Security Engineering
Fall 2015

Lecture 11 – Authentication

Fabio Massacci

What is authentication?

- ***It is the process of verifying a claimed identity by r for a system entity***
- ***It consists of two main steps:***
  - – Identification
    - Present an identifier to the security system
    - You annouce who you are
  - – Verification
    - Presenting or generating authentication Information that provides evidence of the binding between the entity and the identifier
    - You prove who you are
- ***Remember: you are authenticating a stranger***

Means of Authentication

- ***Something the individual knows***
  - – Password-based
- ***Something the individual owns***
  - – Token-based
- ***Something the individual is***
  - – Static biometric
- ***Something the individual does***
  - – Dynamic biometrics
- ***Somewhere the individual is***
  - – Location-based

Something You Know

- ***The user has to know some secret to be authenticated.***
  - – password,
  - – personal identification number (PIN),
  - – personal information like home address, date of birth, name of mother maiden name (used e.g. by banks to authenticate customers on the phone)
- ***Password-based authentication***
  - – user provides name/login and password
  - – system compares password with that saved for specified login
  - – authenticates ID of user wishing to log
  - – AC starts from that user's ID

Password Authentication

- ***Typical issues that need to be addressed:***
  - how to get the password to the user,
  - forgotten passwords,
  - password guessing,
  - protection of the password file
- ***Dangers***
  - User accounts without passwords.
  - Unchanged default passwords.
  - Badly chosen passwords – dictionary/brute force attacks.
  - Passwords stored in the clear.
  - Passwords transmitted in the clear.
  - Users forget passwords
    - the infrastructure for re-issuing passwords can be quite expensive (if it has to be truly secure)

Password Choices

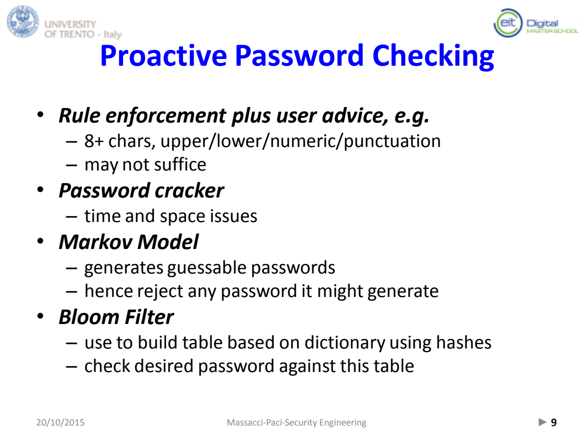
- ***users may pick short passwords***
  - e.g. 3% were 3 chars or less, easily guessed
  - system can reject choices that are too short
- ***users may pick guessable passwords***
  - so crackers use lists of likely passwords
  - e.g. one study of 14000 encrypted passwords guessed nearly 1/4 of them
  - would take about 1 hour on fastest systems to compute all variants, and only need 1 break!

Old Password Guessing Tools

- ***Hydra http://www.thc.org***
  - guess all sorts of passwords, including HTTP, Telnet, and Windows logons
- ***TSGrinder http://www.hammerofgod.com/download.htm***
  - for brute-force attacks against Terminal Services and RDP connections
- ***SQLRecon http://www.sqlsecurity.com/DesktopDefault.aspx?tabid=26)***
  - for brute-force attacks against SQL authentication

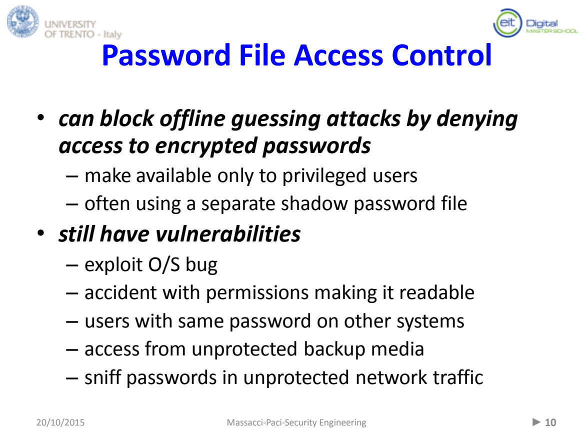
Password Cracking

- ***dictionary attacks***
  - try each word then obvious variants in large dictionary against hash in password file
- ***rainbow table attacks***
  - precompute tables of hash values for all salts
  - a mammoth table of hash values
  - e.g. 1.4GB table cracks 99.9% of alphanumeric Windows passwords in 13.8 secs
  - not feasible if larger salt values used



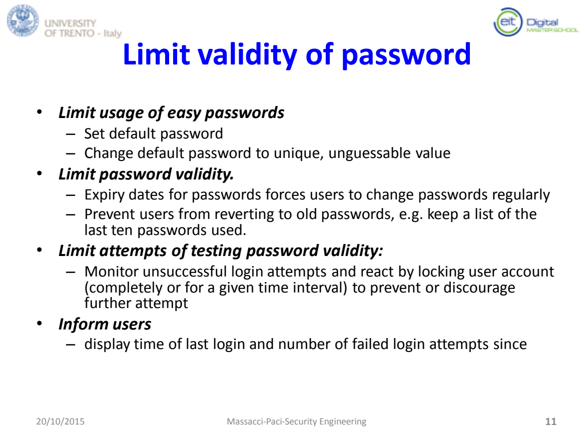
Proactive Password Checking

- ***Rule enforcement plus user advice, e.g.***
  - 8+ chars, upper/lower/numeric/punctuation
  - may not suffice
- ***Password cracker***
  - time and space issues
- ***Markov Model***
  - generates guessable passwords
  - hence reject any password it might generate
- ***Bloom Filter***
  - use to build table based on dictionary using hashes
  - check desired password against this table



