

# L13: Representing Identity



Hui Chen, Ph.D.  
Dept. of Engineering & Computer Science  
Virginia State University  
Petersburg, VA 23806

# Acknowledgement

---

- ❑ Many slides are from or are revised from the slides of the author of the textbook
  - Matt Bishop, Introduction to Computer Security, Addison-Wesley Professional, October, 2004, ISBN-13: 978-0-321-24774-5. [Introduction to Computer Security @ VSU's Safari Book Online subscription](#)
  - <http://nob.cs.ucdavis.edu/book/book-intro/slides/>

# Outline

---

- ❑ Concept of identity
- ❑ Principal in computer systems
- ❑ Identity defined by functions
  - Files and objects
  - Users, groups, and roles
- ❑ Identity for certificates: Certificates and names

# Identity

---

- ❑ *Identity* is simply a computer's representation of an *entity*
- ❑ Principal
  - A *principal* is a unique *entity*.
  - An *identity* specifies a *principal*
- ❑ Authentication
  - Authentication binds a principal to a presentation of identity internal to the system
  - All decisions of access and resource allocation assume that the binding is correct

# Purposes of Identities

---

## □ Accountability

- Requires an identity that tracks principals so that the principal taking actions can be unambiguously identified
- Logging and auditing
  - In most systems, logged identity maps to a user account, to a group or to a role

## □ Access control

- Requires an identity to determine a specific access (or type of access) should be allowed
  - In most systems, access rights are on the identity of the principal executing process

# Principal

---

- ❑ Principal: a unique entity
- ❑ Identity: specifies a principal
- ❑ Authentication: binding of a principal to a representation of identity internal to the system
  - All access, resource allocation decisions assume binding is correct

# Purposes of Principals

---

## □ Accountability

- Tracks principals across actions and changes of other principals
- Any actions taken by the principals can then be unambiguously identified
- Tied to logging and auditing

## □ Access control

- Allow or disallow a specific access or a type of access
- Most systems adopt the *process* model
  - A process executed by a user has a subset of the user's rights where the user is presented by an identity

# Identity in Computer Systems

---

- ❑ One would state,
  - User *Alice* can read *Bob's* files
  - Subject: Alice
  - Object: Bob's files
- ❑ Both subjects and objects require identities
- ❑ Be aware of the complexity
  - Multiple names for one thing in different contexts and environments



# Files and Objects

---

- ❑ Identity depends on system containing object
- ❑ Different names for one object
  - Human use
    - ❑ e.g., file name
  - Process use
    - ❑ e.g., file descriptor or handle
  - Kernel use
    - ❑ e.g., file allocation table entry, *inode*

# Multiple Names for an Object

---

- ❑ 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: one file may have multiple names
    - ❑ On some systems, “deleting a file” is to mean removing the given file name; while on the others, it is to mean to remove the name and the file object.

# Example: Files in Linux/Unix

---

□ 4 different types of file names

1. *inode*
2. File descriptor
3. File names: absolute path names
4. File names: relative path names

# inode

---

- ❑ uniquely identifies a file
- ❑ contains file attribute information, e.g., access control permission and owner information
- ❑ identifies the specific disk blocks that contains the file's data

# File Descriptor

---

- ❑ Abstracts *inode* into a presentation that a process can read from, write to, and so forth
  - i.e., Processes read and write files using a file descriptor
- ❑ Interpretation of Linux/UNIX file descriptor
  - Refers to a specific inode
  - Refers to same inode from creation to deallocation
  - File descriptor cannot rebound to a different file

# File Names

---

- ❑ Identity files by describing their positions in the file hierarchy
- ❑ Absolute path names
  - Describe the locations of files with respect to the root of the Linux/UNIX file hierarchy
- ❑ Relative path names
  - Describe the locations of files with respect to the directory in which the current process is executing
- ❑ Processes and users can use file names to identify files

# File Names and inode

---

- ❑ Interpretation of Linux/UNIX file name
  - Kernel maps name into an *inode* using iterative procedure
  - Same name can refer to different objects at different times without being deallocated
    - ❑ Causes race conditions

# Example: Different Systems

---

- ❑ Object name must encode location or pointer to location
  - rsh, ssh style: host:object
  - URLs: protocol://host/object
    - ❑ <http://www.vsu.edu/academics/registrar/final-exam-schedule.php>
    - ❑ where
      - protocol: http
      - host: www.vsu.edu
      - object: /academics/registrar/final-exam-schedule.php
  - Need not to name actual object
    - ❑ rsh, ssh style may name pointer (link) to actual object
    - ❑ URL may forward to another host



# Users

---

- ❑ One would state,
  - User *Alice* can read *Bob's* files
  - Subject: Alice
  - Object: Bob's files
- ❑ *Identity* tied to a single *entity*
- ❑ Exact representation tied to system
- ❑ Often as identities of principals executing processes

