
Identity

Elisa Bertino
CERIAS and CS & ECE Departments
Purdue University

Topics

(Chapter 13 of Textbook)

- What is identity
 - Identity for objects (referred to as *naming mechanisms*)
 - Identity for “users”
- Multiple names for one thing
- Different contexts, environments
- Pseudonymity and anonymity

Overview

- A simple definition of identity
- Object naming
- Users, principals, and subjects
- Certificates and names
- Hosts and domains
- State and cookies
- Anonymity

A Definition of Identity

- Identity is simply a computer's representation of an entity.
- Identity depends on the context where the object or user are referenced

Object Naming

- Identity depends on the system containing the object
- Different names for one object
 - Human use, *eg.* file name
 - Process use, *eg.* file descriptor or handle
 - Kernel use, *eg.* file allocation table entry, inode
 - In databases, content information (example, primary keys) is used to identify single records

Object Naming

- Different names for one context
 - Human: aliases, relative *vs.* absolute path names
 - Kernel: deleting a file identified by name can mean two things:
 - Delete the object that the name identifies
 - Delete the name given, and do not delete actual object until *all* names have been deleted
- Semantics of names may differ

Example: Names and Descriptors

- Interpretation of UNIX file name
 - Kernel maps name into an inode using iterative procedure
- Interpretation of UNIX file descriptor
 - Refers to a specific inode
 - Refers to same inode from creation to deallocation

Example: Different Systems

- Object name must encode location or pointer to location
 - *rsh, ssh* style: *host:object*
 - URLs: *protocol://host/object*
 - Example: <ftp://abccorp.com/pub/README> specifies that the object /pub/README from the host abccorp.com can be accessed by using the FTP protocol
- Need not name actual object
 - *rsh, ssh* style may name pointer (link) to actual object
 - URL may forward to another host

Users

- Exact representation depends from the specific system
- Example: UNIX systems
 - Login name: used to log in to system
 - Logging usually uses this name
 - User identification number (UID): unique integer assigned to user
 - Kernel uses UID to identify users
 - One UID per login name, but multiple login names may have a common UID

Multiple Identities

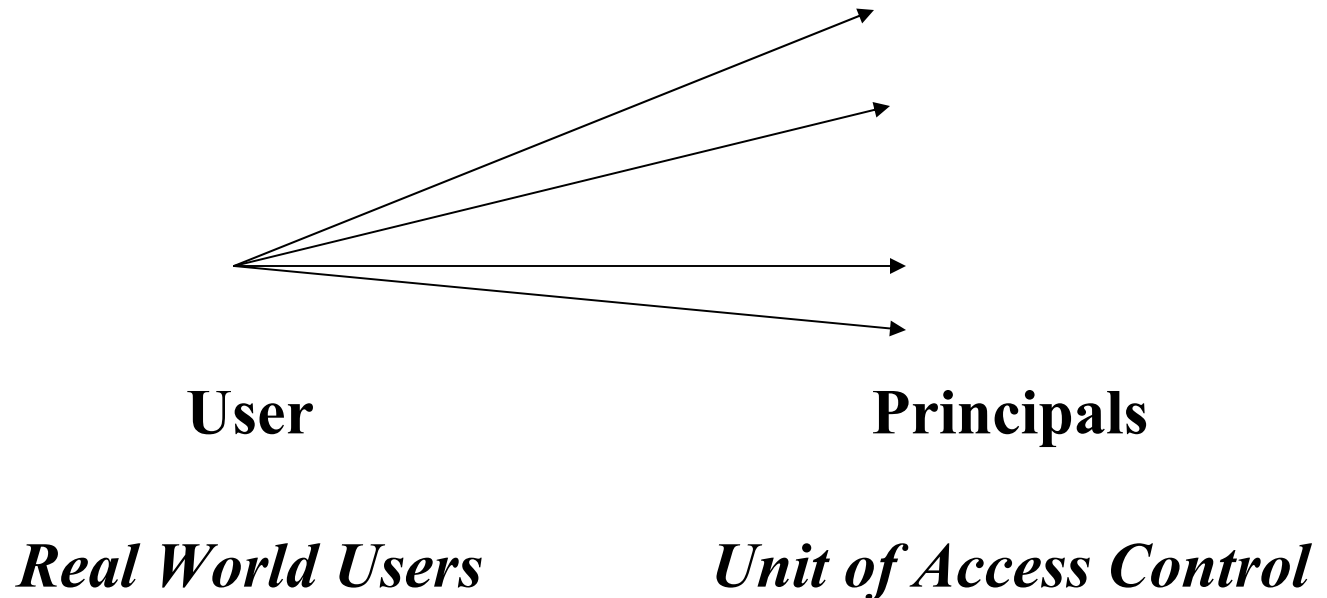
- UNIX systems again
 - Real UID: user identity at login, but changeable
 - Effective UID: user identity used for access control
 - Setuid changes effective UID to the UID of the owner of the program
 - Saved UID: UID before last change of UID
 - Used to implement least privilege
 - Work with privileges, drop them, reclaim them later
 - Audit/Login UID: user identity used to track original UID
 - Cannot be altered; used to tie actions to login identity

