What does S & P Mean to You?

• Let us start with a few examples that are close to you...

  Password: ********

  website

  stackoverflow

  Siri

• As a user, an engineer, a future researcher, ...
A Story about Passwords

• What are these guidelines for?
  • Good/bad passwords?

• How are passwords cracked?
  • Brute-force attack
  • Dictionary attack
    • 30% of people have one of 10,000 top passwords [1]
    • Did you ever use first/last name, username, friends, edu/work, contacts, birthdays etc. as part of your password [2]?

• How do you protect your passwords?
  • Ask chrome to remember them
  • Password manager (e.g., LastPass)
  • Two-factor authentication
  • Secure Shell (SSH) with public/private key in remote login/Github/Gitlab

Almost no one lives without passwords.

[1] https://xato.net/10-000-top-passwords-6d6380716fe0
Another Story about Passwords

• This is a real story:
  • “Alice” was once a Masters student in our lab.
  • “Bob” was her boyfriend (now is her husband). He was a software developer, running a startup on building websites for small businesses.

  How do you manage passwords for your website?

  Oh, I just store them in the SQL DB. For every login, I compare the one that the user provides against what we have.

  What? Does it mean that you send and store passwords in PLAINTEXT? This is insecure!

  Well, it works and no one cares.

  That is a serious security issue! What if user account DB is hacked? You should at least implement Salted Password Hashing [*] in the right way...

  Murphy’ Law: Anything that can go wrong will go wrong.

  [*] https://crackstation.net/hashing-security.htm
My boss wants me to make the App secure, and asks me to implement RSA. But I don’t know how to do that in Java?

This is exactly what I want! let’s copy and paste the code … and it works!

Our App is secure, because we implemented RSA!

And some months later, there is a data breach …..
• A few facts from a recent study [*]:
  • 15.4% of all 1.3 million Android applications contains security-related code snippets from Stack Overflow.
  • 97.9% out of these contain at least one insecure code snippet.

• The design of a cryptosystem may be secure, but …
  • Did you randomly generate the keys for shared key cryptography?
  • Did you use strong keys for public key cryptography (e.g., RSA)?
  • Did you use a secure random number generator?
  • Did you use an outdated algorithm? …

• Insights on how/why it works >> “It works”

Siri and Alexa are Cool, But what if …

• Some **malicious voice commands** are embedded by Eve into normal voice commands, which allow Eve to control the device secretly?
Google Video Content Analysis is Cool, But What If …

- A **system** that samples the video and possibly run image processing on the frames.

- What if **the adversary** wants to manipulate the labels (e.g. to bypass the video censorship)?

- But the system seems to be **vulnerable**, because it only takes the first frame of each second.

- Possible **threat/attack**?

- Possible **defense**?

This Is What Might Happen To You

Before you take EE418

Every system just looks cool:
• I like this feature... It works great... This is amazing ...

After you take EE418

Every system still looks cool, but also **suspicious**
• Is it really secure? How do they ensure that?
• Is it vulnerable? What if a bad guy does something like this?
• (Many more questions in your mind)
Goals for Information Security

• Confidentiality/privacy
• Authentication
• Authorization
• Data Integrity
• Anonymity
• Certification
• Non-repudiation
Confidentiality/Privacy

- **Confidentiality** – Restricting access to the information to authorized entities.

Hi Bob, I want to tell you a secret, ...

Insecure/public channel

Eves:

$#&%#^&^(*$%^$%^$&^%$

Insecure/public channel
Authentication, Authorization

- **Identity Authentication** – Verifying someone’s identity
  - Are you really who you say you are?
- **Message Authentication** – Verifying message source/origin
- **Authorization** – Verifying that you have access to something or permission to do something.

Husky Card

Handwritten letter
Data Integrity

- **Data Integrity** – Ensuring that the data/message has not been altered by an unauthorized entity.

In crypto, the **hash** of a message is often used as the tape/seal.
Anonymity, Certification

- **Anonymity** – Concealing the identity of an entity
  - Example: Anonymous questionnaire (digital or handwritten?)

- **K-anonymity**: the identity of each user cannot be distinguished from the other \((k - 1)\) users.
  - Example: Location-based services. Anonymized databases.

- **Certification** – Endorsement of information by a trusted party
  - Example: Diploma, birth certificates ...

- **Non-repudiation** – Preventing the denial of previous actions
  - Example: Contract, signature
Approaches to Security

- **Prevention**
  - Stop an attack
  - Example: system patches, firewall, ...

- **Detection**
  - Detect an ongoing or past attack
  - Example: Antivirus software

- **Response**:  
  - Manage or contain attack in progress
  - Example: isolate in a sandbox, or delete it

- **Recover**:  
  - Graceful degradation
  - Example: restore from a backup
Summary

- S & P is so close to your daily life and work.
- The “Security Mindset” – a “new” way of thinking about systems
  - As a user, an engineer, a businessman, a future researcher ...
- **Basic cryptographic theory and practices**
  - To help you understand why things work the way they do
  - To help you implement security features in correctly
  - To help you perform security analysis, and design a secure system
- **Basic cryptographic techniques**
  - Cryptanalysis, shared key/public key cryptograph, hashing, etc.
- **Security goals and approaches**
Terminology (Backup Slides)

- **Communicating parties**: Alice and Bob
- **Eavesdropper** or adversary: Eve
- **Unicast**, multicast, broadcast
- **Comm. Channel**: physical medium, wired (e.g., copper wire, fiber) or wireless (e.g., radio, Wi-Fi)
- **Encryption**: create a ciphertext from plaintext using an encryption key $K$ and following an encryption rule $e_K$.
- **Decryption**: obtain the plaintext from ciphertext using an decryption key $K$ and following an decryption rule $d_K$. 
Formal Description of a Cryptosystem

- A cryptosystem can be represented in terms of \((\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})\)
  - \(\mathcal{P}\): the set of possible plaintexts
  - \(\mathcal{C}\): the set of possible ciphertexts
  - \(\mathcal{K}\): the set of possible keys
  - \(\mathcal{E}\): the set of encryption rules
  - \(\mathcal{D}\): the set of decryption rules

- Let \(x \in \mathcal{P}, K \in \mathcal{K}\). Encryption is a rule \(e_K \in \mathcal{E}\), and decryption is a rule \(d_K \in \mathcal{D}\). We require

\[
d_K(e_K(x)) = x
\]
Example

- A wants to send a message to B over an insecure channel.

- What does A do?
  - Message: $x = x_1x_2 \ldots x_n$, where $n \geq 1$, and $x_i \in \mathcal{P}, \forall i$
  - Encrypt $x$ with key $K$: for each $x_i$, generate a ciphertext $Y_i = e_K(x_i) \in \mathcal{C}$
  - Send the encrypted message $Y = Y_1Y_2 \ldots Y_n$ over the channel

- What does B do?
  - B knows the key $K$ and the decryption algorithm $d_K$
  - Decrypt $Y$: for each $Y_i$, find the plaintext $x_i = d_K(Y_i)$
  - Reconstruct the message: $x = x_1x_2 \ldots x_n$