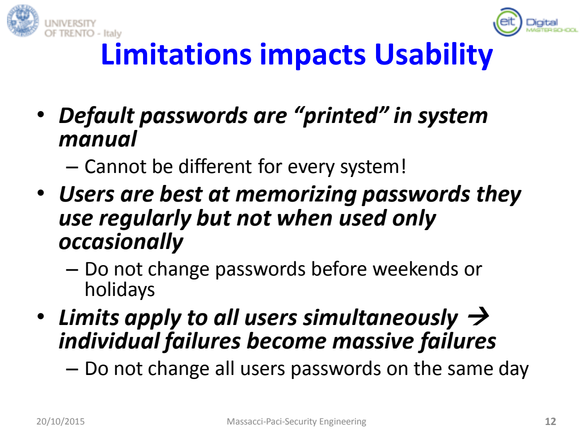
Password File Access Control

- ***can block offline guessing attacks by denying access to encrypted passwords***
  - make available only to privileged users
  - often using a separate shadow password file
- ***still have vulnerabilities***
  - exploit O/S bug
  - accident with permissions making it readable
  - users with same password on other systems
  - access from unprotected backup media
  - sniff passwords in unprotected network traffic



Limit validity of password

- ***Limit usage of easy passwords***
  - Set default password
  - Change default password to unique, unguessable value
- ***Limit password validity.***
  - Expiry dates for passwords forces users to change passwords regularly
  - Prevent users from reverting to old passwords, e.g. keep a list of the last ten passwords used.
- ***Limit attempts of testing password validity:***
  - Monitor unsuccessful login attempts and react by locking user account (completely or for a given time interval) to prevent or discourage further attempt
- ***Inform users***
  - display time of last login and number of failed login attempts since



Limitations impacts Usability

- ***Default passwords are "printed" in system manual***
  - Cannot be different for every system!
- ***Users are best at memorizing passwords they use regularly but not when used only occasionally***
  - Do not change passwords before weekends or holidays
- ***Limits apply to all users simultaneously → individual failures become massive failures***
  - Do not change all users passwords on the same day

Bootstrapping authentication

- ***Passwords are secrets shared between user and system***
  - "The" user is whoever knows the secret
- ***How do you bootstrap a system so that the password ends up in the right places, but nowhere else?***
  - In an enterprise, users can collect their password personally.
  - In Web applications you want to deal with remote users.

(Weak) authentication of a remote user

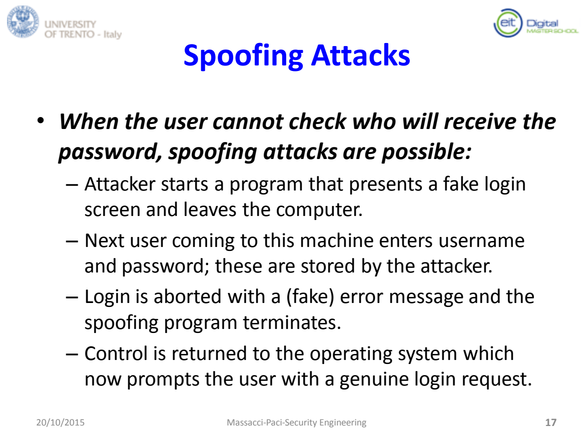
- ***For remote users, passwords could be sent by mail, email, or phone, or entered by the user on a web page.***
- ***"Normally" your forgotten password is sent to your email address.***
  - Ability to reading an email is a proxy for your ability to know the password → to read the email you must know a password
  - How secure is that?
- ***You have to consider who might intercept the message and who might actually pick it up.***
  - E.g., a letter containing the password for an online bank account might be stolen or an impersonator may phone in asking for another user's password.

(Stronger) Authention of a Remote User

- ***Send passwords that are valid only for a single log-in request so that the user has to change immediately to a password not known by the sender***
  - Assume attacker does not control server's email, backbone network, local network, local email
- ***Request confirmation on a different channel to activate user account,***
  - Enter password on a webpage and send confirmation by SMS.
  - Send mail by courier with personal delivery.
- ***In an organisation:***
  - Don't give password to caller but call back an authorized phone number, e.g. from an internal company address book.
  - Call back someone else, e.g. caller's manager or local security officer.
- ***More details later when we discuss application authentication***

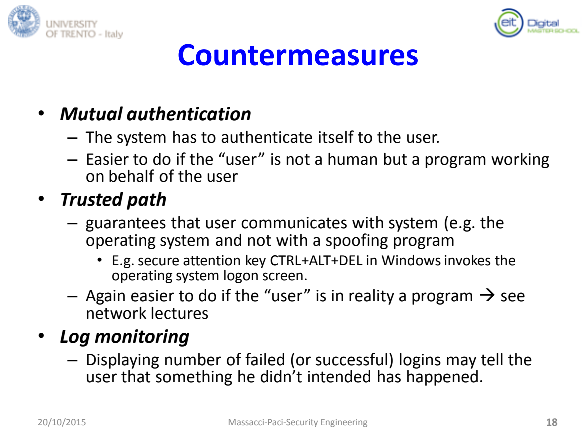
Resetting Passwords

- ***When setting up a new user account some delay in getting the password may be tolerated.***
- ***If you have forgotten your password but are in the middle of an important task you need instant help.***
- ***The procedures for resetting a password are the same as mentioned previously, but now instant reaction is desirable.***
  - In global organisations a hot desk has to be available round the clock.
  - Proper security training has to be given to personnel at the hot desk → e.g. call back



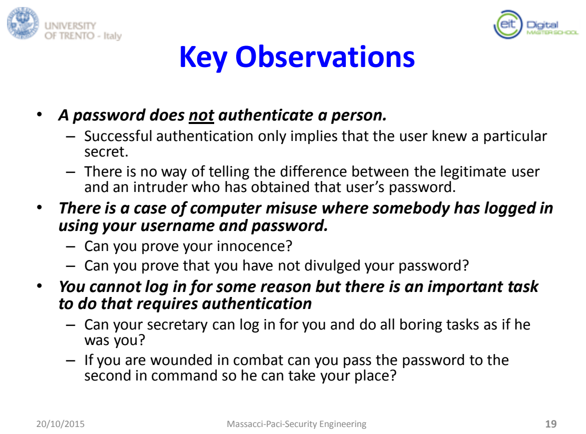
Spoofing Attacks

- ***When the user cannot check who will receive the password, spoofing attacks are possible:***
  - Attacker starts a program that presents a fake login screen and leaves the computer.
  - Next user coming to this machine enters username and password; these are stored by the attacker.
  - Login is aborted with a (fake) error message and the spoofing program terminates.
  - Control is returned to the operating system which now prompts the user with a genuine login request.



