SDSI and naming

CS380L: Mike Dahlin

November 24, 2004

> Our system is 'key-centric': SDSI principals *are* public digital signature verification keys.
> *Rivest and Lampson*

1 Preliminaries

1.1 Review

1.2 Outline

- Naming problem
- SDSI key ideas
- A few implementation details
- Self-verifying names

1.3 Preview

2 Naming: fundamental problem

- symmetric key: "authentication server"
  - e.g., Kerberos, most BAN examples
  - Problem: authentication server must be on line
  - Problem: Single authentication server inappropriate for large-scale system spanning administrative domains
- Fundamental problem: naming
  - Which names correspond to which keys?
- Public key infrastructure
  - Use public-key encryption: certificates binds *name* to *public key*
    - "certificate authority" with well-known public key rather than on-line authentication server with whom I share a secret key
  - Allow hierarchies of certificates
    - Note: though this is typically done with public key, there is no fundamental reason hierarchies of authentication servers could not be constructed
  - e.g.,

        * (Mike Dahlin K_{dahlin})$^{K_{UT}}$
        * (University of Texas K_{UT})$^{K_{VeriSign}}$
        * "verisign vouches that K_{UT} is the University of Texas's public key and the unversity of texas vouches that K_{dahlin} is Mike Dahlin's public key
    - Advantages:
        * Allows off-line CA
        * Allows complex hierarchies of trust
    - Key questions:
        * How do we know to trust verisign?
        * When should we trust UT to vouch for names (should you believe "foo.com vouches for K_X is Microsoft's public key"?)

- Existing solutions
    - DNS – no security (yet)
    - ship root CA with binaries (e.g., browser ships with root keys for VeriSign, USPS, etc.)
    - non-hierarchical
        * ssh – first time you contact a new machine, you get to decide whether to accept binding of name to key
        * pgp – similar (with some efforts to build "web of trust")

- Example of more general problem: which data correspond to *name*
    - If I can solve this problem for keys, I should be able to solve it for files, IP addresses, etc.
    - SFS, Freenet – files
    - Replace DNS's administrative root (mechanism) with "anyone is a root" (policy)
        * Verisign is a source of names just like ICAAN
        * Research project: replace DNS with "anyone can be root"

3 SDSI

Editorial: one of the lessons in operating system design is to make the interface actually reflect what is really going on. E.g.,

"An interface should capture the *minimum* essentials of an abstraction....the interface must not promise *more than the implementor knows how to deliver.*" (B. Lampson "Hints for computer system design") Examples in class: exokernel – export th hardware interface; active messages – high performance messaging by making interface match what hardware really does; scheduler activations; ...

Key ideas:

- **principals are public keys**
    - If you remember one thing, remember this
    - Much cleaner (imho) than traditional definition of naming
        * Traditionally: start with principals (users, machines, ...)
        * which keys belong to which principals?
        * bind key to a principal?

- SDSI: principals *are* keys (within the system nothing else makes sense)
  - Certificates are just opinions about names you might find convenient to use when referring to keys (principals)
- This seems subtle, but until I read this paper, I never realized how broken the standard way is...

- local name space
  - principal – global
  - name – local
    - "The principal you call alice-smith may be different from the principal I call alice-smith."
    - Think of names as "local names" or "nicknames"
  - Everyone is a CA
    - natural – of course anyone can sign certificates, the question is how to interpret them...
    - Natural – anyone can generate a nickname for a principal
    - Interface reflects reality
  - linked local name space
    - Everyone is a CA and can generate nicknames
    - Anyone is welcome to use anyone else's nicknames
    - syntactic sugar:
      - bob's alice's mother
      - verisign's visa's account-3402938402384
      - edu's cmu's cs's search-committee's chair
    - an explicit principal ID is a global name (albeit hard to remember)
      - *explicit principal ID* == public key (or in some systems cryptographic hash of public key)
      - "[crypt/RSA-512] =ldkfli23u0934ndfli3248jl28dfh2l4098dsf="'s bob
      - This case not emphasized in the SDSI paper, but it turns out to be a powerful idea...
    - a few easy to remember global names
      - verisign!!
      - USPS!!
      - DNS!!
      - Highly desirable that these be standardized everywhere; but no mechanism to enforce this (Assert-Hash provides sanity check)
    - so really all names I see are (a) relative to "me", (b) global, or (c) meaningless