# Example: Linux/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
    - ❑ e.g., the superuser is any user whose UID is 0 regardless of that user's login name
  - One UID per login name, but multiple login names may have a common UID

# Multiple Identities for Users in Linux/Unix Systems

---

- ❑ Real UID: user identity at login, but *changeable*
  - see `setreuid(2)`
- ❑ Effective UID: user identity used for access control
  - `setuid` programs changes effective UID, see `setuid(2)`
- ❑ 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

# Further Reading

---

- ❑ Setuid Program Example
  - [http://www.gnu.org/software/libc/manual/html\\_node/Setuid-Program-Example.html](http://www.gnu.org/software/libc/manual/html_node/Setuid-Program-Example.html)
- ❑ Mark S. Dittmer and Mahesh V. Tripunitara. 2014. [The UNIX Process Identity Crisis: A Standards-Driven Approach to Setuid](#). In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS '14)*. ACM, New York, NY, USA, 1391-1402. DOI=<http://dx.doi.org/10.1145/2660267.2660333>
- ❑ D. Tsafrir, D. D. Silva, and D. Wagner. [The murky issue of changing process identity: revising "setuid demystified"](#). *USENIX;login*, 33(3):55--66, June 2008.
- ❑ Hao Chen, David Wagner, and Drew Dean. 2002. [Setuid Demystified](#). In *Proceedings of the 11th USENIX Security Symposium*, Dan Boneh (Ed.). USENIX Association, Berkeley, CA, USA, 171-190.

# Groups and Roles

---

- The “*entity*” may be a *set of entities* referred to by a single *identifier*
  - Members of the set must be distinguishable
  - The set may have an identity separate from any its members

# Groups

---

- ❑ Used to share access privileges
  - Principals often need to share access to file
  - e.g., all students have access to “StudentActivityPlan.txt”
  - Most systems allow principals to be grouped into sets called *groups*
- ❑ First model: alias for set of principals
  - Processes assigned to groups
  - Processes stay in those groups for their lifetime
- ❑ Second model: principals can change groups
  - Rights due to old group discarded; rights due to new group added

# Roles

---

- ❑ Group with membership tied to function
  - Rights given are consistent with rights needed to perform function
- ❑ Uses second model of groups
- ❑ Example: DG/UX
  - User *root* does not have administration functionality
  - System administrator privileges are in *sysadmin* role
  - Network administration privileges are in *netadmin* role
  - Users can assume either role as needed

# Naming and Certificates

---

- ❑ Certificates as a mechanism for binding cryptographic keys to identifiers
  - Certificates issued to a principal
    - ❑ Principal uniquely identified to avoid confusion
    - ❑ An identifier corresponds to a principal
- ❑ Problem: names may be ambiguous
  - Does the name “Matt Bishop” refer to:
    - ❑ The author of the textbook?
    - ❑ 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 principal uniquely
  - X.509v3 Distinguished Names do this
- ❑ Example: X.509v3 Distinguished Names
  - /O=University A/OU=Department of Computer Science/CN=David Smith/  
refers to the David Smith (CN is *common name*) in the Department of Computer Science (OU is *organizational unit*) at University A (O is *organization*)

# Certificate Authorities and Policies

---

- “David Smith” wants a certificate from Certs-from-Us
  - How does Certs-from-Us know this is “David Smith”?
    - CA’s *authentication policy* says what type and strength of authentication is needed to identify David Smith to satisfy the CA that this is, in fact, David Smith
  - Will Certs-from-Us issue this “David Smith” a certificate once he is suitably authenticated?
    - CA’s *issuance policy* says to which principals the CA will issue certificates

# Example: Vendor Defined Certificate Classes

---

- See <http://www.symantec.com/page.jsp?id=roots>

# Internet Certification Hierarchy

---

## □ Tree structured arrangement of CAs

- Root is [Internet Policy Registration Authority](#), or [IPRA](#)
  - Sets policies all subordinate CAs must follow
  - Certifies subordinate CAs (called *policy certification authorities*, or PCAs), each of which has own authentication, issuance policies
  - Does not issue certificates to individuals or organizations other than subordinate CAs
- PCAs issue certificates to ordinary CAs
  - Does not issue certificates to individuals or organizations other than subordinate CAs
- CAs issue certificates to organizations or individuals

# Example

---

- ❑ University of Valmont issues certificates to students, staff
  - Students must present valid registration cards (considered low assurance)
  - Staff must present proof of employment and fingerprints, which are compared to those taken when staff member hired (considered high assurance)