Users, Principals, and Subjects

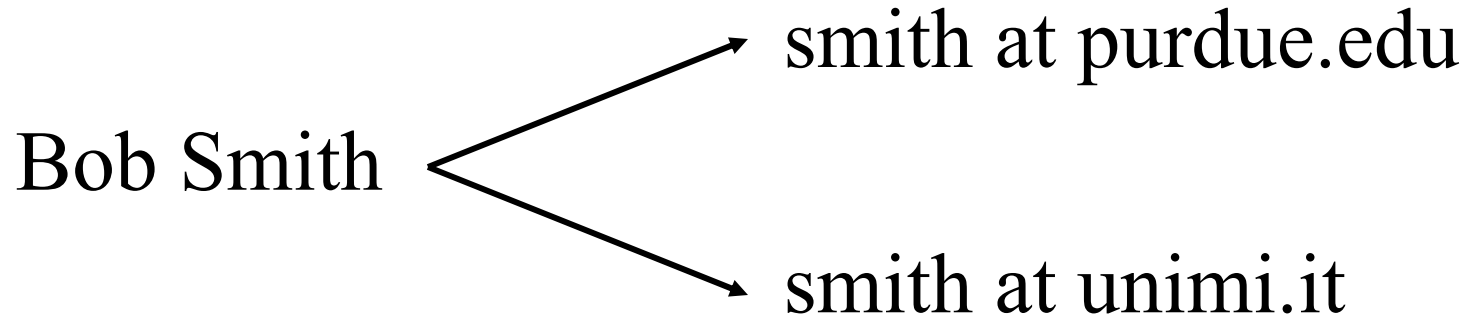
- A principal is an entity that can be granted access to objects or can make statements affecting access control decisions.
- Subjects operate on behalf of principals, and access is based on the principal's name bound to a user in some unforgeable manner at authentication time.
- Because access control structures identify principals, it is important that principal names be globally unique, human-readable and memorable, easily and reliably associated with known people.

Users and Principals

The system authenticates the user in the context of a particular principal



Users and Principals - Example



User

Principals

Users and Principals

- There should be a one-to-many mapping from users to principals
 - a user may have many principals, but
 - each principal is associated with a unique user
- This ensures accountability of a user's actions
- *In other words, shared accounts (principals) are bad for accountability*

Principals and Subjects

- ‘Principal’ and ‘subject’ are both used to denote the active entity in an access operation.
- The word ‘principal’ has many different meanings and is the source of much confusion:
 - Principals are public keys. [SDSI, 1996]
 - The term principal represents a name associated with a subject. Since subjects may have multiple names, a subject essentially consists of a collection of principals. [Li Gong, 1999]

Principals and Subjects

- A subject is a program (application) executing on behalf of a principal
- A principal may at any time be idle, or have one or more subjects executing on its behalf

Principals and Subjects

A recommendation

- *Policy*: A principal is an entity that can be granted access to objects or can make statements affecting access control decisions.
- *Example*: a user ID
- *System*: Subjects operate on behalf of principals; access is based on the principal's name bound to the subject
- *Example*: a process (running under a user ID)

Naming and Certificates

- Certificates were introduced to provide assurance about the entity associated with a public key
- Webster's Dictionary defines a certificate as “a document containing a certified statement, especially as to the truth of something”
- Certificates are issued by certification authorities (CA). A CA handles the certificates on behalf of his constituency and takes some sort of liability for having performed the necessary trust and security checks