      MY bob's alice's mother
      - verisign!!'s visa's account-234234802384
      - DNS!!'s edu's cmu's cs's search-committee's chair
      - "[crypt/RSA-512] =ldkfli23u0934ndfli3248jl28dfh2l4098dsf="'s bob
    - Different names can refer to same principal
      - We can have different nicknames for different people
      - Verisign!!'s microsoft's CEO
      - DNS!!'s com's microsoft's "Bill Gates"

- Group
  - convenient for describing access control
    - Define group in one step
    - Add group to ACL in another step
    - Simplify auditing – group can be given meaningful name
  - each group "owned" by a principal
    - DNS!!'s mit's biology-department's faculty
    - owning principal can issue signed statements "¡principal¿ is member of ¡group¿"
- Role
  - 2 reasons for rolls
    - convenient for access control – make intentions more clear (e.g., email from "mike as faculty at UT" v. "mike as independent consultant")
    - Support for "least privlege"(e.g., "mike as machine user" v. "mike as machine administrator" v. "mike as running software I downloaded from who knows where on the web")
    - Note: least privlege roles talked about a lot but used less...psychological acceptability?
  - an individual may create a public key for each role he/she plays
    - faculty-role
    - mike-travel-key
    - "bob can sign statements as the principal I call bob or as the principal I call bob's faculty role. I can distinguish these cases, and recognize when Bob is acting in his faculty role as opposed to acting as bob."
  - another strategy: create a group for each role (e.g., "¡principalX¿ is a member of hospital's head-nurse")
  - the two strategies differ in who controls who is acting in the role
  - also, "the second method has the slight disadvantage that the principal playing that roll signs with his full authority, rather than with the authority restricted to permissions of his role"
- Mechanism: certificate
  - 3 types of certificate
    1. identity
    2. name/value
    3. assert membership

Delegation certificate delegates the authority to sign statements of certain types on behalf of the delegator

4 Admin

The end is nigh

Midterm

- closed notes

- You may bring one 8.5x11 sheet of paper with anything written on it (both sides OK; restriction: 11pt or larger font or handwritten)

Project

- Written report due Friday May 3 5PM (roughly)
- Advice
  - Writing is hard and I could write a multi-page document about my advice for making it better (if not easier). If I could tell you one thing, I would tell you this (stolen from Armando Fox).
  - *Write from an outline. Let me say that again, because it's really important: write from an outline. I know no one who can reel off any coherent technical writing more than one page long without some kind of top-down strategy. At least sketch out the major sections of the paper, and what points you want to make in each, from 10000 feet. If you write any complete sentences during this phase, you're getting mired in detail already. Bullets are what you want.*
  - Armando goes on to provide more detail on how to start writing:
    1. start from the outline.
    2. Make the outline reflect the level of subsections: for each subsection, write no more than two lines describing the purpose/goal of that subsection. This text will NOT be part of the paper - it is only there to remind you what you are trying to accomplish. It is ESSENTIAL that you be able to capture the purpose of a subsection in one or two lines. If you cannot do this, then you probably don't understand what the subsection is really about, and when you try to write the text, it will be jumbled.
    3. Then, for each subsection, map out specific paragraphs: for each paragraph, write one sentence that explains the topic or main goal of just that paragraph. Again, this sentence probably will NOT make it into the actual text. It's important to keep it to one sentence. (As every style manual will tell you, including Strunk & White, virtually every well-formed paragraph does indeed have one sentence that explains the point of the paragraph, with the other sentences supporting or expanding on the point of the topic sentence.) If you cannot fit the point of the pargraph into 1 sentence, the paragraph is probably making ¿1 point, so should be split into multiple paragraphs.
    4. Read through everything you have written and see if it has a logical flow, ie if you believe it represents your work adequately.
    5. Give what you have written to a technical colleague completely unfamiliar with your work (but able to understand the computer science part), have them read it, then have them tell you (without looking at it) what s/he thinks the main point and contributions are.
    6. If all goes well, now replace the topic sentences with complete paragraphs.
  - This way of writing will not yield a shakespearean work of literature, but it is consistent and will result in readable, logically organized prose by construction.