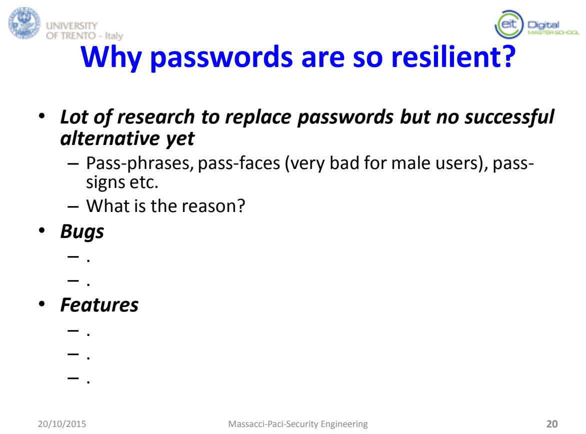
Countermeasures

- ***Mutual authentication***
  - The system has to authenticate itself to the user.
  - Easier to do if the "user" is not a human but a program working on behalf of the user
- ***Trusted path***
  - guarantees that user communicates with system (e.g. the operating system and not with a spoofing program
    - E.g. secure attention key CTRL+ALT+DEL in Windows invokes the operating system logon screen.
  - Again easier to do if the "user" is in reality a program → see network lectures
- ***Log monitoring***
  - Displaying number of failed (or successful) logins may tell the user that something he didn't intended has happened.



Key Observations

- ***A password does <u>not</u> authenticate a person.***
  - Successful authentication only implies that the user knew a particular secret.
  - There is no way of telling the difference between the legitimate user and an intruder who has obtained that user's password.
- ***There is a case of computer misuse where somebody has logged in using your username and password.***
  - Can you prove your innocence?
  - Can you prove that you have not divulged your password?
- ***You cannot log in for some reason but there is an important task to do that requires authentication***
  - Can your secretary can log in for you and do all boring tasks as if he was you?
  - If you are wounded in combat can you pass the password to the second in command so he can take your place?



Why passwords are so resilient?

- ***Lot of research to replace passwords but no successful alternative yet***
  - Pass-phrases, pass-faces (very bad for male users), pass-signs etc.
  - What is the reason?
- ***Bugs***
  - .
  - .
- ***Features***
  - .
  - .
  - .

Why passwords are so resilient?

- ***Lot of research to replace passwords but no successful alternative yet***
  – Pass-phrases, pass-faces (very bad for male users), pass-signs etc.
  – What is the reason?
- ***Bug***
  – You only need a keyboard to generate your secrete
  – Anybody who obtains your secret is “you”.
  – You leave no trace if you pass your secret to somebody else.
- ***Feature***
  – You only need a keyboard to generate your secret
  – Anybody who obtains your secret is “you”.
  – You leave no trace if you pass your secret to somebody else.

Something You Hold

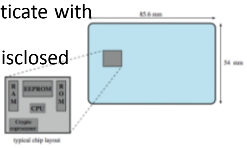
- ***The user has to present a physical token to be authenticated.***
  – In the past: keys (for self-access), seals (for access monitored by humans)
  – Today: Cards or identity tags (access to buildings), smart cards.
- ***Feature + Bug***
  – Anybody who is in possession of the token has the same rights as the legitimate owner.
  – Physical tokens can be lost or stolen without the user’s cooperation
- ***To increase security, physical tokens are often used in combination with something that cannot be stolen***
  – bank cards come with a PIN or with a photo of the user.

Memory Card

- ***store but do not process data***
- ***magnetic stripe card, e.g. bank card***
- ***electronic memory card***
- ***used alone for physical access***
- ***with password/PIN for computer use***
- ***drawbacks of memory cards include:***
  – need special reader
  – loss of token issues
  – user dissatisfaction

Smartcard

- ***Smartcard has own processor, memory, I/O ports***
  – wired or wireless access by reader
  – may have crypto co-processor
  – ROM, EEPROM, RAM memory
- ***Can store secrets***
  – executes protocol to authenticate with reader/computer
  – secrets are “used” but not disclosed
  – secrets are tamperproof
- ***Alternative: USB dongles***



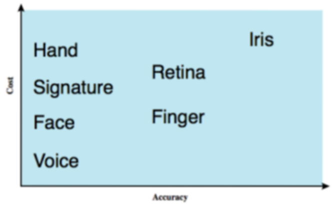
typical chip layout

Who You Are

- ***Biometric schemes use unique physical characteristics (traits, features) of a person***
  - face,
  - finger prints,
  - iris patterns,
  - hand geometry
- ***Biometrics may seem to offer the most secure solution for authenticating a person***
  - Very good for specialized/limited access → e.g. access to ACC may require biometric authentication
- ***Little experience from large scale field trials on the performance of biometrics***
  - So far only large scale is biometric on mobile devices, but not know if most people actually turned that on

Biometric Authentication

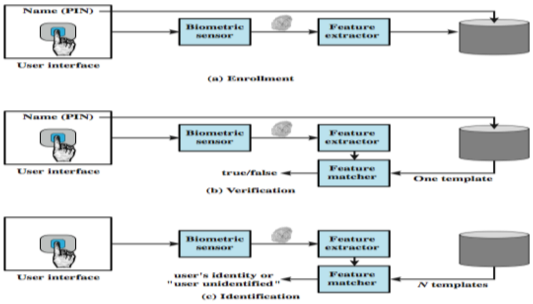
- ***authenticate user based on one of their physical characteristics***



Biometrics

- ***Use physical traits unique for each individual:***
  - Fingerprints
  - Iris patterns
- ***Biometric authentication (1:1 comparison, also called verification):***
  - Register biometric sample (fingerprint).
  - For authentication, compare new biometric sample with the user's registered reference value.
- ***Biometric identification (1:n comparison):***
  - Take biometric sample and compare against a database of samples to find out who the user is.

