Secure Programming Techniques

Margus Freudenthal, Mart Oruaas

Lecture 3, 2022-02-25
Identity, Authentication, Access Control
Identity
What are We Talking About?

- **Social identity:** a social entity’s own understanding of its place in a wider society, its self-worth.
- **Philosophical identity:** deals with “sameness” of entities.
- Unless otherwise stated, computer software, esp. its security functions, deal with **philosophical identity**.
- There are tons of marketing materials around identity technologies that confuse the two.
ISO 24760 Terminology 1/2

See [ISO, 2019]

- **Identity**: set of attributes related to an entity
- **Identifier**: attribute or set of attributes that uniquely characterizes an identity in a domain
- **Reference identifier**: identifier that is intended to remain the same for the duration an entity is known in the domain and is not associated with another entity for a period specified in a policy after the entity ceases to be known in that domain
ISO 24760 Terminology 2/2

- **Identification**: process of recognizing an **entity** in a particular **domain** as distinct from other entities
- **Verification**: process of establishing that **identity information** associated with a particular **entity** is correct
- **Authentication**: formalized process of **verification** that, if successful, results in an authenticated identity for an **entity**
Issues with Identification

- Often there are no good identifiers for entities, especially people
  - In many countries, having an unique strong identifier is culturally unacceptable, so the identifying attributes contain name, date of birth, address and more
  - It is almost impossible to determine if two similar sets of attributes refer to the same person or not
- Many things used as identifiers are not reference identifiers
- Several popular identifiers (e-mails, mobile numbers) are not directly linked to persons
Question of Domain (Context)

- Identifiers are always defined in some domain, the same value may identify something else in a different domain.
- Defining an identifier in the global domain is problematic and requires coordination.
- An example: Estonian and Latvian (older) personal ID codes which overlap without adding the country code.
- ETSI EN 319 412-1 [ETSI, 2020, 5.1.3, 5.1.4] tries to offer a limited solution for entities in EU.
- Randomly generated long identifiers (e.g. UUID) seem to work the best, at least they do not clash accidentally.
Estonian Identity System

- A person has a single, unique, state-issued public identifier that all government and non-government organizations use.
- In a government system, there is no concept of a “username”.
- All identity documents contain the minimal amount of data absolutely necessary. The identifier allows an authorized person to get anything else from any registry.
- There is always a trace back to the original identity as new documents are issued based on previous ones.
- The state serves physical people, not digital abstractions.
Structure of Estonian Digital Identity

- A physical identity backed by a state-held registry (population register)
- The strong identity can be used to issue electronic identity documents (like the ID-card) and certificates that, again, contain the same citizen identifier
- A citizen can use their electronic identity to generate various secondary identity artifacts like authorization tokens or private authentication means
Estonian National Identity Structure

- Physical Identity Documents (Passport)
- Electronic Identity Documents (ID-Card, Mobile-ID)
- Secondary Electronic Identities (Tokens, Smart-ID)

Physical Identity
Authentication
Definition of “Authentication”

- “Authentication is the act of proving an assertion, such as the identity of a computer system user.” (Wikipedia)
- The definition sounds deceivingly simple and straightforward.
- First problem: see the issues with identity and its ambiguity.
- Second...
“Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. (They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed. But they are sufficiently pervasive that we must design our protocols around their limitations.)”

[Kaufman et al., 1995]
The Nature of Authentication in Software Systems

- The result of authentication is always some kind of state change in the system to which the user authenticates themselves.
- The *meaning* and nature of that state change must be clear and explicit (i.e., it is not a side effect of some business logic state change).
- The state must be carried over to every subsequent request. Also, the state must be revokable.
- One possible failure mode is that an authenticated sequence of requests allows users to obtain a non-secure handle (for example, an unpublished but guessable URL) for issuing subsequent requests without any authentication checks.
Authentication Factors

- Knowledge – what the user knows: passwords, “secret” questions
- Possession – what the user has: keys, tokens, devices
- Inherence – what the user is, biometrics
- Location – where the user is, physical access to a device or network-based localization for low-grade security
- Behavior – mostly fuzzy, ML-based; usable for detection of anomalous behavior
Knowledge 1/3

Consider passwords a low-grade security measure

- Picking and remembering long passwords is difficult (in practice, impossible)
- Passwords can be snooped
  - They must always be sent over secure connection, but the user’s device is still vulnerable
- Always verify passwords (like any other data) in the backend
- In some applications it is okay to generate random secure passwords and ask for the user to write them down and store securely
- You should support password managers

Good resource for password-based authentication is [OWASP, 2021a]
Knowledge 2/3

- Beware of "pseudo" passwords: SSN, credit card number, mobile phone number
- The password recovery mechanism must be as strong as the main authentication method
Security Questions Be Like
Beware of "pseudo" passwords: SSN, credit card number, mobile phone number

The password recovery mechanism must be as strong as the main authentication

Security questions are just weak passwords

How many people had pets with the most popular names?