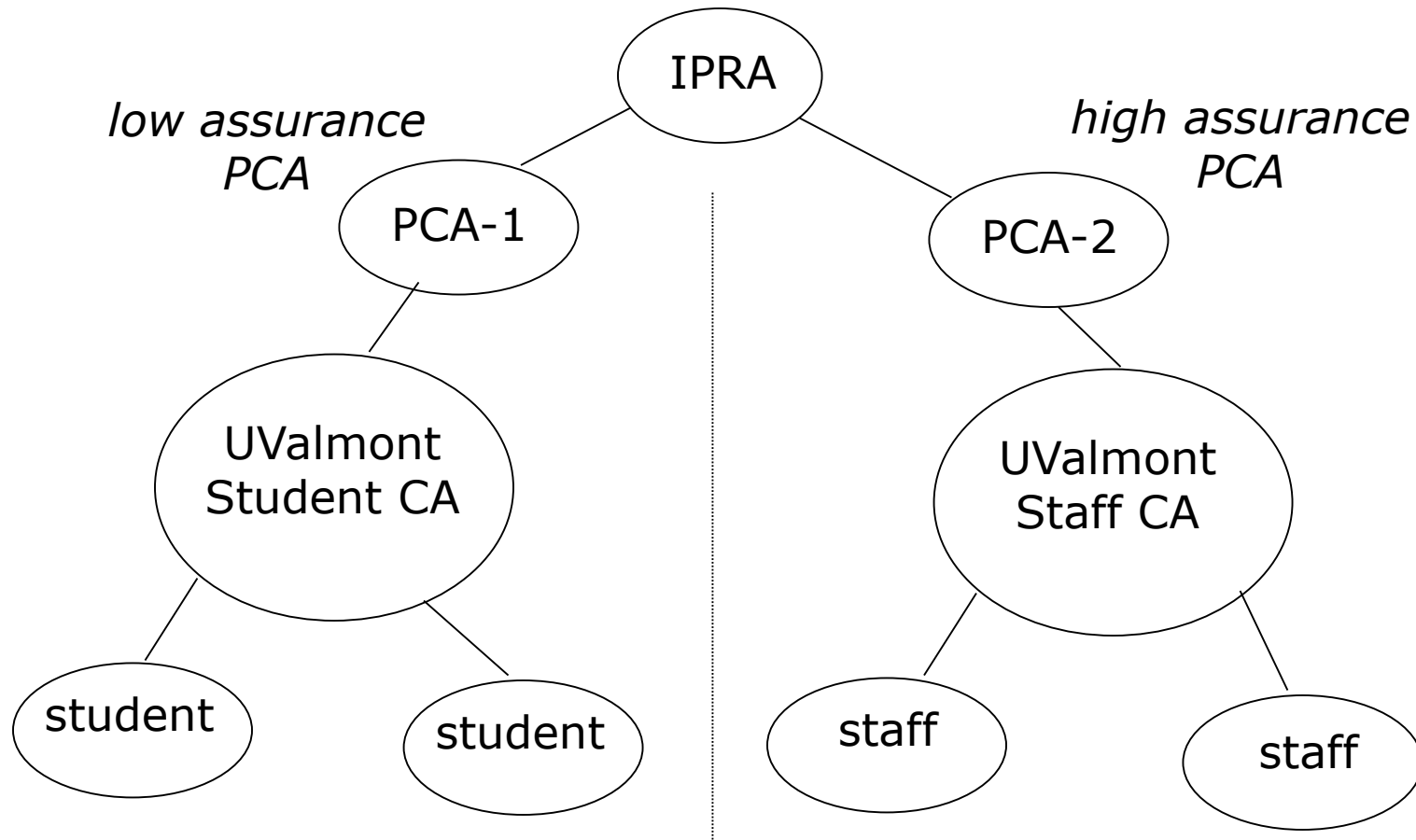
# UValmont and PCAs

---

- ❑ First PCA: requires subordinate CAs to make good-faith effort to verify identities of principals to whom it issues certificates
  - Student authentication requirements meet this
- ❑ Second PCA: requires use of biometrics to verify identity
  - Student authentication requirements do not meet this
  - Staff authentication requirements do meet this
- ❑ UValmont establishes two CAs, one under each PCA above

# UValmont and Certification Hierarchy

---



# Certificate Differences

---

- ❑ Student, staff certificates signed using different private keys (for different CAs)
  - Student's signed by key corresponding to low assurance certificate signed by first PCA
  - Staff's signed by key corresponding to high assurance certificate signed by second PCA
- ❑ To see what policy used to authenticate:
  - Determine CA signing certificate, check its policy
  - Also go to PCA that signed CA's certificate
    - ❑ CAs are restricted by PCA's policy, but CA can restrict itself further



# Types of Certificates

---

## ❑ Organizational certificate

- Issued based on principal's affiliation with organization
- Example Distinguished Name  
/O=University of Valmont/OU=Computer Science  
Department/CN=Marsha Merteuille/

