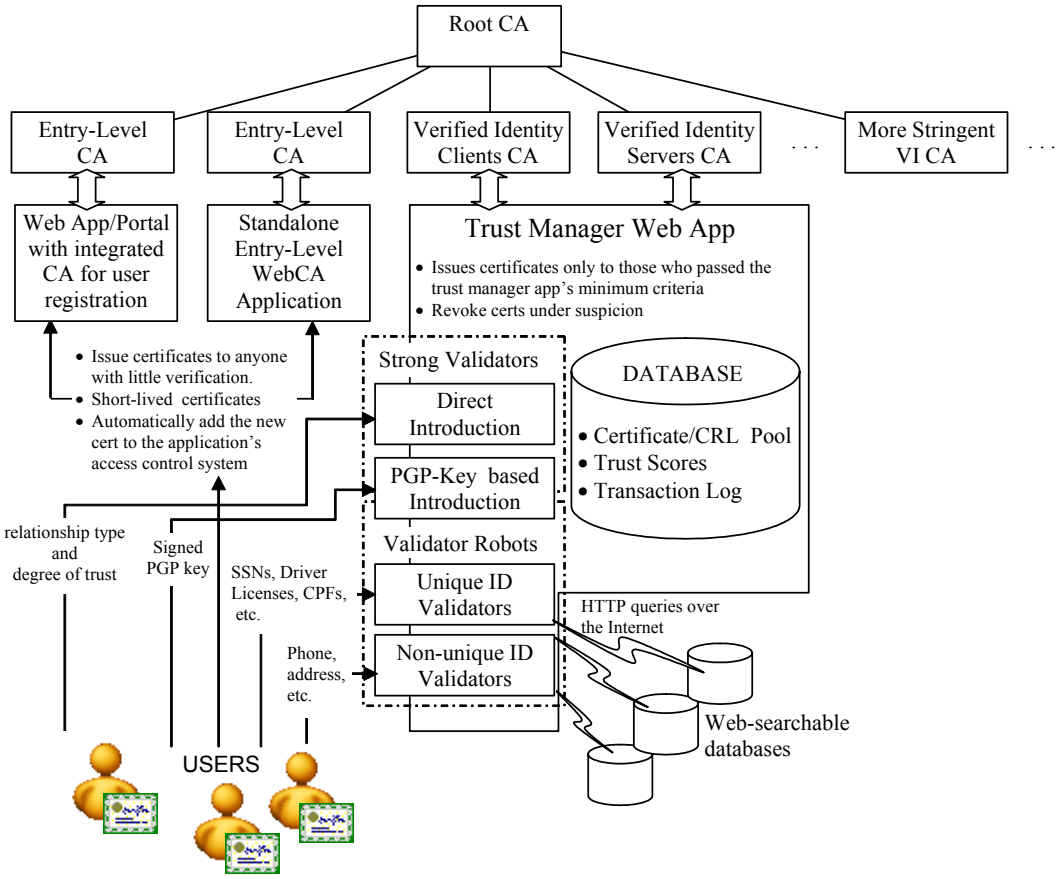

Figure 1: Overall system architecture: above, the X.509 treelike root CA → intermediate CAs key heirarchy. The two main CAs are the Entry-Level and the Verified Identity, associated to their respective web sites. The first issues certificates to nearly any user, just to allow them to log on the trust manager web application. Web portals might have an integrated EL CA as its user registration system. Users are allowed to issue certificates under the Verified Identity CA only when they meet a minimum set of scores. The way to increase them is by passing through several kinds of validators: the validators robots check the users' personal information on web databases; and the strong validators are based on user-to-user introduction. The VI CA also has a PGP keypair and adds his signature to users with PGP keys (when they qualify); and accepts signed PGP keys as strong-validation introductions. This leverages the userbases of both PKIs, helping them to reinforce each other.

2 SYSTEM ARCHITECTURE

2.1 CA and Key Hierarchy

We start by proposing a fairly standard CA hierarchy: a root CA which certifies two families of intermediate CAs:

- **The Entry-Level (EL) CA family:** these CA applications generate certificates online to any user that requests one, with just minimal validation, such as complying with a simple naming policy, avoiding duplicates and challenging the validity of the email address by replying to it. Its sole purpose is to put a valid, working, fully functional digital certificate into the users' client applications – most likely, their web browsers – immediately and for free. These certificates have short validity periods compared with the more trusted ones – two or three months seem sensible. They should be accepted only for testing or initial enrollment in web applications.
- **The Verified Identity (VI) CA:** this CA issues digital certificates when users meet some specific credibility and trustability scoring. These certificates have a larger validity period, something like six to twelve months. Actually, there could be several such CAs, each with successively more stringent scoring requirements. Large-scale production applications should require these certificates for the bulk of their functionality; the applications should "insist" that the users "upgrade" to a VI certificate as soon as possible.

The VI CAs would have both X.509 certificates/private keys and PGP keypairs, so they could act as cross-certifiers. The idea is to leverage each PKI's user base to reinforce each other and foster wider adoption.

2.2 The Entry-Level Certification Authorities

We wanted to make users able to generate a new SSL client certificate just as easily as PGP users can

generate their key pairs. However, most web browsers don't have provisions for properly signing Certificate Signing Requets; and even if they did, we would have to be part of a hierarchy anyway. So, we need a CA; we call it an Entry-Level CA.

We use the term "Entry-Level" to suggest a certificate with no identity guarantees – just like PGP keypairs –, in analogy with "temporary" membership or airline frequent flyer cards. The user should expect that he will be required to change it for a "definitive" one. Applications should grant minimum "guest-like" privileges to accesses made with this certificate.

To maintain parity with PGP, users are identified by their email addresses and a nickname (possibly, but not necessarily, their real names). In our prototype implementation we followed a Google-like UI simplicity principle: the user types in his name and email in a *single form* field and gets the certificate installed in the very next screen.

This ideal has been implemented with reasonable success on Netscape and similar browsers (Mozilla, Opera), as shown in figure 2: from the moment the user hits the submit button to the point the certificate gets installed, those browsers add just a few simple steps to ask or set the private key container's passphrase. Internet Explorer, however, proved much more intimidating: figure 3 shows that even when it's not necessary to update the ActiveX control that handles key/CSR generation, the process may take up to 12 steps with scary messages about scripting violations; and the user interface almost compels the user to store his certificate with no passphrase.

Another catch is that the user has to have already installed our root CA's certificate. In our intranet setting, this is part of our customized OS installation procedure, so our users don't have to perform this step. In other environments, however, this will be needed; although the process is not complex, some browsers present a multi-step wizard with many choices that non-technical users often misunderstand.