Operation of a Biometric System

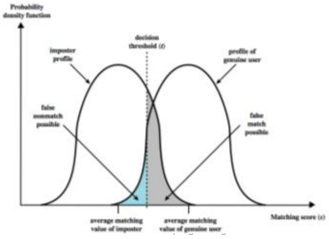


Enrolment and Authentication

- ***Enrolment:***
  - A reference template of the user’s fingerprint is acquired at a fingerprint reader.
  - Templates are stored in a secure database.
- ***Failure-to-enrol (FTR):***
  - not every person has usable fingerprints.
  - For higher accuracy, several templates may be recorded, possibly for more than one finger.
- ***Authentication***
  - When the user logs on, a new reading of the fingerprint is taken and compared against the reference template.

Biometric Accuracy

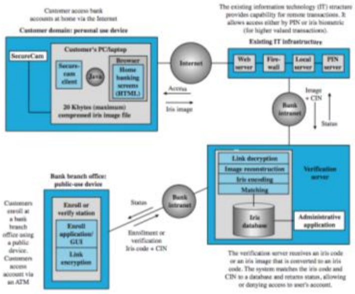
- ***never get identical templates***
- ***problems of false match / false non-match***

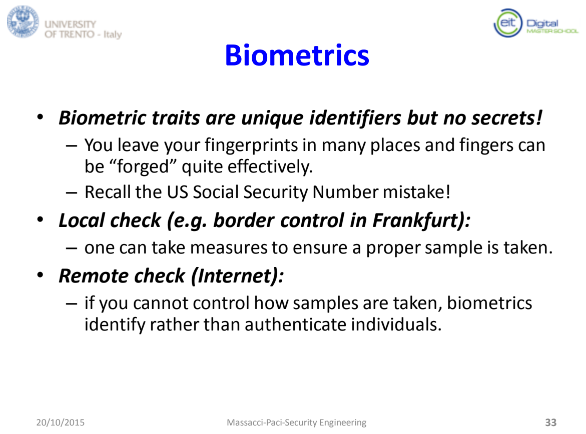


Biometrics

- ***Current and registered sample will never match perfectly → a user will be accepted if the match scores above a given threshold***
- ***False acceptance rate (FAR):***
  - wrong user accepted; problem in sensitive areas
- ***False rejection rate (FRR):***
  - user wrongly rejected; problem in commercial areas
- ***Equal error rate (EER):***
  - threshold set so that FAR= FRR
  - Best EER for fingerprint systems 1-2%; iris recognition has better performance.
- ***In practice threshold depends on application***

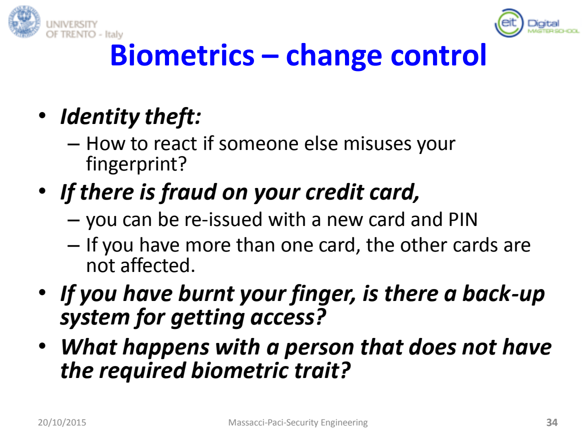
Practical Application


Customer access bank accounts at home via the Internet
Customer domain: personal use device
Customer's PC/laptop
SecureCam
Secure-cam client
Java
Browser
Home banking screens (HTML)
20 Kbytes (maximum) compressed iris image file
Internet
Access
Iris image
The existing information technology (IT) structure provides capability for remote transactions. It allows access either by PIN or iris biometric (for higher valued transactions).
Existing IT infrastructure
Web server
Fire-wall
Local server
PIN server
Bank intranet
Image + CIN
Status
Link decryption
Image reconstruction
Iris encoding
Matching
Verification server
Iris database
Administrative application
Bank branch office: public-use device
Customers enroll at a bank branch office using a public device. Customers access account via an ATM
Enroll or verify station
Enroll application/ GUI
Link encryption
Status
Bank intranet
Enrollment or verification Iris code + CIN
The verification server receives an iris code or an iris image that is converted to an iris code. The system matches the iris code and CIN to a database and returns status, allowing or denying access to user's account.



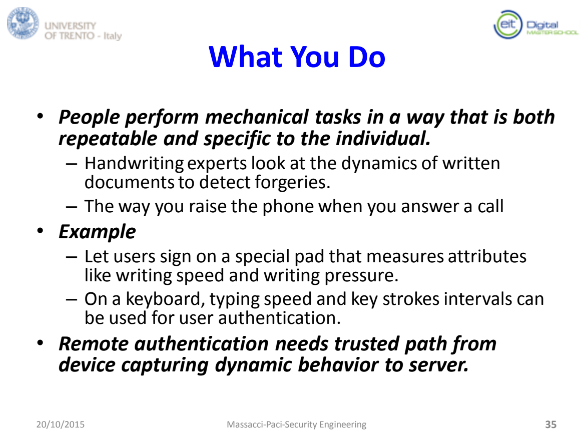
Biometrics

- ***Biometric traits are unique identifiers but no secrets!***
  - You leave your fingerprints in many places and fingers can be “forged” quite effectively.
  - Recall the US Social Security Number mistake!
- ***Local check (e.g. border control in Frankfurt):***
  - one can take measures to ensure a proper sample is taken.
- ***Remote check (Internet):***
  - if you cannot control how samples are taken, biometrics identify rather than authenticate individuals.