## ❑ Residential certificate

- Issued based on where principal lives
- No affiliation with organization implied
- Example Distinguished Name  
/C=US/SP=Louisiana/L=Valmont/PA=1 Express  
Way/CN=Marsha Merteuille/

# Certificates for Roles

---

- Certificate tied to a role

- Example

- UValmont wants comptroller to have a certificate

- This way, she can sign contracts and documents digitally

- Distinguished Name

- /O=University of Valmont/OU=Office of the Big Bucks/RN=Comptroller

- where “RN” is *role name*; note the individual using the certificate is not named, so no CN

# Meaning of Identity

---

- ❑ Authentication validates identity
  - CA specifies type of authentication
  - If incorrect, CA may misidentify entity unintentionally
- ❑ Certificate binds *external* identity to crypto key and Distinguished Name
  - Need confidentiality, integrity, anonymity
    - ❑ Recipient knows same entity sent all messages, but *not* who that entity is

# Persona Certificate

---

- ❑ Certificate with meaningless Distinguished Name
  - If DN is  
/C=US/O=Microsoft Corp./CN=Bill Gates/  
the real subject may not (or may) be Mr. Gates
  - Issued by CAs with persona policies under a PCA with  
policy that supports this
- ❑ PGP certificates can use any name, so provide this  
implicitly

# Example: Whistleblower

---

- ❑ Government requires all citizens with gene Y to register
  - Anecdotal evidence people with this gene become criminals with probability 0.5.
  - Law to be made quietly, as no scientific evidence supports this, and government wants no civil rights fuss
- ❑ Government employee wants to alert media
  - Government will deny plan, change approach
  - Government employee will be fired, prosecuted
- ❑ Must notify media anonymously

# Example: Whistleblower

---

- ❑ Employee gets persona certificate, sends copy of plan to media
  - Media knows message unchanged during transit, but not who sent it
  - Government denies plan, changes it
- ❑ Employee sends copy of new plan signed using same certificate
  - Media can tell it's from original whistleblower
  - Media cannot track back whom that whistleblower is

# Trust

---

- ❑ Goal of certificate: bind correct identity to DN
- ❑ Question: what is degree of assurance?
- ❑ X.509v3, certificate hierarchy
  - Depends on policy of CA issuing certificate
  - Depends on how well CA follows that policy
  - Depends on how easy the required authentication can be spoofed
- ❑ Really, estimate based on the above factors

# Example: Passport Required

---

- ❑ DN has name on passport, number and issuer of passport
- ❑ What are points of trust?
  - Passport not forged and name on it not altered
  - Passport issued to person named in passport
  - Person presenting passport is person to whom it was issued
  - CA has checked passport and individual using passport



# PGP Certificates

---

- ❑ Level of trust in signature field
- ❑ Four levels
  - Generic (no trust assertions made)
  - Persona (no verification)
  - Casual (some verification)
  - Positive (substantial verification)
- ❑ What do these mean?
  - Meaning not given by OpenPGP standard
  - Signer determines what level to use
  - Casual to one signer may be positive to another

# Example: Whistleblower

---

## □ Story of a whistleblower

*From: anon108@■■■■■■■■■■*

*To: Micah Lee*

*Date: Fri, 11 Jan 2013*

*Micah,*

*I'm a friend. I need to get information securely to Laura Poitras her alone, but I can't find an email/gpg key for her.*

*Can you help?*

*From: Laura Poitras*

*To: Micah Lee*

*Date: Sat, 12 Jan 2013*

*Hey Micah,*

*Thanks for asking. Sure, you can tell this person I can be reached with GPG at: laurapoitras@gmail.com*

*I'll reply with my public key.*

*I'm also on jabber/OTR at:  
l.p.@jabber.org*

*I hope all is good with you!*

*Laura*

# Example: Whistleblower

## □ Story of a whistleblower continues

*From: 303@riseup.net*

*To: Micah Lee*

*Date: Mon, 28 Jan 2013*

*Hey Micah,*

*This is Laura Poitras.*

*Someone is trying to verify my fingerprint to this email. The person has proposed you tweet the fingerprint. Would you be able to tweet this to your acct:*

*1EBF 5F15 850C 540B 3142 F158 4BDD 496D 4C6C 5F25*

*Let me know if possible.*

*Thanks,*

*Laura*



**Micah Lee**

@micahflee

Follow

1EBF 5F15 850C 540B 3142 F158 4BDD 496D 4C6C  
5F25

11:56 PM - 28 Jan 2013

1 FAVORITE



# Summary

---

- ❑ Identity specifies a principal (unique entity)
  - Same principal may have many different identities
    - ❑ Function (role)
    - ❑ Associated principals (group)
    - ❑ Individual (user/host)
  - These may vary with view of principal
    - ❑ Different names at each network layer, for example