5 SDSI implementation

5.1 Principal

```
(Principal:
   ( Public key: ...)
```

```
( Global-name: (ref: Verisign!! University-of-texas-at-austin Computer-science-dept dah
( Global-name: (ref: DNS!! edu utexas cs dahlin))
( Principal-at: ‘‘http://www.cs.utexas.edu/cgi-bin/sdsi-server’’)
( Server-at: ‘‘http://www.cs.utexas.edu/cgi-bin/sdsi-server’’) )
```

- "Information other than the public key are 'advisory'. Two principals are taken as being 'the same' iff they have the same public key"
- Note: nothing here is signed. Everything other than the key is an *optional hint*
- Global name – optional – *hint* – "how one might obtain relevant certificates if one wants to verify that the global names are indeed correct"
- Principal-at – optional – *hint* – queries to the principal may be addressed here
- Server-at – optional – *hint* – queries to the principal may be addressed here (preferred to "Principal-at"; servers are expected to be on line; principal-at may be off line most of the time)
  - Server also distributes a principal's auto-certificate, which contains information about that principal (similar to "finger")
  - Most information pushed into auto-certificates to keep Principal short
  - Server also distributes any bindings (e.g., groups, local names relative to this principal) this principal wants to export

5.2 Signed object

```
( Signed:
   ( Object-Hash: ( SHA1: =...= ) )
   ( Date: 2002-04-22T07:32:05.042-0700 )
   ( Signature: #... ) )
```

- Object-hash – hash of object being signed
- Signature – obtained by hashing the entire signed object (except for any subobjects of type Signature: or Object:) and applying signature algorithm to the result

Optional fields

- Signer – a principal – included when signed object might be used in a context where it is not obvious who the signer is
- Expiration-date, signing-key-hash, signing-for, reconfirm, ...

Example

- Object (Hash(Object) Date Expires K_{signer}) Signature

5.3 Local names and certificates

- I can bind a local name to a principal (key) or to another name
  - bind (ref: (Principal: (PublicKey: ...))) to bob
  - bind (ref: bob's friends) to myEnemies
  - bind (ref: bob's boss's secretary) to myInformant
- A principal may assign a value to a local name by issuing a corresponding certificate
  - identity certificate
    - assert a binding between an individual and a public key
    - bind a meaningful local name to a public key
    - "identity certificates have human-readable content"
    - "identity certificates must typically in the end be examined by people to see if the name and other attributes given are consistent with the attributes known to the human reader"
    - manual process for creating identity certificates
  - Example: my bob – bind a key to "bob"

```
( Cert:
    ( Local-Name: bob )
    ( Value:
        ( Principal:
           ( Public-Key: ... )
         )
    )
    ( Signed:
      ( Object-Hash: ( SHA1: =...= ) )
      ( Date: ... )
      ( Signer:
         ( Principal:
             ( Public-Key: ... )
             ( Global-Name: ( ref: Verisign!! Utexas CS dahlin ) ) // a hint
         )
      ( Signature: ... )
    )
```

  - Example: myEnemies: bind "bob's friends" to "myEnemies"

```
( Cert:
    ( Local-Name: myEnemies )
    ( Value: ( ref: bob friends ) )
    ( Signed: ... )
)
```

5.4 Auto-cert

- Signed by principal whom it is about
- Additional info not found in Principal: object
- No third party vouching for info – should not be trusted without suitable corroboration

- Example fields:
  - Photo: [image/gif] =...
  - Email-address: foo@example.org
  - Local name: Mike Dahlin *// the principal's favorite nickname for him/herself. One can consider using this nickname in one's own space for this principal, but it is not required*
  - Description: ... *// arbitrary free-form text by the entity of the controlling key about himself*
  - Fax: ...
  - Phone: ...
  - etc.

6 Self-certifying names

- All names I see are (a) local – relative to "me", (b) global, or (c) meaningless?
  - (MY) bob's alice's mother
  - verisign!!'s visa's account-234234802384
  - DNS!!'s edu's cmu's cs's search-committee's chair
  - "[crypt/RSA-512] =ldkfli23u0934ndfli3248jl28dfh2l4098dsf="'s bob
- Note: all of these names have the same structure:

```
(Cert:
  (Local-Name: N1)
  (Value: ( Principal: ( Public-Key: K1 ) ) )
  (Signed:
     ( Object-Hash: H1 )
     ( Signer: ( Principal: ( Public-Key: K2 ) ) )
 )


(Cert:
  (Local-Name: N2)
  (Value: ( Principal: ( Public-Key: K2 ) ) )
  (Signed:
     ( Object-Hash: H2 )
     ( Signer: ( Principal: ( Public-Key: K3 ) ) )
 )

(Cert:
  (Local-Name: N3)
  (Value: ( Principal: ( Public-Key: K3 ) ) )
  (Signed:
     ( Object-Hash: H3 )
     ( Signer: ( Principal: ( Public-Key: K4 ) ) )
 )
```