Naming and Certificates

- Certificates are issued to a principal
 - Principals need to be uniquely identified to avoid ambiguities
- Problem: names may be ambiguous
 - Does the name “Matt Bishop” refer to:
 - The author of the textybook?
 - A programmer in Australia?
 - A stock car driver in Muncie, Indiana?
 - Someone else who was named “Matt Bishop”

Disambiguating Identity

- Include ancillary information in names
 - Enough to identify each principal uniquely
 - X.509v3 Distinguished Names provide an approach to the unique identification of each principal
- A distinguished name (DN) consists of a series of fields, each with a *key* and a *value*
- Example: X.509v3 DN
 - /O=University of California/OU=Davis campus/OU=Department of Computer Science/CN=Matt Bishop/
refers to the Matt Bishop (CN is *common name*) in the Department of Computer Science (OU is *organizational unit*) on the Davis Campus of the University of California (O is *organization*)
 - /O=Microsoft Corporation/OU=Quality assurance/CN=Matt Bishop/
refers to the Matt Bishop that works at Microsoft

CAs and Policies

- Each CA has two main policies controlling how it issues certificates:
 - CA's *authentication policy* describes the level of authentication required to identify the principal to whom the certificate is to be issued
 - CA's *issuance policy* describes the principals to whom the CA will issue certificates
- The difference between these two types of policies is as follows:

The first simply establishes the level of proof of identity needed for the CA to accept the principal's claim of identity whereas the second answers the question: "Given the identity of the principal, will the CA issue a certificate?"

Example: Verisign CAs

- Class 1 CA issued certificates to individuals
 - Authenticated principal by email address
 - Idea: certificate used for sending, receiving email with various security services at that address
- Class 2 CA issued certificates to individuals
 - Authenticated by verifying user-supplied real name and address through an online database
 - Idea: certificate used for online purchasing

Example: Verisign CAs

- Class 3 CA issued certificates to individuals
 - Authentication by background check from investigative service
 - Idea: higher level of assurance of identity than Class 1 and Class 2 CAs
- Fourth CA issued certificates to web servers
 - Same authentication policy as Class 3 CA
 - Idea: consumers using these sites had high degree of assurance the web site was not spoofed

Types of Certificates

- Identity certificate: it binds together a public-key and some information that uniquely identifies the certificate's principal – the certificates we have discussed so far.
- Attribute certificate: it binds an identity to an authorization, title or role by a digital signature. That signature is produced by a trusted third party, referred to as *Attribute Authority*. This type of certificate is being increasingly used.
- Authorization certificate: it binds an authorization, role or title directly to a public key rather than to an identity. This type of certificate has been proposed to shorten the authorization process. It is not frequently used.

Identity on the Web

- Host identity
- State and Cookies
- Anonymity
 - Anonymous email
 - Anonymity: good or bad?

Host Identity

- Bound up to networking
 - If the host is not connected: pick any name
 - If the host is connected Connected: one or more names depending on interfaces, network structure, context
- Each host, conceptually, has a principal at each layer that communicates with a peer on other hosts
- Databases contain mappings between different names.

Example

- Layered network
 - Media Access Control (MAC) layer
 - Ethernet address: 00:05:02:6B:A8:21
 - AppleTalk address: network 51, node 235
 - Network layer
 - IP address: 192.168.35.89
 - Transport layer
 - Host name: cherry.orchard.chekhov.ru

Host spoofing

- Attacker spoofs identity of another host
 - Protocols above the identity being spoofed will fail
 - They rely on spoofed, and hence faulty, information
- Example: if an attacker can alter the entries in databases containing the mapping of a lower-level identity to a higher-level identity, the attacker can spoof one host by routing the traffic to another

Domain Name Servers

- The best known mapping databases is the Domain Name Service (DNS) which associates host names and IP addresses
- In absence of cryptographic authentication of hosts, the consistency of DNS is used to provide a weak authentication

Domain Name Servers

- A DNS maps transport identifiers (host names) to network identifiers (host addresses)
 - Forward records: host names → IP addresses
 - Reverse records: IP addresses → host names
- Weak authentication
 - Not cryptographically based
 - Various techniques used, such as reverse domain name lookup

Reverse Domain Name Lookup

- Validate identity of host name
 - Get IP address of host
 - Get associated host name via DNS
 - Get IP addresses associated with host name from DNS
 - If first IP address in this set, accept name as correct; otherwise, reject as spoofed
- If DNS corrupted, such an approach would not work

