一、引入.

1、什么是 Security?

Achieving some goals in the presence of adversaries

Physical World	Cyber World
• Goals	• Goals
• Personal safety	• Dependability
• Property safety	• Data safety
• …	• …
• Adversaries	• Adversaries
• Thieves	• Crackers
• Criminals	• Cyber attackers
• …	• …

2、为什么 Security 越来越重要?

① 计算机的使用愈发广泛

② 网络的"双刃剑"

其中最核心的一个议题就是 Information Flow Security

二、Information Flow Security

1、什么是 Information Flow?

- Information flow: if the information in variable x is transferred to variable y, then there is information flow $x \rightarrow y$

- Examples


```
y = x;
```

```
a = x;
b.f = a;
y = b.f;
```

*Dorothy E. Denning and Peter J. Denning, "*Certification of Programs for Secure Information Flow*". CACM 1977.

信息流是一个比数据流更为普适的概念，它可以显式地基于赋值传递，也能够以各种意想不到的方式隐式传递。

2. Information Flow Security 是在干什么？

信息可以在状态机内顺着各种信道流动，其中我们主观上构建的旨在传递信息的称为显式信道，而各种 unexpected 的通道是隐式信道。I-F Security 就是为了避免系统与不信任的第三方之间的数据流。

17

Ps: Access Control Vs. I-F Security.

两种避免系统与恶意第三方之间信息流的两种

方案。前者使用接入控制，后者倾向于追踪信息流动的路径，判断其是否有可能流入恶意第三方。

- Access control (a standard way to protect sensitive data)
  - Checks if the program has the rights/permissions to access certain information
  - Concerns how information is **accessed**

What happens after that?

- Information flow security (end-to-end)
  - Tracks how information flows through the program to make sure that the program handles the information securely
  - Concerns how information is **propagated**

"*A practical system needs both **access** and **flow control** to satisfy all security requirements.*"
— D. Denning, 1976

互补而技术

抽象地讲，IF Security

①将变量分成不同的安全等级

- The most basic model is two-level policy, i.e., a variable is classified into one of two security levels:
  1. H, meaning *high* security, secret information
  2. L, meaning *low* security, public observable information

```
h = getPassword(); // h is high security
broadcast(l);      // l is low security
```

- Security levels can be modeled as *lattice**
  - L ≤ H

*Dorothy E. Denning, "*A Lattice Model of Secure Information Flow*". CACM 1976.

②在等级格上增加约束，限制不同等级变量之间的信息流动，也即 I-F Policy.

3. Confidentiality Vs. Integrity.

对于不同的应用场景，我们实制订的 Policy 自然也是不同的。

(1) 机密性.

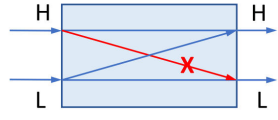
防止高密级的信息发生泄露，也即防止 H → 恶意第三方的信息流。

★ 不干涉原则

- Noninterference policy*
  - Requires the information of high variables have no effect on (i.e., should not interfere with) the information of low variables
  - Intuitively, you should not be able to conclude anything about high information by observing low variables

*J. A. Goguen and J. Meseguer, "*Security policies and security models*". S&P 1982.

- ✓ $x_H = y_H$
- ✓ $x_L = y_L$
- ✗ $x_L = y_H$
- ✓ $x_H = y_L$
- ✗ $x_L = y_L + z_H$



Ensures that information flows only upwards in the lattice

(2) 完整性

避免恶意第三方 → H 的信息流，防止其破坏 H 中信息的准确性，完整性、一致性。

- Prevent untrusted information from corrupting (trusted) critical information[1]

```
x = readInput(); // untrusted
cmd = "..." + x;
execute(cmd); // critical (trusted)
```

- Injection errors (#1 cause of vulnerabilities in 2013-2019[2])
  - Command injection
  - SQL injection
  - XSS attacks
  - ...

对对比.

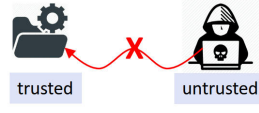
两者实际上是对偶的，其问题结构一致，因此一般可以利用同样的技术手段解决。

Confidentiality ⟷ **Duals** ⟷ **Integrity**

- Security classification
  - Secret (**H**igh secret) **H**
  - Public (**L**ow secret) **L**

- Read protection

- Security classification
  - Trusted (**H**igh integrity) **L**
  - Untrusted (**L**ow integrity) **H**



- Write protection

4. 详解信息流

由于IT-Security本质上就是在做信息流的追踪、控制，因此再详细研究一下信息流是十分有必要的。

(1) 显式流 (explicit flow)

- $x_H = y_H$
- $x_L = y_H$
- $x_L = y_L + z_H$

We have seen how information flows through direct copying.
This is called explicit flow.

(2) 隐式流

- Mechanisms for signalling information through a computing system are known as ***channels***.
- Channels that exploit a mechanism whose primary purpose is not information transfer are called ***covert channels****.

- Implicit flows `if (secret_H < 0) p_L = 1; else p_L = 0;`
  signal information through the control structure of a program
- Termination channels `while (secret_H < 0) { … };`
  signal information through the (non)termination of a computation
- Timing channels `if (secret_H < 0) for (…) { … };`
  signal information through the computation time
- Exceptions `if (secret_H < 0) throw new Exception("…");`
  signal information through the exceptions
- …

*Butler W. Lampson, "*A Note on the Confinement Problem*". CACM 1973.

H <联系> 可观测结果

⇒ 产生信息泄露

但是通过隐式信道泄露的信息量一般是很小的，所以我们主要追踪的对象是显式信道。

```
int secret_H = getSecret();
int publik_L = secret_H;
```

Explicit flow:
transmits 32 bits of information

```
int secret_H = getSecret();
if (secret_H % 2 == 0)
    publik_L = 1;
else
    publik_L = 0;
```

Implicit flow:
transmits 1 bit of information