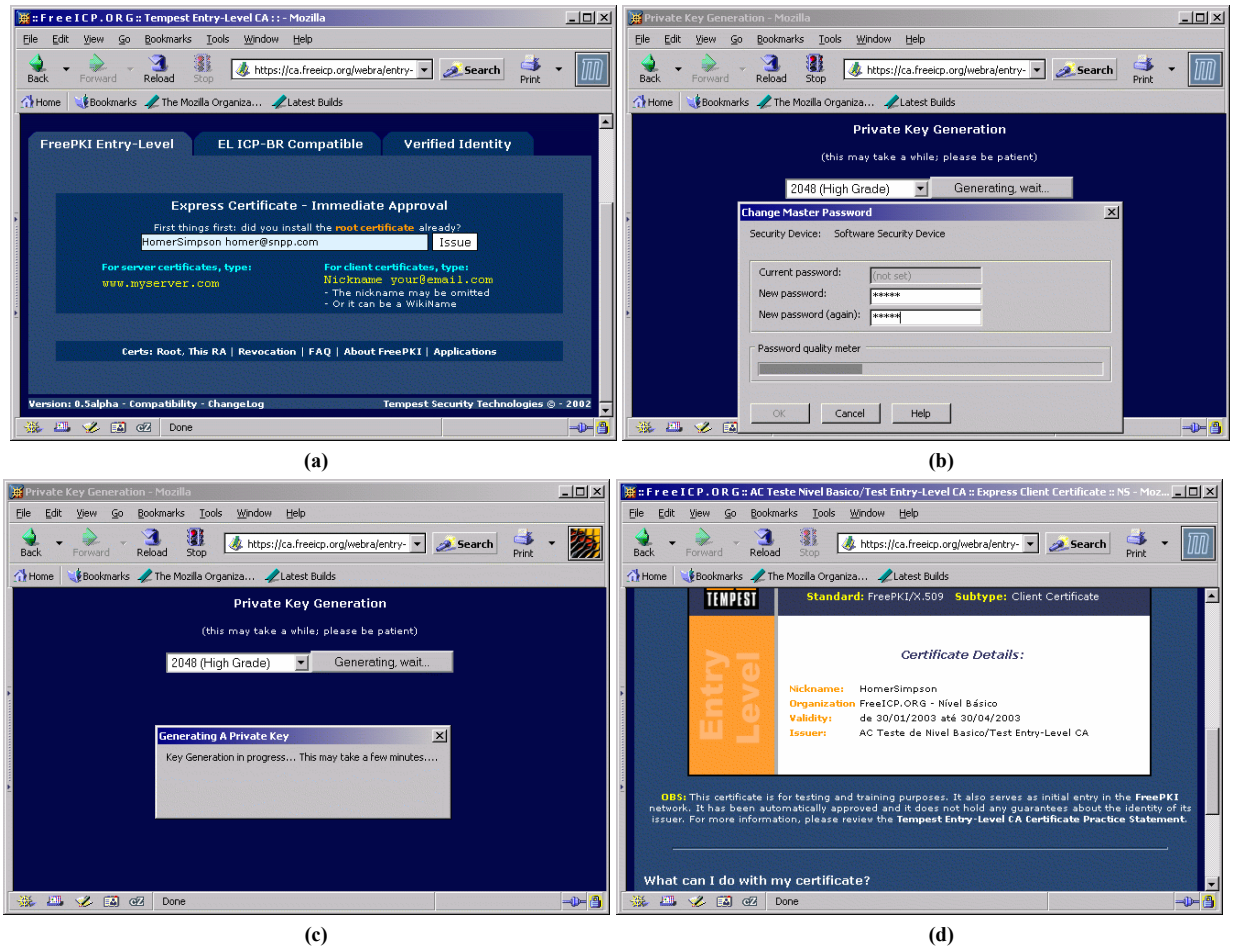

Figure 2: Mozilla & Netscape-derived browsers are more amenable to the express certificate concept: the whole process can be done in 4 steps. In **(a)**, the user types his name, email address and hits the Issue button. In **(b)**, the user is asked its container passphrase, or, in this case, to set one up. After that the user needs to perform no further action but to wait the key generation to finish **(c)** and the installation to complete **(d)**. In Opera, the dialogs look different but the process is essentially the same. All that supposes that the user has already installed the root certificate, which is conducted by a 1-6 step "wizard-like" sequence of dialog boxes, depending on the exact browser used. Mozilla, shown here, is the simplest.

Users also often don't get the point of the fingerprint verification and in many cases proceed without actually performing it rigorously. We can only hope to gain enough popularity in the future to be able to include our final root CA certificate in upcoming versions of common web browsers.

Figure 3: Express certificate generation in IE takes at least 11 steps, six of which shown here: in **(a)** the user types his username and email. In **(b)**, IE warns the novice with a somewhat needless obvious question. In **(c)** and **(d)**, the uncommonly well informed and disciplined user sets the security level to high; if he didn't explicitly ask for the high security setting (which the API doesn't allow the CA to set), the certificate would be stored without a passphrase. In **(e)**, the user finally gets to set his passphrase. In **(f)**, after confirming two levels of dialog boxes, there user receives another scary message before getting his certificate installed. All this supposes that the user has already installed the root certificate (a 9-step wizard) and has a recently patched (pos-Q323172) version of Internet Explorer (2 more steps which may fail silently if the computer is configured with policies restricting software installation); these were omitted for sake of brevity. All this ends up making certificate generation a frustrating process that fails in more than 60% of the attempts.

After having the certificate installed, a confirmation email is sent to the address the user specified. It clearly informs:

- The website name and the exact URL he accessed to perform the enrollment;
- An URL to revoke this certificate in a single click – for instance, in case the user feels the certificate was issued by someone other than himself;
- A succint description of the certificate's purpose – often, its sole purpose is to access some web application; in some EL CAs, we redirect the user to a directory of services that require these certificates;
- The fact that EL certificates are to be understood as "temporary, limited access", having a somewhat brief validity period (a few days or weeks); and the fact that the user can apply for a Verified Identity Certificate which grants greater validity and, possibly, more access privileges; an URL where the user can learn more about the certificate classes, CPSs, etc., is also given.

The loose identity tie, based mostly on the email address, makes the EL certificate somewhat like Verisign's Class 1 certificates [22]. Their process, however, require confirmation that the user controls that email address: they send an email with an URL that the user must access to pick up his/her certificate. Since it's very easy to create a valid email address in one of the many free webmail services, this doesn't add much security. Since commercial CAs usually take several hours to issue the certificate and send the email inviting the user to pick it up, this also sets the stage for a very common mistake: trying to pick it up in a different browser or computer from where the user requested it.

Every now and then some users lose the email with the revocation URL. In these cases, we tell them to go to the EL CA enrollment page just as if they wanted a new certificate. It detects that the supplied email addresses already have a valid certificate associated with them and offers the users three choices:

- **Revocation**: the EL Web CA application resends the email with the revocation URL; if and when the user wants to revoke the certificate, he accesses the URL.
- **Reissuing:** the EL Web CA sends a email with a special URL that revokes the previous certificate and issues a new one in a single step.
- **Do nothing:** leave things as they are.

There is considerable debate about whether revocation is a good idea or even needed at all in PKI systems [18, 16]. In light of this "revoke if it wasn't you who requested it" philosophy, along with the need to reissue certificates often due to the short certificate validity, revocation seems well suited, even though most relying-party applications neither correctly process CRLs nor support OCSP [26] or the like. We decided that all our "FreeICP.ORG-compliant" applications should include full support to a policy-based revocation verification system.

Another important point is the naming policy. It follows the following principles:

- **Globally Unique DNs**: The certificate holder's Distinguished Name in the Subject field should identify only his email address (with the Email OID), his name (in the Common Name OID) and the name of the Entry-Level CA who issued the certificate in a OU field. This makes DNs globally unique, preventing name clashes in case some user tries to issue certificates under more than one EL CA. Thus, the EL CA does not need to check elsewhere to see if this DN has already been taken. This also simplifies building associated directory services, like a global LDAP database.
- **Only one certificate per email address:** otherwise, the identity guarantee would be even slacker and the reissuing/revocation detection wouldn't work.
- **Server Certificates:** If a user supplies a valid DNS name as his name, the EL CA may issue a server certificate instead. It does need to do any kind of checks to see if the address exists. Several server certificates may be issued for the same contact email address (presumably, the servers' administrator). These certificates, however, are to be used for testing purposes only, since they bear no identity guarantees, and, as we shall see, the process for generating Verified Identity server certificates does not need them. The EL CA has the option, according to its own policies, of *not* issuing server certificates at all.
- **Identity privacy:** nickname and email offer little to correlate the user with his real world persona and sound very familiar to oldtime PGP users. This is in stark contrast with several other certification services, which require lots of personal data *in advance* to perform identity validation – an extreme example being the Brazilian National PKI, which not only demands the user's ID in the four most proeminent national registries [4], but includes them in the certificate, making them easy prey for spammers and identity thieves. In our system, personal data is required only when the user wants to get his identity validated, as shown in section 3.1 .