Never store passwords – always hash (see [OWASP, 2021b])

Prevent brute forcing of passwords

Throttle the login screen, add delays for continuous unsuccessful events

Never allow offline authentication operations
Possession

- Historically, different keys, tokens, smart cards (including chip-and-pin bank cards)
  - One-time password sheets, code cards partially fall under this
- Nowadays, mobile phone is typically used
  - Either with SMS check or with special apps
- The main requirement here is that it must be difficult to copy the token
Possession: Estonian Authentication Tokens

- ID-Card and other chip cards, Mobile-ID, Smart-ID
- Use the strongest possible authentication method conceivably available on a channel. Downgrade only if absolutely necessary
- Beware of verification of authentication results. There are several common pitfalls (i.e. [Cybernetica, RIA, 2021, 3.1.4.1])
Inherence

- Biometrics only works, if
  - you have secure enrollment mechanism and
  - you trust the device that captures biometric (camera, fingerprint reader) and
  - you trust that the person interacted with the device in a normal manner and
  - the channel from the device to your system is protected
- In short, remote biometric based login is usually snake oil
  - Biometrics can be used for local low-risk applications (and/or when the capture is under supervision)
- Beware of false acceptance and false rejection rates
- Many (most?) of commercial biometric devices can be fooled with various means
Location

- Similar to biometrics: almost impossible to verify online
- Used to be somewhat useful before ubiquitous connectivity
  - In the old times, if you could physically connect to network you were trusted
- Nowadays can be used as a weak check (e.g., for determining which content is legal in which country)
- Can be useful for audit, anomaly detection
Multi-Factor Authentication

- In practice, two-factor authentication
- Typically password + possession of a device (nowadays, typically a mobile phone)
  - Password sent to server in parallel with some code from device
  - Alternatively, password is used to activate device
- Smartcards, tokens used for high-security cases
- Many devices (computers, phones, SIM cards) have embedded trusted hardware that can run cryptographic protocol with your system
- MFA only works if the factors are independent!
Federated Authentication, Single-Sign-On

- “Outsourcing” of user- and method specific authentication flow to an external service
- Needs a federated identity system as a base
  - Simple with state-issued ID codes, much more complicated in the wild west of internet services
- Needs a protocol (like OpenID Connect) as a base
- Examples: Estonian TARA, Facebook, MS Azure AD
- Removes the need to maintain specific authentication factors in the business application
- **Trades it for the need to support a complicated protocol and federated identity**
Example: OpenID Connect

1: AuthN Request

2: AuthN & AuthZ

3: AuthN Response

Relying Party

End User

OpenID Provider

4: UserInfo Request

5: UserInfo Response
Examples of Authentication Protocols

- **OpenID Connect 1.0** – An authentication protocol for web applications, based on OAuth 2.0. Generally good, but very complicated.
- **SAML** – A older analogue of OpenID Connect, still widely in use, but contains significant flaws.
- **LDAP** – A directory service, which has an authentication mechanism that can be re-purposed as an application authentication protocol.
- **PAM** – Local authentication protocol of UNIX-ish operating system.
Choosing Authentication Protocols and Tools

- A very complicated topic, depends on the interplay between user population, business and regulatory requirements and technical platforms
- Many protocols have really small number of good implementations, for specific platforms, which may narrow options further
- Many ready-made tools come as a full package, even when only fraction of the functionality is needed, so the tool comes with a significant maintenance burden
- Still, using a ready-made and maintained tool should be preferred over developing one, but the overall setup should be carefully designed here
Access Control
Role-Based Access Control

- **Basic rules:**
  - Users are assigned roles
  - Users can take roles that are authorized for them
  - Users can take actions that are authorized for active role

- **Recommendation for implementing RBAC:**
  - Define atomic privileges, for example "can invoke operation X", in code check for presence of a privilege
  - Define roles by listing privileges allowed for the role (can be a config file)
  - Users are assigned roles
  - Only roles are granted access, never the individual users
Attribute Based Access Control 1/2

- Access rights are granted to users through the use of policies which combine attributes together.
  - Subject – the user
  - Action – action being attempted
  - Object – the object of the action
  - Contextual – environment (location, time, ...)
- Policies – statements that bring together attributes to express what can happen and what is not allowed.
  - Policies are basically logical statements describing the rules.
Atributed Based Access Control 2/2

1. View record #123
2. Can Alice view record #123?
3. Evaluate policies
4. Retrieve additional attributes
5. Permit, Alice can view record #123
6. View record #123
Delegation

- Authentication/identity level delegation – effective identity is different from validated identity
  - Typical examples: `sudo`, `su`
- Authorization/access control level delegation – Alice assigns set of roles/permissions (that they have) to Bob
- When trying to implement at least somewhat complex delegation mechanism, do yourself a favor and read SPKI RFC [Ylonen et al., 1999]
Questions?
References I


References II


Prentice-Hall.

Authentication cheat sheet.
References III

Password storage cheat sheet.
https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html.

SPKI Certificate Theory.
RFC 2693.