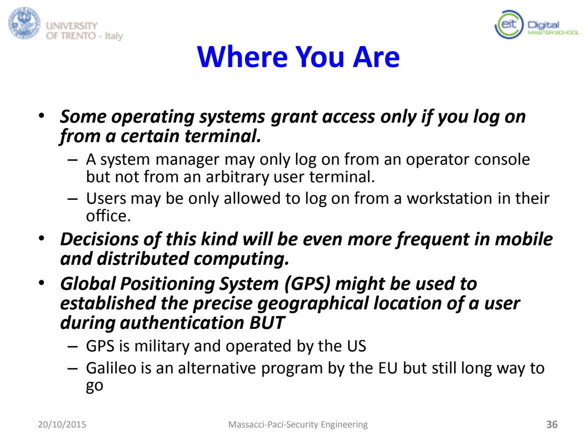
Biometrics – change control

- ***Identity theft:***
  - How to react if someone else misuses your fingerprint?
- ***If there is fraud on your credit card,***
  - you can be re-issued with a new card and PIN
  - If you have more than one card, the other cards are not affected.
- ***If you have burnt your finger, is there a back-up system for getting access?***
- ***What happens with a person that does not have the required biometric trait?***



What You Do

- ***People perform mechanical tasks in a way that is both repeatable and specific to the individual.***
  - Handwriting experts look at the dynamics of written documents to detect forgeries.
  - The way you raise the phone when you answer a call
- ***Example***
  - Let users sign on a special pad that measures attributes like writing speed and writing pressure.
  - On a keyboard, typing speed and key strokes intervals can be used for user authentication.
- ***Remote authentication needs trusted path from device capturing dynamic behavior to server.***



Where You Are

- ***Some operating systems grant access only if you log on from a certain terminal.***
  - A system manager may only log on from an operator console but not from an arbitrary user terminal.
  - Users may be only allowed to log on from a workstation in their office.
- ***Decisions of this kind will be even more frequent in mobile and distributed computing.***
- ***Global Positioning System (GPS) might be used to established the precise geographical location of a user during authentication BUT***
  - GPS is military and operated by the US
  - Galileo is an alternative program by the EU but still long way to go

Remote User Authentication

- ***authentication over network more complex***
  - problems of eavesdropping, replay
- ***generally use challenge-response***
  - user sends identity
  - host responds with random number N
  - user computes some function with N that only user can generate and sends back
  - host compares value from user with own computed value (or other similar function), if match user authenticated
- ***protects against a number of attacks***
- ***More of this in the application security part***

Challenge Response - II

- ***The Simplest protocol***
  - A & B agrees on some parameters off line
  - A → B: I'm A
  - B → A: Nonce = random number
  - B → A: f(B,Nonce) = function that only B can make but that A can check
  - A : ok
- ***Example instantiation***
  - A, B share secret S
  - …
  - B → A: Hash(S.B.A.Hash(S.Nonce)) provided Hash is a function that is easy to compute but hard to invert.
    - Why this is better than sending H(S.B.A.N)?
    - Why this better than sending H(S.B.N)? Why H(S.A.N) is worst of all?
    - When Hash(Hash(S).B.A.Hash(Hash(S).N)) would be desirable?
- ***Another example: send secure code via phone***
  - How really secure is that?

Beware: Security mechanisms fail

- ***Two equally important worries:***
  - failures that wrongly permit an action
  - failures that wrongly deny access
- ***Forgotten passwords, lost token, false biometric rejection, too frequent re-authentication, etc. etc.***
  - If not adequately addressed → system not available to legitimate users
  - If you believe your technology is perfect (or forget about this issue) → your system will fail

Reading Material

- ***Chapter 2. Stallings & Brown. Computer Security Principles and Practice.***
- ***Papers on password studies by***
  - Angela Sasse at UCL on what doesn't work
  - Frank Stajano at Cambridge on large studies and possible alternatives
- ***More sophisticated methods based on credentials***
  - See OpenAuth white paper

Recaps: Types of Access Control

- ***Discretionary Access Control***
  - Policy decided by individual subjects
  - Access based on identity of subjects
- ***Role based Access Control***
  - Policy decided by system
  - Subjects assigned to Roles,
  - (Action,Objects) assigned to Roles
  - Access based on roles activated by subjects
- ***Mandatory Access Control***
  - Policy decided by system
  - Subject assigned to security levels (clearance),
  - Object assigned to security labels
  - Access based on matching objects' labels to subjects' clearances
- ***Credential based Access Control***
  - Access based on attributes qualifying a subject
    - Essentially "self-service" PIP signed by accredited PAPs

Mandatory Access Control

- ***Organization Access Policy is always MAC***
  - I do not decide who can read the grades of my course
- ***Implements***
  - Legislation
  - Commercial Confidentiality – Integrity requirements
  - Paranoia of Board of Directors
  - Pet projects of the above (security holes)
- ***Any policy can be specified → enough to have gigantic tables***
  - Objects → Labels
  - Subject → Labels
  - Match: Action x Object x Subject → {True/False}
- ***Example on RedHat Security Enhanced Linux***
  - "TE uses a matrix of domains and object types derived from the policy. "
  - allow httpd_t net_conf_t:file { read getattr lock ioctl }; gives the domain associated with httpd [=subject] the permissions to read data out of specific network configuration files [=object] such as /etc/resolv.conf.
- ***Example on TSA for flying armed [=object]***
  - Subject [=subject] must be Federal Law Enforcement Officer AND ….
  - Be commissioned to enforce criminal statutes or immigration statutes AND
  - Be authorized by the employing agency to have the weapon in connection with assigned duties:
  - provision of protective duties… OR control of a prisoner… OR …

Security Models

- ***MAC is complicated…***
  - "For Red Hat Enterprise Linux 4 the policy has been designed to restrict only a specific list of daemons. All other processes run in an unconfined state. This policy is designed to help integrate SELinux into your development and production environment. It is possible to have a much more strict policy, which comes with an increase in maintenance complexity."
- ***Security Model = MAC with specific focus***
  - Policy encodes some "default" action in the match function
- ***Security Models allows***
  - Simplification of matching process (essential for humans, less for computers)
  - Simplification of administration
  - Formal verification of security

Bell-LaPadula Confidentiality Model

- ***BLP is a model that covers the confidentiality aspects of access control***
  - Initially invented for the military
  - OS Multics Operating Systems
  - Implemented in physical security
    - Eg photocopier won't copy document with a "Top Secret" mark
- ***Prevents low-security level subjects to read high-security level objects***
- ***Consider information flows when a subject reads or alters an object***