  - If I happen to already know that K4 corresponds to Verisign!! then when I receive the above set of certificates, I can interpret them as *Verisign!!'s N3's N2's N1* (e.g., Verisign's Microsoft's CEO's Lawyer)
  - If I happen to already have a nickname (say, "mom") for K4, I can interpret that as *mom's N3's N2's N1* (e.g., mom's sister's son's parole-officer)
  - If I happen to already have a nickname (say, "joe") for K2, I can skip some of these steps (e.g., I can have a different name for the same principal) *joe's parole-officer*
  - What if I don't yet have nicknames for any of these keys?
    * Then all I have is *???'s N3's N2's N1*
    * Who is ??? → K4
    * Treat this as K4's N3's N2's N1
    * Why do I have to know the name of my CAs if they seem to be useful (and sign useful stuff)?
      · Suppose somone wants to moderate a newsgroup finish this thought

- Point is: You don't need nicknames to talk about principals
  - Principal (key) is a *global name*
    * Someone can send you a raw key and you can use this
    * Of course, you will usually quickly assign a nickname
    * Many existing systems build on this idea (using SDSI principles if not implementation) to bootstrap trust w/o trying to get everyone to agree on a global CA
  - Even in SDSI, "global name" is really just local nicknames that are widely known
    * Who will decide what names are "global"?
    * Political quagmire – sure, Verisign gets in. But does microsoft put Netscape in (or vice versa). Does the #10 CA get in? #11? The CA for the US (USPS)? For Uruguay ?
    * If we have to agree on a *global* list, can SDSI ever be deployed
    * If we don't all agree, then what is a "global name"?
    * Are global names other than keys/principals needed in SDSI at all?
    * Couldn't we sick with local names and make the key the only global name?
    * Each software package could ship with some particularly convenient names (DNS, VeriSign), but there is no assumption that they are global
    * Can the "assert-match" primitive be used to bootstrap this to let nodes try to use the local name "verisign" as if it were global (but to have a way to fall back if not)?

6.1 SSH

- Each machine has a public/private key
- Log in to a machine for first time:

```
$ ssh poboy.cs.utexas.edu
The authenticity of host 'poboy.cs.utexas.edu (128.83.144.55)' can't be established.
RSA key fingerprint is 17:23:05:16:78:d2:bc:4d:6e:a6:99:d8:11:ff:5f:8c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'poboy.cs.utexas.edu,128.83.144.55' (RSA) to the list of kno
```

- What does this mean in terms of SDSI?

- I am really logging into machine 17:23:05:16... (the key *is* the principal)
- If I connect, the nickname "poboy.cs.utexas.edu" will be stored as a local name for "17:23:05:..."
- Suppose I wanted to call a friend on the phone and get them to securely log in to poboy, what I should tell them is "log on to 17:23:05:16:...; you can find this machine using the DNS name poboy.cs.utexas.edu"
- (This is a bit weird because we are accustomed to treating the DNS name is the truth...)

6.2 Freenet

- P2P storage system
- data = read(name)
- Name is crucial for security – who gets to choose name that corresponds to data?
  - Web: DNS gives uniqueness but (a) not secure and (b) ties fetch path to identity
- Solution: Self-certifying names: key/path; to add a file with name key/path, the data must be signed by key
- How to avoid remembering and typing in key whenever you want to read a page
  - Hyperlink from your home page < A HREF=freenet://23oi43l2j32lkjl2kjr/travel > Useful travel information </A>
    - "my linked-useful-travel-information"
    - How did you find this link? Someone sent it to you. Or you searched a search engine. Or ...
  - Hyperlink to someone else's page
    - "my freenet-yahoo's useful-travel-information"

6.3 SFS

- Kaashoek et al
- Path names like: sfs.lcs.mit.edu-17:23:05:16:.../homes/kaashoek/foo.txt
- the "sfs.lcs.mit.edu" is a *hint* on the address to use to connect
- All communication encrypted under key 17:23:05:16...
- Use symbolic links, etc. to hide key for convenience (my /localmount/sfs.lcs.mit.edu/)