Design Principle: Security Through Obscurity

Relying on secrecy or concealing the details of a system or its components to provide security

- If an attacker does not know how a system works, they are less likely to compromise it.
- This is often regarded as insufficient and unreliable as the sole basis for security. Attackers may reverse-engineer or uncover hidden details. We cannot solely rely on its obscurity to keep attackers away.

Examples:

- A company hides sensitive files behind obscure URLs without implementing proper authentication. Attacker could discover the URL through guessing, web crawling or server logs.
- A software developer uses code obfuscation to hide the details of source code and potential vulnerabilities. Skilled attacker can deobfuscate or analyze the binary to discover the vulnerabilities.

Design Principle: Kerckhoffs's Principle and Shannon's Maxim

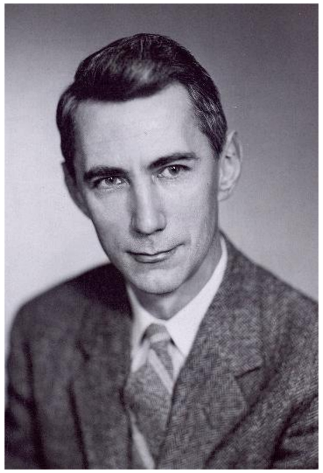
Claude Shannon: "the enemy knows the system"

- The security of a system should not depend on the secrecy of its design or algorithms.
- It is always necessary to assume that the attacker knows every detail about the system you are designing, including algorithms, hardware, defenses, etc.
- This makes your system resilient even if the design or implementation becomes public knowledge

Examples:

- Cryptography: the secrecy of the cryptographic key is the only thing that ensures security. If the key is kept confidential, the system remains secure

Auguste Kerckhoffs
Dutch linguist and
cryptographer


Claude Shannon
American mathematician and
cryptographer
Father of information theory