BLP Components

- ***S - set of subjects***
- ***O - set of objects***
- ***A - set of access operations***
  - read, write, append, execute
- ***L - set of partially ordered security levels***
  - Top secret > secret > confidential > unclassified

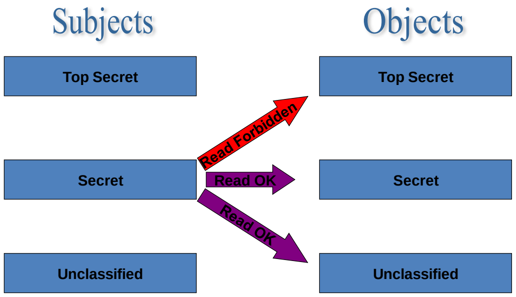
BLP State: assign security levels

- ***fs: S ➔ L***
  - Assign to a subject the maximum security level
- ***fc: S ➔ L***
  - Assign to a subject the current security level
- ***fo: O ➔ L***
  - Assign to an object its security level
- ***The security level assigned to a subject is also called security clearance***

BLP properties – ss property

- ***A subject can only read an object of less or equal security level***
- ***Formally***
  - A system satisfy the simple security property if for every granted read access the security level of the subject s dominates the security level of the object o
  - fo (o) ≤ fs (s)
- ***Also known as no read-up security policy***

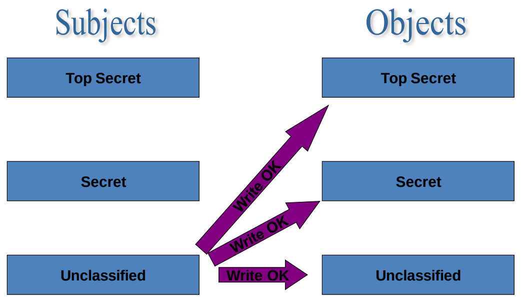
BLP properties: ss property - II



BLP properties - * property

- ***A subject can only write objects of greater or equal security level***
- ***Formally***
  - A system satisfies the * property if for every granted write/modify request the security level of the subject o dominates the security level of the object o
  - fs (s) ≤ fo (s)
- ***Also known as no write-down policy***

BLP properties - * property - II



The Basic Security Theorem

- ***A state is secure, if all current assignment of permissions to subjects satisfies the ss-property, ∗ - property.***
- ***A state transition is secure if it goes from a secure state to a secure state***
- ***Basic Security Theorem***
  - If all the transitions are secure and the intial state is secure all the subsequent states will be secure regardless the input

BLP properties - * property limitation

- ***The ∗ - property implies that a high level subject is not able to send messages to a low level subject***
  - How can a general send an email to the secretary?
- ***There are several ways to escape from this restriction***
  - Allow a human to work at the same time on two systems
    - That was the original implementation.
  - Temporarily downgrade a high level subject. This is the reason for the current security level f_C.
  - Identify a set of trusted subjects, which are permitted to violate the ∗ - property.
  - Have a "declassification" function to downgrade some information

Tranquillity

- ***McLean: consider a system with an operation downgrade:***
  - downgrades all subjects to system low
  - downgrades all objects to system low
  - enters all access rights in all positions of the access control matrix
- ***The resulting state is secure according to BLP***
- ***Should such a system be regarded as secure?***
  - McLean: no, everybody is allowed to do everything
  - Bell: yes, if downgrade was part of the system specification
- ***Fact: BLP assumes tranquility, i.e. access control rules do not change "on-the-fly"***

Limitations of Bell-LaPadula

- ***Restricted to confidentiality***
- ***No policies for changing access rights***
  - A general and complete downgrade is secure
  - However, BLP is intended for systems with static security levels
- ***BLP contains covert channels***
  - Information flow that is not controlled by the model

Covert Channels

- ***Covert channels are information channels that are not controlled by the security mechanism of the system***
- ***Information can flow (leak) from a high security level to a low security level***
  - A subject assigned to a low-security level can detect the existence of an high-security level object when it is denied access
  - Sometimes, it is not sufficient to hide only the content of objects. Also their existence may have to be hidden.
- ***Telling a subject that a certain operation is not permitted constitutes information flow***

Bell-LaPadula Example

- ***ESSE3 Clearances***
  - Students' Secretariat > Professor > Assistant > Student
  - Not really true (ESSE3 is RBAC not BLP)
- ***Kate is a teacher for the Security Engineering course → clearance A***
  - She can login into the esse3 system as teacher and as student
- ***Andrea is student enrolled in the Security Engineering course → clearance S***
  - He can only login as student

Bell-LaPadula Example

- *Kate*
  - creates file f1 with P security level
- *Andrea*
  - creates file f2 with S security level
- *Is Kate*
  - authorized to read f2?
  - authorized to write f2?
- *Kate*
  - creates an exam file f3 with A security level
- *Is Andrea*
  - authorized to read the f3?

Biba Integrity Model

- ***State-machine model similar to BLP which focuses on integrity aspects of access control***
- ***Focus on preventing unauthorized modifications of data***
- ***Access permission based on***
  - Assignment of subjects and objects to integrity levels
- ***Prevents information flow from low-integrity levels to high-integrity levels***

Biba Integrity Model Components

- ***S – set of subjects***
- ***O – set of objects***
- ***A – set of access operations***
  - modify, observe, execute, invoke
- ***fs: S ➔ L***
  - Assign to a subject the integrity level
- ***fo: O ➔ L***
  - Assign to an object its integrity level

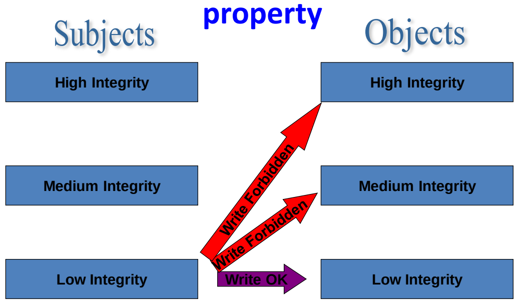
•

Biba Integrity Model properties: si property

- ***A subject can modify an object only if the integrity level of the subject dominates the integrity level of the object***
- ***Formally***
  - A subject s can modify (alter) an object o if fs (s) ≥ fo (s)
- ***Also known as no write-up policy***