The deliberate bias towards user friendliness instead of "security" (as represented by identity guarantees) may be regarded as distateful by PKI purists. In fact, the scheme proposed above provides only slightly more

features than the PGP PKI (because of the much clearer revocation process) and the same level of identity validation – nearly none at all.

We argue, however, that all these usability trade-offs are fundamental to get user acceptance – both the end-user and the web application developers and administrators. To the best of our knowledge, there are no comprehensive studies on how usability problems affect the X.509 PKI – despite profuse folkloric horror user support stories within CA managers and PKI practitioners communities. However, [23] explains why PGP, widely thought as being "user friendly" because its Windows versions have a decent GUI, is much more non-intuitive and less usable than many of its enthusiasts would like to admit. Many of its results are very well applicable to the X.509 arena and have inspired our design for extreme simplicity.

Admittedly, even this simplicity cannot solve many *compliance defects* [5] inherent in PKI systems, like the impossibility to enforce good passphrases to protect the private key (since its generated by the client software; as shown in Figure 3, Internet Explorer, in particular, makes it upsettingly easy to have a private key with no passphrase at all; both Netscape and IE don't provide a way to require a minimum passphrase complexity) or to securely distribute the root CA's certificate (all of our EL CA's pages invite the user to reinstall the root CA certificate and check their fingerprints). However, yet again we are trusting the client software and user to do the "Right Thing". It is hoped that future versions of these clients may rectify these deficiencies.

Notwithstanding, the EL CA's Certificate Practice Statement must make it very clear what "Entry-Level certificate" means: no identity guarantees, good for testing, learning and initial entry in the trust system; and that the user's ultimate goal should be to upgrade the Entry-Level certificate to a Verified Identity one.

2.3 The Trust Manager Application and the Verified Identity CA

The PGP PKI adds in-band identity guarantess by allowing public keys to bear (possibly many) signatures from other users. In the X.509 PKI we can't to that because certficates can't have more than one signature; and end entities can't sign certificates – only CAs can. Thus, the natural solution is to make a CA that issues the user another certificate, which we call a "Verified Identity" certificate, when he passes some set of identity verification criteria.

Along with this Verified Identity CA there is the *trust manager* web application (TMWA, for short). It requires SSL client certificate authentication, accepting any user whose certificate was issued by both the EL and VI CAs. For each of them, the application would store their certificates, personal and contact data that the user voluntarily made available for purposes of identity checking and three *trust scores*:

- **Credibility score**: measures how certain we are that this individual is who s/he claims to be. It will be calculated as a weighted average of several *validators*, described below. It is analogous to PGP Key's "validity" rating, but much more granular – PGP's validity can be only "valid" or "invalid", while our credibility score is an integer number.
- **Introducer score:** indicates how trustable this user is when attesting or repudiating other users' identities. EL-certified users cannot have introducer points; only VI-class users may introduce other users. It is akin to PGP's "trust" rating, but, again, much more granular.
- **Suspicion score:** keeps track of how much this user is involved in identity contention with someone else. Users under suspicion (i.e., with non-zero suspicion scores) cannot have certificates issued or reissued under the VI CAs; besides, their introducer power is suspended. Notwithstanding, they can accumulate credibility points normally. If his credibility score exceeds his suspicion points, his privileges will be granted back and his suspicion points will be reset to zero. If a user spends too much time (say, a month) under suspicion, his account is deleted ("garbage collected", in our jargon) after being sent an email warning a few days before.

It is instructive to compare this scheme with other proposals like Thawte's Freemail Web-of-Trust program [21]: in their system, there is only one score that handles both the user's credibility and its experience/reliability as an introducer (which are called "notaries"). There is no suspicion management, since each notary is required to meet in person with any individual he introduces and it is seems to be thought that this makes the system immune to disputes.

Each VI CA would have an "eligibility criteria", based on the trust scores, metrics from the trust graph and, possibly, other criteria (e.g., requiring a specialized client, more secure than the mainstream web browsers). When some EL certificate user meets or exceeds these criteria, the VI CA would send an email inviting him to issue a VI certificate (this most likely requires generating a new private key, since most client software requires a one-to-one mapping between a certificate and a private key).

3 VALIDATORS

Validators are procedures executed by the TMWA for verifying the identity a certificate holder. We propose the following kinds of validators:

- **Automatic validators:** scripts/robots that verify some of the users personal data through automated

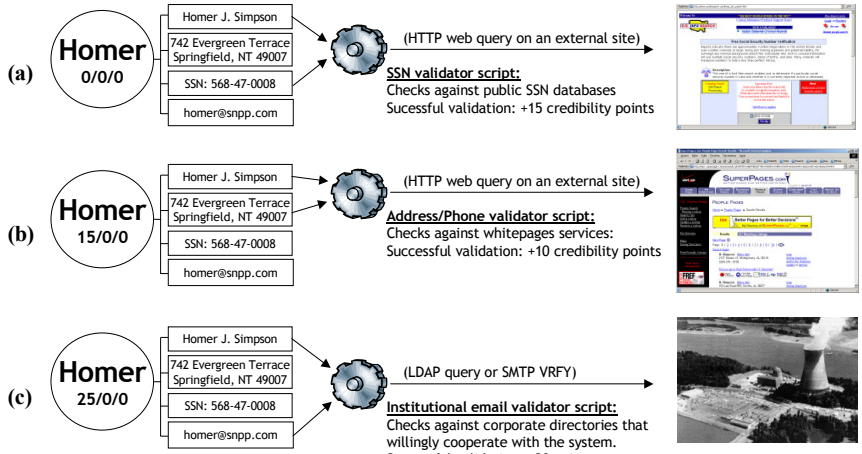

Figure 4: Automatic ("weak") validators in action: Homer Simpson enrolls in the TMWA and starts with no credibility, introduction or suspicion points. After having posted some verifiable personal information in the TMWA database, the system runs several scripts to confirm his claims: in (a), the TMWA queries an external web site (say, usinfosearch.com) to validate his name and SSN, earning him 15 points. In (b), the TMWA queries another website (in the example, superpages.com) to verify his address, earning 10 more points. In (c), it queries his empolyer's LDAP database and/or mail server. Homer gets out of the weak validator process with 45 points (not shown in the picture), with shouldn't be enough to grant him a VI certificate.

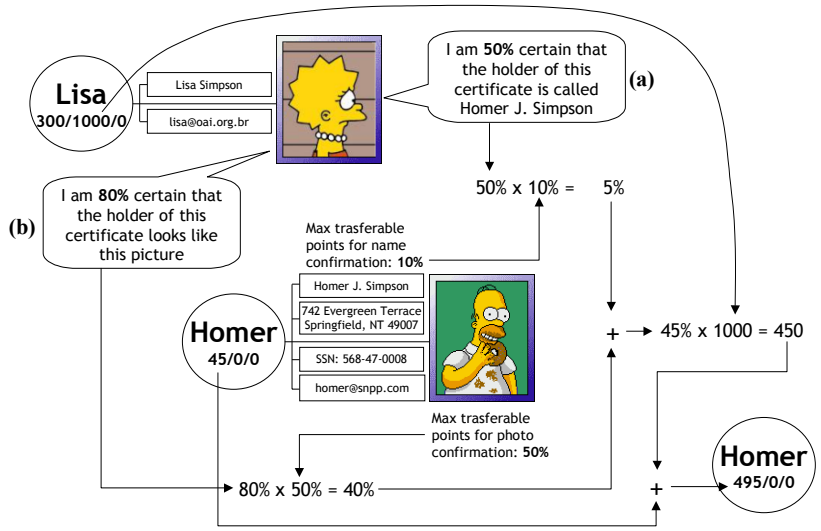

Figure 5: Strong validator process: A case of strong validation through direct introduction: Lisa is a highly trusted introducer, with 1000 introducer points. In (a), she transfers 50% of certainty that the certificate's owner name is Homer J. Simpson. The TMWA presets this validaton as yielding at most 10% of the introducer's trust score, so it transfers only 5% of Lisa's 1000 intro points to Homer. In (b), Lisa also attests with 80% certainty that this is Homer's picture, which, multiplied by the TMWA built-in limit of 50% for photo validations, grant him 40% of Lisa's 1000 points. He finishes this accredidation session with 45 points from the weak validators and 450 points from the strong validators. If this exceeds some VI CA minimum thresholds, it will grant him a VI certificate.

queries on public websites. For instance, checking names and addresses in whitepage services such as knowx.com or public government services (section 3.1 presents more specific examples). Users passing on these validators would receive a small amount of credibility points. It has to be small because, since it is based on public data, it's rather easily spoofed – because of that, the automatic validators are also called "weak" validators. However, they fulfill an important role: tying the certificate holder with a verifiable identity in the real world, as maintained by other independent sources. If anyone wants to spoof anyone else, they would spoof someone who probably exists and may eventually expose the spoof and/or dispute with the spoofer.