DNS Security Issues

- Trust is that name/IP address binding is correct
- Goal of attacker: associate incorrectly an IP address with a host name
 - Assume attacker controls name server, or can intercept queries and send responses

Attacks to DNS

- Change records on server
- Add extra record to response, giving incorrect name/IP address association
 - Called “cache poisoning”
- Attacker sends victim request that must be resolved by asking attacker
 - Attacker responds with answer plus two records for address spoofing (1 forward, 1 reverse)
 - Called “ask me”

Cookies

- A cookie is a token containing information about state of transaction on network
 - Usual use: refers to state of interaction between web browsers and clients
 - Idea is to minimize storage requirements of servers, and put information on clients
- Client sends cookies to server

Some Fields in Cookies

- *name, value*: are encoded into the cookie and present the status; the interpretation is that *name* has an associated *value*
- *expires*: how long cookie valid
 - Expired cookies discarded, not sent to server
 - If omitted, cookie deleted at end of session
- *domain*: domain for which cookie intended
 - Consists of last n fields of domain name of server
 - *Must* have at least one “.” in it
- *path*: it further restricts the dissemination of the cookie. When a Web server requests a cookie, it provides a domain (its own). Cookies that match that domain may be sent to the server.
- *secure*: send only over secured (SSL, HTTPS) connection

Example

- Caroline puts 2 books in shopping cart at books.com
 - Cookie: *name* bought, *value* BK=234&BK=8753, *domain* .books.com
- Caroline looks at other books, but decides to buy only those
 - She goes to the purchase page to order them
- Server requests cookie, gets above
 - From cookie, the server determines books in shopping cart

Sending and Requesting Cookies?

- A Web server can *only* request cookies for its domain
- A Web server can however send to the browser cookies marked for the domain of another Web server
- When the client accesses the second Web server, this server can request the cookies marked for its domain but sent by the first server

Caroline Example

- Server books.com sends Caroline 2 cookies
 - First described earlier
 - Second has *name* “id”, *value* “books.com”, *domain* “adv.com”
- Advertisements at books.com include some from site adv.com
 - When drawing a page, Caroline’s browser requests content for ads from server “adv.com”
 - Server requests cookies from Caroline’s browser
 - By looking at *value*, server can tell Caroline visited “books.com”

Confidentiality of Cookies

- Cookies can contain authentication information, both user-related and host-related
- Depending on the sensitivity of the interactions with the server, protecting the confidentiality of these cookies may be critical

Anonymity on the Web

- Recipients can determine origin of incoming packets
 - Sometimes not desirable
- Anonymizer: a site that hides origins of connections
 - Usually a proxy server
 - User connects to anonymizer, tells its destination
 - Anonymizer makes connection, sends traffic in both directions
 - Destination host sees only anonymizer

Example: *anon.penet.fi*

- Offered anonymous email service
 - Sender sends letter to it, naming another destination
 - Anonymizer strips headers, forwards message
 - Assigns an ID (say, 1234) to sender, records real sender and ID in database
 - Letter delivered as if from anon1234@anon.penet.fi
 - Recipient replies to that address
 - Anonymizer strips headers, forwards message as indicated by database entry

Problem

- Anonymizer knows who sender and recipient *really* are
- Called *pseudo-anonymous remailer* or *pseudonymous remailer*
 - Keeps mappings of anonymous identities and associated identities
- If you can get the mappings, you can figure out who sent what

More *anon.penet.fi*

- Material claimed to be copyrighted was sent through the remailer
- Finnish court directed owner to reveal mapping so plaintiffs could determine sender
- Although the owner appealed, he subsequently shut down the site
- More sophisticated approaches have been developed, such as Cypherpunk remailer, and Mixmaster remailer

Anonymity

- Anonymity provides a mechanism to protect people from having to associate their identities with some data or actions
- Is this desirable?
- Some purposes for anonymity
 - Removes personalities from debate
 - With appropriate choice of pseudonym, shapes course of debate by implication
 - Prevents retaliation
- Are these benefits or drawbacks?
 - Depends on society, and who is involved

Anonymity and Privacy

- Anonymity protects privacy by obstructing amalgamation of individual records
- It is important, because amalgamation poses 3 risks:
 - Incorrect conclusions from misinterpreted data
 - Harm from erroneous information
 - Not being let alone
- However, anonymity hinders monitoring to deter or prevent crime
- Conclusion: anonymity can be used for good or ill
 - Right to remain anonymous entails responsibility to use that right wisely