–

Biba Integrity Model properties: si property



Subjects

Objects

High Integrity

Medium Integrity

Low Integrity

Write Forbidden

Write Forbidden

Write OK

High Integrity

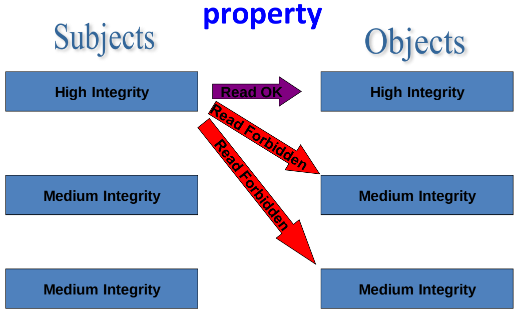
Medium Integrity

Low Integrity

Biba Integrity Model properties: property

- ***A subject can read an object only if the integrity level of the subject is dominated by the integrity level of the object***
- ***Formally***
  – A subject s can read (observe) an object o if fs (s) ≤ fo (s)
- ***Also known as no read-down policy***

Biba Integrity Model properties: property



Subjects

Objects

High Integrity

Medium Integrity

Medium Integrity

Read OK

Read Forbidden

Read Forbidden

High Integrity

Medium Integrity

Medium Integrity

Biba Integrity Model: dynamic integrity properties

- ***Automatically adjust subjects and objects assigned integrity levels***
- ***Subject Low Watermark Security Policy***
  – A subject s can read (observe) an object o at any integrity level. The new integrity level of the subject s is the greatest lower bound of fs (s) and fo (o).
- ***Object Low Watermark Security Policy***
  – A subject s can modify (alter) an object o at any integrity level. The new integrity level of the subject s is the greatest lower bound of fs (s) and fo(o).

Biba Integrity Model properties: invoke and ring property

- ***Invoke Property***
  - A subject is only authorized to invoke subjects (tools) at lower integrity levels
  - Formally
    - A subject s1 can invoke a subject s2 if fs (s2) ≤ fs (s1)
- ***Ring property***
  - A subject s can read objects at any integrity level. It can only modify objects o with fo (o) ≤ fs (s); it can invoke a subject s' only if fs (s) ≤ fs (s')

Biba Implementation in Vista

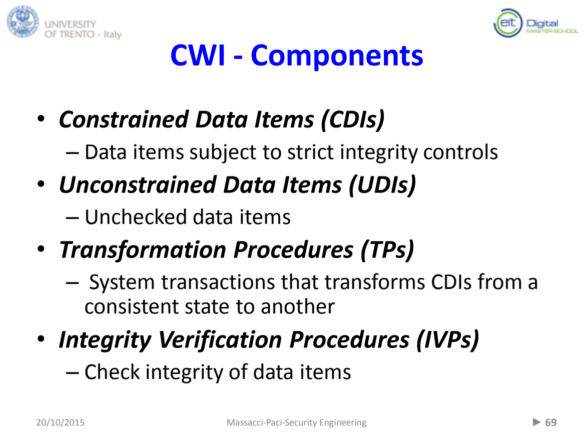
- ***Vista marks files with an integrity level***
  - Low, Medium, High and System
  - Critical files are assigned System integrity level
  - Other objects are assigned Medium integrity level
  - Internet Explorer is assigned Low integrity level
- ***Vista implements the no write-up policy***
  - Files downloaded form IE can read most of the files in Vista file system but cannot write them
  - Limit the damage done by viruses and malwares

Clark Wilson Integrity Model

- ***MAC Model + Emphasis on integrity***
  - internal consistency:
    - properties of the internal state of a system
  - external consistency:
    - relation of the internal state of a system to the outside world
- ***Access permission based on***
  - the assignment of subjects to trusted programs
  - Execution of trusted programs that mantains consistency
- ***May be applicable to you***
  - Instrumentd Flights Programs

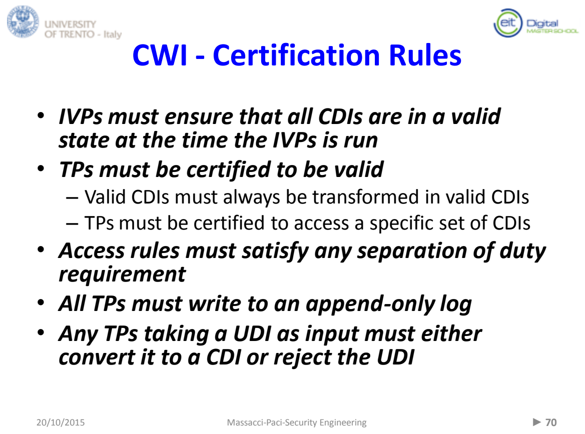
CWI - Mechanisms

- ***Well-formed transactions***
  - A user should only access data through trusted programs
- ***Separation of duty***
  - Any person permitted to create or certify a well-formed transaction should not be permitted to perform it



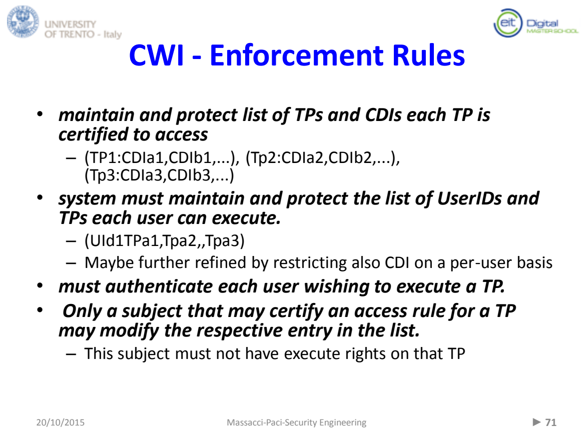
CWI - Components

- ***Constrained Data Items (CDIs)***
  - Data items subject to strict integrity controls
- ***Unconstrained Data Items (UDIs)***
  - Unchecked data items
- ***Transformation Procedures (TPs)***
  - System transactions that transforms CDIs from a consistent state to another
- ***Integrity Verification Procedures (IVPs)***
  - Check integrity of data items