An important design principle is that they should not, insofar as possible, require on-site CA operators; they should be performed automatically, either by querying an online public Web database or being driven by remote users' input. They are to be triggered by the client users themselves, by accessing the proper web pages in the TMWA. One of their main functions is to allow users who already possess an entry-level certificate to increase their scores, up to the point for qualifying to get a VI certificate issued – without having to go in person or send paper credentials over snail mail to the CA.

- **User-driven Introduction:** the traditional way of cross-certification through trusted introducers – the user gets someone else *already with a high introducer trust rating* to vouch for his identity. The introducer would access his personal account in the Web CA and fill in a form saying that he has *x* percent certainty that the newcomer is who he says he is. This number would be multiplied by his introducer trust score and an attenuation factor

dependent on the exact identifier being validated. For instance, photographs are harder to fake than names or email addresses, so they should grant more points. The final result is added to the newcomer's credibility score.

All that means that the web-of-trust exists on the Web CA application's database as a set of trust scores; a graph of introducer-introductee relationships; and a log of validation procedures followed by each user. This last item is especially interesting for debugging and auditing, for it allows us to reconstruct the user's history and justify why the system has given him the score he has.

Any given user is capable of, at any time, check his scores and be informed of what steps to take in order to increase them. When the scores of a particular user grow beyond a specified threshold, he should be issued a certificate under the Verified Identity CA. That would mean that the user passed enough challenges and validations for the VI CA to be sure enough of his identity to issue him a certificate.

3.1 Automatic Validators ("Validator Robots")

Automatic validators provide new users a way to gain a small but significant initial credibility quickly, online, without having to ask other people to vouch for them.

It works like this: a new user would log on the TMWA using his EL certificate/private key pair and supply certain kinds of online-verifiable personal data, such as postal address, phone numbers, IDs in public services – Social Security Numbers, driver license numbers, etc.

The user would not be *required* to enroll his personal data in the TMWA; however, as he earns credibility points for each successful validation, he has an incentive to voluntarily do so. The Web CA/TMWA has a strict and clearly published privacy policy about keeping this data.

Also notice that the newcomer's personal data will be seen only by introducers (and possibly external auditors), which are expected to be much less than all Verified Identity users, and even less that the public at large. The TMWA may also offer to show the newcomer's personal data only to introducers he explicitly allows or invites, such as close friends, business associates, etc.

Groupings of the user's personal data could be validated by performing a HTTP web query on widely known and respected services. (This query would be performed by an automated script; no CA operator or human assistance should be necessary.) For instance:

- **Addresses and phone numbers:** these could be validated by checking them on whitepages directories such as knowx.com, whitepages.com and the like. It is considered valid only if the phone/address is registered with the user's name. Other people living in the same address would not pass this validation, but they have other alternatives.
- **Country-specific identifiers in public national databases:** Unique identifiers would be especially desired. For instance, several Brazilian governmental agencies' web sites provide web interfaces for querying their databases. Our prototype implementation has robots to check users' driver licenses, elector IDs, and others. The sites usually return the users full name and other status information when given the numeric IDs the user entered in the TMWA. If the name they give match (with some fuzziness factor to account for slight misspellings and truncations) with what the user provided, he is granted a few points.
- **PGP Key-based validation/introduction:** if the newcomer has a PGP key, he could post it to the TMWA. The PGP automatic validator then sends him an email encrypted with his PGP key containing an URL with a random validation code. If the TMWA receives the hit in this URL (which, remember, requires SSL client authentication), we take it as proof that the owner of the PGP key is the same person that owns the SSL client certificate. For that, we grant him a few credibility points.

  Since we are sure the user controls the PGP private key, we can take it a step further: if the user's PGP public key is signed by some trusted introducer, then it will be regarded as a direct user introduction, as described in section 3.2 – but performed in an entirely automatic manner. This is a special case where a "weak" validation may become a "strong" introduction.
- **Photographs and other human-verifiable data:** Certain personal data, such as headshot photographs, could also be accepted. Since they cannot be validated automatically, they would just "sit there" waiting for a human introducer to validate (as a means of saying "I attest that I checked that the individual who owns the private key corresponding to this certificate looks like this photo") or repudiate ("This is the picture of a slug and this certificate holder is fooling around with the system"). More about that in section 3.2 .

It is important to remind that all this personal data is kept in the TMWA database only. It is not included in the digital certificate when it is finally granted to the user. The VI certificate's DN is nearly the same of the EL certificate (except for the name of the VI CA in the OU field).

As each successful validation is achieved, the user's credibility points should be increased by the validator's trust weight multiplied by a measure of the

success of the validation. Figure 4 illustrates the process schematically, while figure 6b shows some snapshots of the process being conducted in our prototype implementation.

These kinds of validations are said to be "weak" because they are based on public data. They don't really prove the user is who he says he is. Thus, the amount of credibility points a user receive by these validations should be small compared with other validators, given that anyone can get personal data from some random individual in the very same services the TMWA uses to validate them and claim to be someone else.

The primary security function of the weak validators is to make it harder for a spoofer to get a certificate issued to an entirely fictious individual whose existence would be unlikely to be challenged. By having to assign a verifiable identity to the certificate, a spoofer incurs the risk of being challenged by the spoofed individual, as detailed in section 3.3 .

3.2 Strong Validators

The fastest way for a user to gain credibility points in the trust scoring system is by having other participants, especially highly trusted ones, to *voluntarily* verify his identity. This is particularly easy if the newcomer has a friend, supervisor, business associate or anyone within his acquaintance that holds a sizeable amount of introducer points.

The process is envisaged in the following ways:

- **TMWA introduction:** suppose Newton the newcomer asked (by email or though the TMWA community service) Ingus the introducer to vouch for him. Ingus logs on the TMWA, searches Newton in the database and fills a form specifying the amount of certainty he has that the individual he is introducing is who he says he is. This number, multiplied by his introducer score and an attenuation factor, is added to the Newton's credibility score. The attenuation is to prevent a single introducer from being able to escalate someone else's credibility too fast. Figure 5 sketches the situation schematically and figure 6c/d show the same situation happening in our prototype implementation.

  In order to encourage Ingus to perform the confidence level evaluation with the greatest care, the system informs him that if Newton is later determined to be a fraud, Ingus will have his introducer points reduced by the same percentual amount of confidence he deposited in Newton; and will receive as many suspicion points – which might put him directly in suspicion mode if it turns out to exceed his credibility. In other words, Ingus' evaluation is interpreted to be like an *insurance*: the amount of his own trust he would be willing to lose if Newton is found not to be who he says he is.

- **Cross-Certification**: A natural generalization of the PGP key-based "weak validator that may become an automatic strong introduction" is to accept certificates from other CAs or key hierarchies whose validation processes are known and that can be easly assigned a credibility rating. For instance, Verisign certificates could be accepted as another level of validation – Class 1 certificates, which validate only the email address, would add little extra credibility, while Class 2 and 3 certificates, which rely on institutional credentials and in-person enrollment, respectively, would grant much more points. Certificates from the CAs within our own hierarchy that employed traditional validation processes, as described in section 3 , could be likewise accepted.

It is worth reminding again that all these operations should be carefully logged, both for debugging and auditing purposes, so it becomes possible to reconstruct exactly why any particular user has got his scores.

3.3 Contentions

If a user Charlie the challenger supplies an unique identifier (say, his name, e-mail address, SSN, etc.) already claimed by someone else, he is to be put in *suspicious mode*: he earns as many suspicion points as the sum of the credibility scores of each user (himself included) having the same ID.

If Charlie's credibility reaches a certain fraction (say, half) of the credibility of some user he is contending with, the challenged user gets notified of this fact by email. This warning should give him time to take precautions against *takeover*: if Charlie's credibility exceeds the challenged user's, Charlie is awarded possession of the contended IDs. The challenged user is then put into suspicious mode: it's now his problem to prove his identity beyond Charlie's credibility.

These rules attempt to foil some avenue of identity theft attacks outlined below:

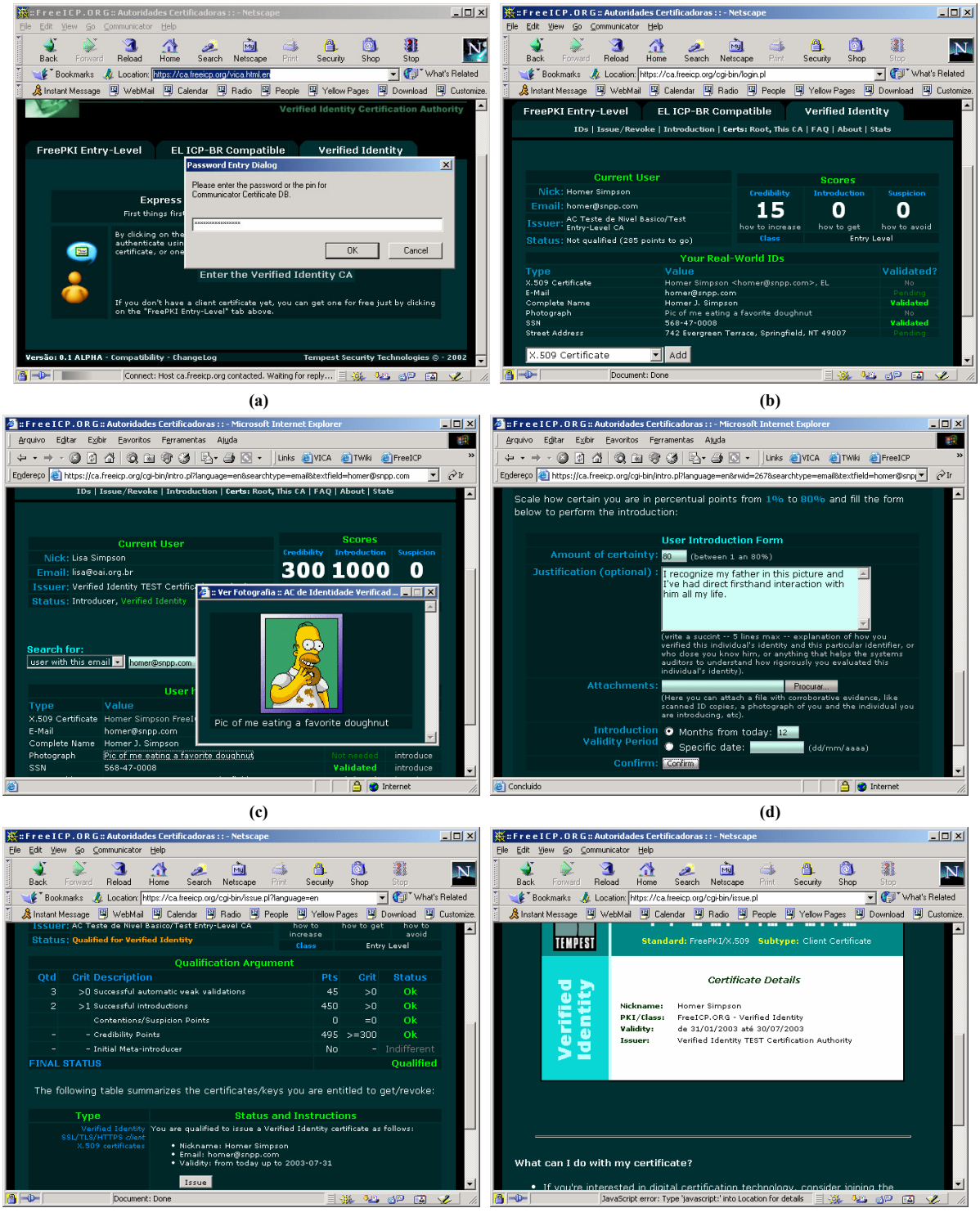

Figure 6: A simulation of Homer getting his VI cert: In **(a)**, he uses his EL client certificate to log on to the TMWA/VI CA; in **(b)**, we see him after he has already inserted some of his real-world IDs; as he was inserting them, some automatic validators have started their jobs: we see that his name/SSN tuple has already been validated, while the email and street address validators are still in the execution queue ("pending"). In **(c)**, Lisa logs on to the TMWA and searches for Homer. She checks his photo and in **(d)** vouches for it. In **(e)**, we go back to Homer and see him already qualified. After clicking the Issue button, he finally gets his VI certificate in **(f)**.

- **Post-takeover:** Suppose a legitimate user has already got his VI certificate issued without incident. Then, a persistant attacker issues several EL certificates with his name and uses them to log in the TMWA and generate contentions, supplying the legitimate user's public personal data to pass

through many weak validators and gain a modest amount of credibility points. As long as the legitimate user keeps his own credibility points high, the contenders won't be able to steal his identity. He is probably in the best position to do so, since he can convince introducers to attest his identity and strong validators yield so much more credibility points. The legitimate user gets early notification about contenders and their chances to take over his identity.

- **Pre-population:** The attacker enrolls in the TMWA before the legitimate user, supplying some of the spoofed user's public personal data to pass some of the weak validators. If the eligibility criteria for getting a VI certificate is set to near or more than the sum of what all possible weak validators could give, or requires a minimum number of introducers regardless of the credibility points, the attacker won't be able to assume the (unsuspecting) legitimate user's identity. Later, when the legitimate user enrolls, he will get into suspicion as soon as the starts contending with the spoofer; but since he is "the real one", he should be in the best position to convince the introducers to vouch for him and should win the credibility point fight easily.

All that relies, of course, in the trustworthiness of the introducers and the rigor with which they perform every single identity check. A rogue introducer can help attackers to bootstrap themselves through the credibility ranks or even create whole cliques of self-certifying fake communities (as long as the real users being spoofed don't enroll in the system and start contending with the fakes). The population base of our prototype implementation has not reached enough critical mass to allow these phenomena to be empirically observed, measured and statistically characterized, but it is natural to expect these issues will manifest themselves as the population base grows.

The fact that credibility points given from the introducer to the introductee act as insurance creates an incentive for caution: the introducer should know that if an identity validation error from his part is discovered (say, by other more graduated introducers or external audits), it will revert against himself, almost certainly quelling his privileges – a phenomenon we call "introducer demise".

In our prototype implementation, we didn't make introducer demises propagate through all of his introductees – this forces the whole trust scores to be recalculated, and, if not carefully calibrated, may make the entire trust web collapse. It was felt as undersirable in our small web, making it too fragile; but may be considered a minor local event in a large scale (say, millions of nodes) web, adding a self-correcting nature against introducer-aided fraud. Surely, this deserves deeper study.

At any rate, it is expected that contentions require much more human intervention than identity validations that go about without incidents. On the other hand, it should be possible to calibrate the system so that the former happens much more rarely than the latter.