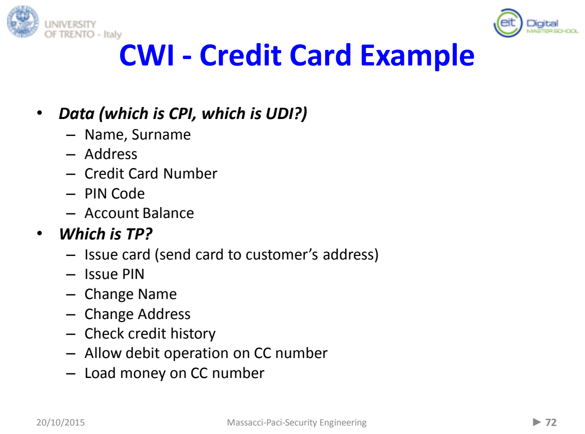
CWI - Certification Rules

- ***IVPs must ensure that all CDIs are in a valid state at the time the IVPs is run***
- ***TPs must be certified to be valid***
  - Valid CDIs must always be transformed in valid CDIs
  - TPs must be certified to access a specific set of CDIs
- ***Access rules must satisfy any separation of duty requirement***
- ***All TPs must write to an append-only log***
- ***Any TPs taking a UDI as input must either convert it to a CDI or reject the UDI***



CWI - Enforcement Rules

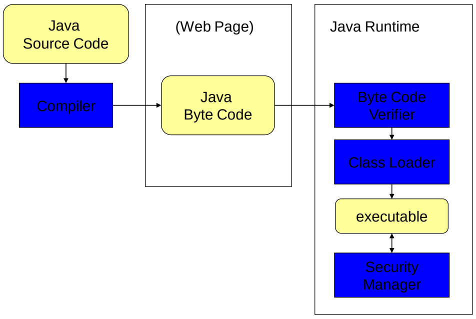
- ***maintain and protect list of TPs and CDIs each TP is certified to access***
  - (TP1:CDIa1,CDIb1,...), (Tp2:CDIa2,CDIb2,...), (Tp3:CDIa3,CDIb3,...)
- ***system must maintain and protect the list of UserIDs and TPs each user can execute.***
  - (UId1TPa1,Tpa2,,Tpa3)
  - Maybe further refined by restricting also CDI on a per-user basis
- ***must authenticate each user wishing to execute a TP.***
- ***Only a subject that may certify an access rule for a TP may modify the respective entry in the list.***
  - This subject must not have execute rights on that TP



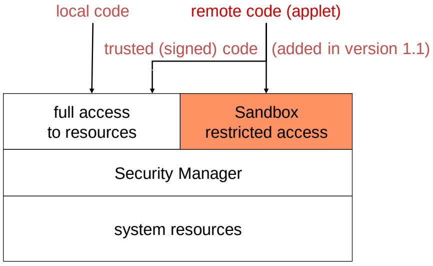
CWI - Credit Card Example

- ***Data (which is CPI, which is UDI?)***
  - Name, Surname
  - Address
  - Credit Card Number
  - PIN Code
  - Account Balance
- ***Which is TP?***
  - Issue card (send card to customer’s address)
  - Issue PIN
  - Change Name
  - Change Address
  - Check credit history
  - Allow debit operation on CC number
  - Load money on CC number

The Java Execution Model



JDK 1.1 Security Model



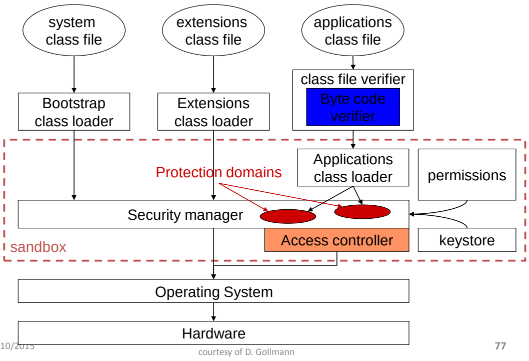
Discussion

- ***What kind of model is that?***

Limitations

- ***Local/remote is not a precise security indicator:***
  - Parts of the local file system could reside on other machines;
  - Downloaded software becomes “trusted” once it is cached or installed on the local system.
- ***Basic policy is quite inflexible:***
  - Local/signed code is unrestricted.
  - Applet/unsigned code is restricted to sandbox.
- ***No intermediate level:***
  - How to give some privileges to a home banking application?
- ***For more flexible security policies a customized security manager needed to be implemented.***
  - Requires security AND programming skills.

Java 2 Security Model



system class file

extensions class file

applications class file

Bootstrap class loader

Extensions class loader

class file verifier

Byte code verifier

Applications class loader

permissions

Protection domains

Security manager

Access controller

keystore

sandbox

Operating System

Hardware

courtesy of D. Gollmann

Terminology

- ***Security Policy***
  - …mapping from a set of properties characterizing code, to a set of resource access permissions granted to the code…
- ***Protection Domain:***
  - …encapsulation of the code characteristics: location, signers and static permission granted to the code…

courtesy of D. Gollmann

Code-based Access Control

- ***Security relevant parameters associated with code.***
  - Which parameters to use?
- ***Code source:***
  - URL (origin)
  - Digital certificates (code signers, if any)
- ***Principals: represent users or services***
- ***Protection domains: each class associated at load time with a protection domain.***
  - Contains: code source, principal, class loader reference, permission collection
- ***Question: is this really different from MAC+CAP?***

courtesy of D. Gollmann

Discussion of modern systems

- ***Operating Systems***
  - Linux + Free BSD (aka Mac OS X) → DAC + ACL
  - Android OS → DAC + ACL + elements of CAP
  - SELinux → MAC + ACL
  - Capsicum (Linux Variant) → MAC/DAC + CAP
- ***Virtual Machines***
  - Android VM + Java VM → ?
  - SurveyMonkey, V8 →
- ***ERP Systems***
  - SAP R3 OR Oracle → RBAC
  - SAP ByD → MAC + AC Matrix
- ***Banking systems***
  - In theory MAC+CWI
- ***Facebook , Gmail "Appiverse"***
  - ???