Contentions also help to avoid "no way out" situations. For instance, it is not rare for users to lose their private keys. In these cases, we simply direct the user to get a new EL certificate and use it to reenroll in the VI CA. He will immediately start a contention with the "old copy" of himself – however, by simply reinserting his personal data and asking the same introducers he used last time to vouch for his identity, he should be able to surpass his old version's scores and get a new VI certificate. This will also cause his old VI certificate to be revoked, the old account to get in suspicion and eventually deleted by the garbage collector.

In short, contentions (and the whole score system) play an essential role in *managing* the users identities and real world IDs associations. It doesn't aim to be 100% fraud-proof; instead, it tries to be good enough for practical purposes and provide means of discovering and correcting errors, insofar as possible in an automatic manner; and appeal to the introducer community as a last resort.

3.4 VI certificate eligibility criteria

Our prototype implementation has only one VI CA with a very simple acceptability criterion: if the user exceeds 300 credibility points given from at least two introducers, he is granted a "VI level 1" certificate. This simplistic approach was chosen because it's easy to explain, simple for users to know what to do and makes the process of getting the VI1 certificate very quick: the user enters as much personal verifiable data as he wants, gathering a small amount of credibility due to the weak validators; then he consults the public list of introducers in the TMWA community page, asking the ones he knows to vouch for him.

Typically a few hours later, when the introducers check their emails (the TMWA informs them that someone asked to be introduced), the newcomer is validated and he is invited to the VI certificate generation page (it is worth noting that all this is made with SSL client authentication, thus requiring his EL certificate). His VI certificate is then issued in the same single-step manner adopted by the EL CAs and, finally, the user is informed of the applications that accept/require his newly issued certificate, along with instructions about how to register with them.

We plan to have "level 2", "level 3", VI certificates with stricter validation requirements, such as requiring several introducers, allowing the introducers to specify the validity of their trust grant and allowing the

newcomer to attain VI status only if at least one introducer vouches for him for at least one year (the suggested validity period of the VI certificates), etc. Another idea is to have a VI CA that requires the introducers to be members of stricter PKIs, such as ICP-BR (the Brazilian National PKI).

This makes a good moment to remind that each certificate from each CA has a life cycle of its own; they are not necessarily coupled or associated in any way. There's no need, for instance, to revoke an EL or a lower level VI certificate because the user has been issued a higher-level VI certificate. The only tying association is that they're kept in the TMWA.

It is interesting to compare this authentication metric with others, such as the ones studied by [17]. It is worth repeating the eight authentication principles they laid out and comment how our system adheres to or deviates from them.

- *Principle 1: The model, to which a metric is applied, should not require the user to infer bindings between keys and their owners. In particular, when representing certificates in a model: entities don't sign certificates, keys do.*

  In our system, the TMWA clearly identifies the several identities associated with a particular keypair/certificate, leaving no room for guesswork.

- *Principle 2: The meaning of the model's parameters should be unambiguous. This especially applies to the meaning of probabilities and trust values in the models that use them.*

  The numeric trust scores provide quantitative estimates of each trust quality (credibility, introducer, suspicion, etc). The scale and calibration may be somewhat arbitrary, but, within itself, it's self-consistent.

- *Principle 3: A metric should take into account as much information as possible that is relevant to the authorization decision that the user is trying to make.*

  The user (or application) doesn't make much more authorization decisions than choosing what EL or VI CAs to trust. But their acceptability criteria can be very well specified. We have tree different scores, which seem already a great deal of relevant authorization information – our system even has suspicion detection and management, a feature not found in many other metrics. We feel that more than that would overcomplicate the system.

- *Principle 4: A metric should consult the user for any authentication relevant decisions that cannot be accurately automated. A decision that could affect authentication should be hidden from the user only if it can be reached using unambiguous, well-documented, and intuitive rules.*

  That's precisely what strong validators are for. Since it was felt that automated validations could be rather easily spoofed, we made them the weak validators.

  On the other hand, our concept of "trust insurance" doesn't mean "monetary insurance" that would be paid in case of system failure (although it may be conceivable that it may provided as a add-on commercial service); instead, it means only a guarantee that introducers will be penalized for errors or misbehavior.

- *Principle 5: The output of a metric should be intuitive. It should be possible to write down a straightforward natural language sentence describing what the output means.*

  It is easy to explain what the metrics measured: "you got *n* points from one introducer, *m* points from another one, *i* points from posting your SSN, *j* points from posting your email, *k* points from posting your Brazilian CPF number, which add up more than the *t* threshold needed to get you a VI certificate."

  This opens up an interesting possibility: the page containing the certificate's CPS could add, within the bulk of the CPS text, an automatically generated, natural language explanation of these metrics and the guarantees (technical and legal) they provide – much like the "Unabridged Certificate" proposed in [7].

  Although the implementation has to take into account a lot possible state transitions, it is surprisingly easy to explain the dynamics of the scores due to its close mapping to how we intuitively transfer trust in the real world: we believe someone is who he says he is when he shows credentials and our acquaintances confirm; the credibility points just put a numeric scale to it. When we are introduced by someone highly regarded, we "gain" his credibility – he doesn't lose it unless we are proven to be a fraud. When we catch two or more people claiming to be someone else, we try to gather more and more evidence that supports one of them and disproves the others. Consistently bad introducers tend to develop bad reputations and become no longer trusted.

- *Principle 6: A metric should be designed to be resilient to manipulations of its model by misbehaving entities, and its sensitivity to various forms of misbehavior should be made explicit.*

  Section 3.3 detailed some of the contention management and their resistance to misbehavior.

More field experience is needed, however, to ascertain their efficiency in practice.

- *Principle 7: A metric should be able to be computed efficiently.*

  Since the TMWA enforces only direct introductions, there is no need to construct the entire introduction graph to compute the trust scores nor run graph-theoretic algorithms with superlinear time complexities (it may be useful to build the graph for other purposes, though). The calculations can be done incrementally and even reconstructed from the transaction log in linear time.

- *Principle 8: A metric's output on partial information should be meaningful.*

  Any user registered in the TMWA has trust scores, even if they have passed no validators. So, the metric is meaninful (although not useful) even in the absence of information.

3.5 The Root CA

The root CA has a very simple website offering the following services:

- **Automatic Entry-Level CA certificate signing:** the Entry-Level CA reference implementation sports a semi-automatic installer. One of its chores is to request the name and administrative email address of the new EL CA use them to generate its private key and CSR. It then sends it to a special URL within the Root CA's site that enqueues CSRs for processing by the signing engine. The queue has some built-in intelligence to discard duplicate attempts within a certain timeframe and avoid some flooding attempts. After being signed, the resulting certificate is sent to its administrative address specified in the beginning of the process.

  It has been suggested that the signing process should demand that the EL CAs administrator should be VI users; this guarantees a contact person and helps minimize rogue EL CAs. Although not implemented at the moment, this will probably be done in the near future.

- **Manual CA certificate signing:** an alternative manual procedure in case the automatic fails; now seldom used.

- **Revocation and non-compliance denounce:** the EL CAs have only a few obligations: they must not generate certificates that diverge from the naming policy nor issue certificates with validity periods greater than three months. But since the EL CAs operators have the source code, they may very well cheat. It's not possible to avoid it preventively, but the root CA can "retaliate": if anyone submits a nonconformant certificate to this service, the root CA will revoke the EL CA's certificate. (Generating an invalid certificate on prupose with a special "self-destruct" string is the correct, although exotic, procedure that the EL CA administrator should follow when he wants to revoke it). Admittedly, this is a rather weak contermeasure, given that most relying parties may not check the root CA's CRLs regularly or at all.

3.6 Server Certificates

Our initial focus was to identify individuals. However, one of the biggest demands – which spawned many commercial CAs – is to provide server identification. A free, collaborative way to securely identify servers might be desirable. Many of the concepts we developed seem to apply equally well to this field.

- The weak validator concept can be, in principle, extended for Internet hosts (say, for IPSec using IKE) or SSL servers: the robot would "ping" the service to see if it is up and running in the DNS or IP address specified by certificate's DN. In the case of SSL or IKE, it could also check if it is returning a proper set of certificates, etc. It should be possible to validate many kinds of services: HTTPS (HTTP over SSL), POP3 and SMTP over SSL, and possibly other less popular services, such as TELNET, FTP, VNC or Jabber over SSL.

- Internet hosts could be introduced in a similar way, except that their administrators would act in their behalf, inviting introducers to vouch for the identity of their SSL servers or IPSec-enabled hosts.

These generalizations, however, may be suceptible to DNS forgeries. Besides, there seems to be some confusion about what kind of guarantee the system could provide: many users misunderstand the term "secure site" and unrealistically expect them to mean "unhackable", or that the institution running the server is trustworthy; among many other interpretations quite different from the correct one. We are still working on a sensible set of validation procedures more easily understood by introducers and final users alike.

4 EXPERIENCES WITH THE FIRST RELYING-PARTY APPLICATIONS

The VI CA itself is the zeroth relying party application, since it is a full-fledged Web aplication requiring SSL client authentication. However, its tight integration with the other CAs makes it too much of a special case; to really grasp what our infrastructure could do, we selected another application to add FreeICP support to. TWiki [32], a web based collaborative content management system, was the natural choice, since we already used to run a few Wiki sites and making a PKI-enabled version with stronger authentication and improved security was a longtime wish of ours.

A short description of TWiki's functionalities follows: it looks just like an ordinary web site but allows editing the web pages (called "topics") directly in the web browser, adding attachments and keeping everything under revision control, so it's possible to reconstruct any past version, know who changed what and when, or undo undesired changes. Topics about

Figure 7: FreeICP integration in applications: The simplicity of Entry-Level CAs allow them to be included as a small visual element (the "Quick Registration" box) in a web application. It is integrated with the application in the sense that it not only generates certificates, but also performs the application setup necessary to create the user's account.

the same subject of interest are grouped in "webs". It has a simple but effective access control system: each web can have an access control list defining which users may be granted or denied permission to read or change the information. Individual topics can also have these ACLs for further granularity.

The original TWiki identified its users by the traditional username/password pair through the standard HTTP BasicAuthentication mechanism. User names are internally mapped to WikiNames satisfying its special naming conventions. To use mod_ssl's FakeBasicAuthentication mechanism is a natural and simple way to "upgrade" the system to use client certificates instead. This, however, proved rather unsatisfying, so we quitted using it and decided to implement client certificate support directly in the core application code:

- The application parses the certificate and maps the DNs to usernames. If it is not found in the user list (which is itself a topic), it redirects the user to an error page (except in the VI listing upgrade case described below).
- If the user's certificate is not a Verified Identity one, it is only granted access to public webs; that is, ones with no ACLs – even if that user is explicitly included in some web's ACL. This implements the "low privilege, guest-like access" principle that Entry-Level certificates should have. In this mode, the user can read the tutorials, practice with the test/sandbox areas, but has no access to sensitive information.
- If the user logs with a VI cert but is still enrolled with *a corresponding* EL certificate (i.e., one with exactly the same name and email address), the user is not redirected to the error page; instead, it is sent to a page offering to automatically upgrade his registration data. After confirming, he can no longer log on with his EL certificate; from this point on, only his VI certificate will be accepted and he will be granted access to the private webs (i.e., ones with explicit ACLs). This implements the "higher level, privileged" access principle that VI certificates are entitled to.
- An integrated Entry-Level CA was added as the new user registration box, as shown in Figure 7. That way, the user gets his certificate issued and his initial setup in the application (creating his personal topic from a template, adding him to the user's list) done in two simple steps (modulo the web browser's idiosyncrasies explained before).
- Besides getting access to the private webs, another motivation for upgrading to a VI certificate is the fact that when EL certificates expire, anyone can issue a new one with the same name/email and thus fake the previous user. To avoid that, we made the system accept EL certificates only up to two weeks from the initial inclusion in the users list, regardless of the age of the certificate.
- To cope with many revocations caused the users intial experimentation with client certificates, we implemented a full-blown policy-based revocation verification system. At first, we used mod_ssl's built-in CRL verification features, but it had a few limitations: first, we had to have external scripts to download the CRLs and put them in files that mod_ssl could read – that is, we couldn't have *on-demand* CRL downloads. Secondly, mod_ssl breaks the SSL connection when it determines the client certificate has been revoked. Although it seems the right thing to do, that denies the application an opportunity to display a page explaining to the user why his access was denied. Worse still, when this happens, Internet Explorer displays a bogus dialog box complaining that "the site cannot be trusted", instead of something more truthful like "this client certificate has been revoked".

  Because of all that, we disabled mod_ssl's revocation checking and patched TWiki to support it natively. It proved to be quite a challenge itself, but in the end it supported on-demand CRL downloading (i.e., it only downloads a CRL when it is needed to check the user's certificate), which contributes to alleviate the classical CRL problem of every relying party wanting to download the

freshest CRL at the same time; it displayed a nice message explaining to the user what happened; and incorporates some features to make it resistant to transient CRL download failures.

We made several other small changes to TWiki's core functionality. Although many of them were security related (for instance, the search feature didn't respect the ACLs; we fixed that) and sometimes quite interesting by themselves, most have little relation with digital identity support and have been omitted here for sake of brevity.

The final result was quite satisfactory: we managed to keep the registration process very quick and simple from the point of view of the novice users wanting immediate access to the tutorials and public webs. And, by compelling users to upgrade to VI certificates, we achieved considerable certainty about their identity and that they could only see information they were strictly authorized. Some informal testing we made in trying to subvert the system was promptly detected, but much greater scale testing is still needed to evaluate its merits relative to other authentication technologies.

It is natural to ask whether how well and quickly the users grasp all those trust scoring rules. In our experience, most users only invest the time to understand what they strictly need. Since most of our users only wanted to get access to TWiki and other apps, they got to learn only the rules related to increasing their credibility score (and many promptly forget them after getting the VI cert). Even so, we consider the fact that many users can get their VI certs in something between a few minutes to a few hours a striking success.

People only dive deeper when a suspicion event happens or we compel someone to become an introducer, requiring a more thorough understanding of the whole process. When their curiosity is then aroused, these users usually didn't feel intimidated by the complexity of the system; many end up making us explain all those rules in great detail. The single biggest reason for user rejection, in our experience, has come from IE users when the EL express certificate issuance process fails – which, unfortunately, happens in more than half of the cases.

5 CONCLUSIONS AND FUTURE WORK

We proposed two CA families to implement a PKI mixing the PGP and X.509 models based on the realization that the process of aggregating strong identity guarantees to a certain key/certificate should not be tied to its issuance; it should be done at a later moment, if and when convenient to the certificate holder. In fact, there are many instances when it's simply not worth the hassle to go through an extremely strict identity validation procedure when a not-so-trusted certificate would do just fine.

In our system, the Entry-Level family of CAs provide this focus on user and administrative simplicity. We've argued that it provides roughly the same kinds of protections that the PGP infrastructure: confidentiality through encryption but with little certainty of who the keys onwers are in respect to other identification systems. The proposed scheme allows the certificate to be granted immediately, becoming well suited for replacing website registration systems and similar end-user applications. The short lived certificates, when combined with application demand, creates an incentive for the user to "upgrade" his entry-level certificate to the longer lived, more widely trusted, Verified Identity ones.

Space constraints prevented us from being able to report the many interoperability pitfalls we ran into, the nontrivial solutions we were often forced to adopt and several other interesting implementation details. These may make material for a future paper; meanwhile, the reader is invited to visit our implementation site: www.freeicp.org.

The proposed Verified Identity family of CAs provide the higher identity assurance levels. It can be seen as a framework to unify several identification services and strictness criteria. It encompasses both the human-operator-based identity check systems now common on commercial or institutional CAs and a novel idea of a trust scoring web application that allows borrows the PGP's web-of-trust model but implemented over a centralized database to provide online-only, semi-automated identity validation – vaguely resembling the credit scoring systems now common in financial institutions. We argue that its collaborative nature may be exploited to make near-zero-cost certificates possible and thus allowing the "commoditization" of trustable digital certificates.

A trust management system was described that allows the users to tie their certificates to automatically verifiable real world identities and accumulate credibility by having these identities verified by veteran users that act as trusted introducers. The proposed model uses a much more precise system based on numeric scores that evaluate the user's identity credibility, trustworthiness as an introducer, and the amount of dispute that the user is having to gain control of other user's identities. In fact, contention detection and control is another area that this system proposes and both PGP and X.509 lack. Precisely because of its novely, it deserves deeper study.

We have shown that Verified Identities CAs can use these trust metrics to decide, according to their own acceptability criteria, whether a particular user or internet host is eligible to one of its certificates. A simple threshold criterion was proposed that subjectively adheres to all the authentication metric design principles posed by Reiter and Stubblebine. An

interesting point is that the metric allows for an easy description of itself in natural language that could be added directly to an automatically generated Certificate Practice Statement.

Other interesting avenue being pursued is the use graph-theoretic algorithms to monitor the growth of the certification network and provide feedback to help calibrate the system parameters to achieve specific security guarantee goals. Their use as authentication metrics may be also considered.

The field of automated identity verification has been blossoming with interesting new proposals. For instance, in [1] it is described a system in which an automated voice system dials to the telephone number the user supplied in the enrollment process and requests the user to confirm a challenge number and record his name and affiliation, for audit purposes. A whole different idea, much more sophisticated, would be to accept digitized fingerprints to be matched against law enforcement's databases. The inclusion of those kinds of automated identity verification systems within an implementation of the framework proposed in this paper may become a worthwhile research avenue.

Finally, we studied the customization of a web application to suport user identification using our CA infrastructure. Several proeminent lessons emerged: first, web browser's UIs could be adjusted to provide simpler certificate generation that could bring us closer of the "express certificate" concept brought by the EL CAs – in particular, Microsoft's Internet Explorer proved to be an endless source of user frustration. Second, applications must undergo significative changes to support the "temporary limited access" semantics of EL certificates and wider privileges of the "Verified Identity" class of users. Besides, revocation checking can no longer be ignored; applications must have full revocation verification support and its interactions and potential vulnerabilities must be carefully understood; this might become quite a challenge by itself. Notwithstanding, we have shown a situation in which the final result was quite acceptable. We plan to add FreeICP-like support to many other kinds of applications.

6 ACKNOWLEDGEMENTS

Thanks are due to Aldo Albuquerque, João Paulo Campello and Felipe Nóbrega for their helpful suggestions, insightful criticisms and invaluable assistance in coding the prototype implementations. We also thank all users at C.E.S.A.R. for agreeing to be our test population base by using our PKI-enabled TWiki and the other FreeICP-compliant applications. Rômulo Albuquerque, Renato Martini and Robson Gomes were especially patient with the quirks of the first versions. We are also indebted to the anonymous referees for their most valuable coments and criticisms on the first version of this paper.

"The Simpsons" characters are trademark and copyright of Fox and its related companies, even though they have arguably became part of popular culture. Used here just for entirely non-profit illustration purposes.

7 REFERENCES

1. Authentify, Inc., *Authentify|Register™ and RSA Keon® OneStep – Assuring User Identities in the Registration Process*, http://www.authentify.com/images/pdf/AR_RSA_Keon.pdf
2. Marc Branchaud, *A Survey of Public-Key Infrastructures*, MSc. Thesis, Department of Computer Science, McGill University, 1997.
3. J. Callas, L. Donnerhacke, H. Finney, R. Thayer, *RFC 2440: OpenPGP Message Format*, 1998, www.ietf.org/rfc/rfc2440.txt
4. Comitê Gestor da ICP-BR, *Resolução nº 11 de 14 de fevereiro de 2002*, www.icpbrasil.gov.br/RES_ICP11.htm
5. Don Davis, *Compliance Defects in Public-Key Cryptography*, Sixth Usenix Security Symposium Proceedings, July 1996, pp 171-178.
6. Simon Garfinkel, *PGP: Pretty Good Privacy. O'Reilly & Associates*, 1994, ISBN 1565920988
7. Ed Gerck, N. Bohm, *X.509 Certificates: A Readable Unabridged Inside View,* www.mcg.org.br/x509cert.htm
8. Ed Gerck, *Overview of Certification Systems: X.509, CA, PGP and SKIP*, MCG Group, 1998, www.mcg.org.br/cert.htm
9. David Goodenough, *A Heretic's view of Certificates*, www.dga.co.uk/customer/publicdo.nsf/public/WP-HERESY
10. Richard Guida, *Rebuttal to "Ten Risks of PKI"*, Computer Security Institute Alert, n 204, 2000, www.gocsi.com/pdfs/expert.pdf
11. Peter Gutmann, *X.509 Style Guide*, 2000, www.cs.auckland.ac.nz/~pgut001/pubs/x509guide.txt
12. Russel Housley, Warwick Ford, Tim Polk & David Solo, *RFC 3280*: *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, 2002
13. ITU-T, *Recommendation X.509/ISO/IEC 9594-8: Information Technology – Open Systems Interconnection – The Directory: Authentication Framework*, Internation Telecommunication Union, 1997.
14. Burton S. Kaliski Jr, *An Overview of the PKCS Standards*, RSA Laboratories, 1993

15. Neal McBurnett, *PGP Web of Trust Statistics*, 1997, bcn.boulder.co.us/~neal/pgpstat
16. Patrick McDaniel & Aviel Rubin, *A Response to "Can We Eliminate Certificate Revocation Lists?"*, Proc. Financial Cryptography 2000, February 2000.
17. Michael K. Reiter & Stuard G. Stubblebine, *Authentication Metric Analysis and Design*, ACM Transactions on Information and System Security, Vol. 2, No. 2, May 1999, pp 138-158.
18. Ronald Rivest, *Can We Eliminate Certificate Revocation Lists?*, Proceedings of Financial Cryptography 98, LNCS 1465, Springer-Verlag, pp. 178-183, Anguilla, BWI, February 1998
19. Bruce Schneier & Carl Ellison, *Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure*, Computer Security Journal, v 16, n 1, 2000, pp. 1-7, www.counterpane.com/pki-risks.pdf
20. William Stallings, *Cryptography & Network Security: Principles & Practice*, 2nd Edition, Prentice-Hall, 1998, ISBN 0138690170
21. THAWTE Inc., *Certifying your Credentials through the Thawte Web of Trust*, www.thawte.com/whitepapers/guides/pdfversion/wotguide.pdf
22. VERISIGN, Inc., *VeriSign PKI Disclosure Statement*, www.verisign.com/repository/disclosure.html
23. Alma Whitten, J. D. Tygar, *Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0*, Carnegie Mellon University, Proceedings of the 8th USENIX Security Symposium, August 1999, www.cs.cmu.edu/~alma/johnny.pdf
24. Jane K. Winn, *The Emperor's New Clothes: The Shocking Truth About Digital Signatures and Internet Commerce*, 2001, faculty.smu.edu/jwinn/shocking-truth.htm
25. Phillip R. Zimmermann, *The Official PGP User's Guide*, MIT Press, 1995.
26. M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, *RFC 2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP*, 1999, www.ietf.org/rfc/rfc2560.txt
27. PGP Corp Web Site: www.pgp.com
28. OpenSSL Project: www.openssl.org
29. Mod-SSL: www.modssl.org
30. Apache Web Server Project: www.apache.org
31. GNU Privacy Guard: www.gnupg.org
32. TWiki: A Web-Based Collaboration Platform: www.twiki.org.
33. Jonah Freeware PKIX implementation: www.foobar.com/jonah
34. Cryptlib security toolkit: www.cryptlib.orion.co.nz