Computer Security (COM-301)
Adversarial thinking
Attacks and defenses

Carmela Troncoso

SPRING Lab

carmela.troncoso@epfl.ch

Some slides/ideas adapted from: Emiliano de Cristofaro, Gianluca Stringhini, George Danezis

Structure of the lecture

- Why studying attacks is so important?
- How are attacks developed?
  - Adversarial thinking process
  - Examples on real world systems
- Which attacks should you worry about?
  - Reasoning process: what can go wrong? what not to do?
  - Example attacks on software

Computer Security (COM-301)
Adversarial thinking
Reasoning as an adversary

Carmela Troncoso
SPRING Lab
carmela.troncoso@epfl.ch

Some slides/ideas adapted from: Emiliano de Cristofaro, Gianluca Stringhini, George Danezis

Why do we study attacks?

Deeper Understanding of Defense

Very good attackers make very good defenders (and vice versa – find many attacks)

Mediocre attackers, make extremely poor defenders (find some attacks…)

Job opportunity: Penetration testing (pentesting) is a **major** industry

Try to bypass controls to establish the security quality of a system

Nowadays also privacy!
Companies need to work with data, and need to make sure that no inferences can be made. They require knowledge to test how well the algorithms they deploy sanitize their data

Why do we study attacks?

Deeper Understanding of Defense

Very good attackers make very good defenders (and vice versa – find many attacks)

Mediocre attackers, make extremely poor defenders (find some attacks…)

Job opportunity: Penetration testing (pentesting) is a **major** industry

Try to bypass controls to establish the security quality of a system

Does lack of found attacks guarantee that the system is secure?

No! we can never be sure we have explored the complete attack space

Related concepts: fail safe principle, sanitization

Why do we study attacks?

Deeper Understanding of Defense

Very good attackers make very good defenders (and vice versa – find many attacks)

Mediocre attackers, make extremely poor defenders (find some attacks…)

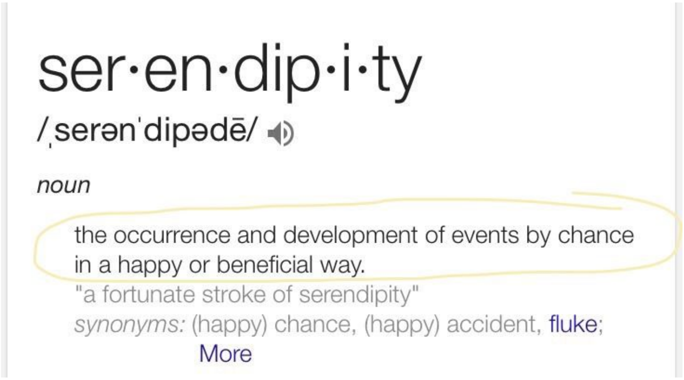
Job opportunity: Penetration testing (pentesting) is a **major** industry

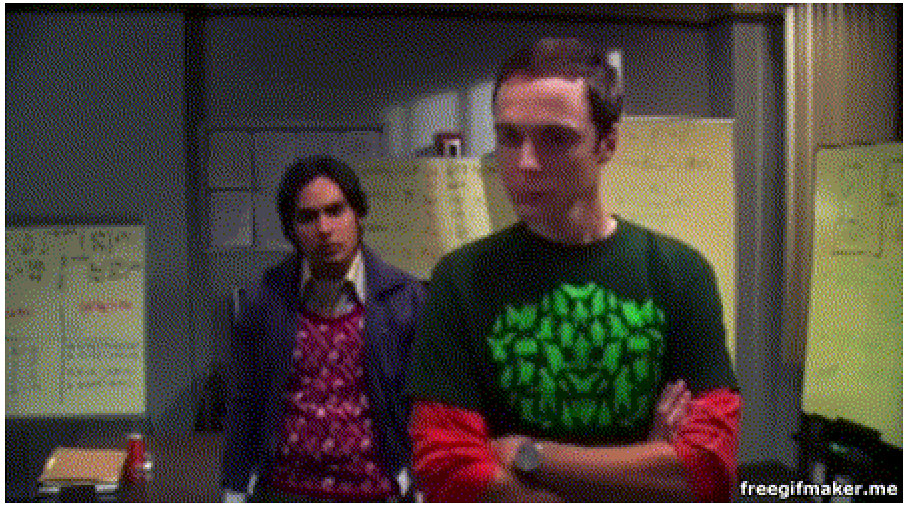
Try to bypass controls to establish the security quality of a system

Remember you cannot freely hack around

Ethics, law, and regulations

How are attacks developed?





The security engineering process

1. **Define a security policy (principals, assets, properties) and a threat model.**

2. **Define security mechanisms that support the policy given the threat model.**

3. **Build an implementation that supports / embodies the mechanisms.**

The attack engineering process

“inverse” approach – exploits flaws in the security engineering process

1. **Define a security policy (principals, assets, properties) and a threat model.**

   Adversary can exploit

   Misidentified principals, assets, or properties

   Capabilities beyond what is considered in threat model

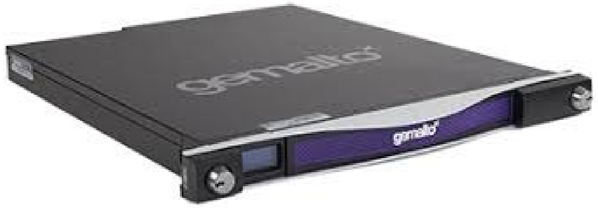
   (more access or more computational/algorithmic capabilities)

2. **Define security mechanisms that support the policy given the threat model.**

3. **Build an implementation that supports / embodies the mechanisms.**

The attack engineering process

Exploiting misidentified assets in the security policy



EXAMPLE 1 – EXTRACTING KEYS FROM HARDWARE SECURE MODULES (HSMS)

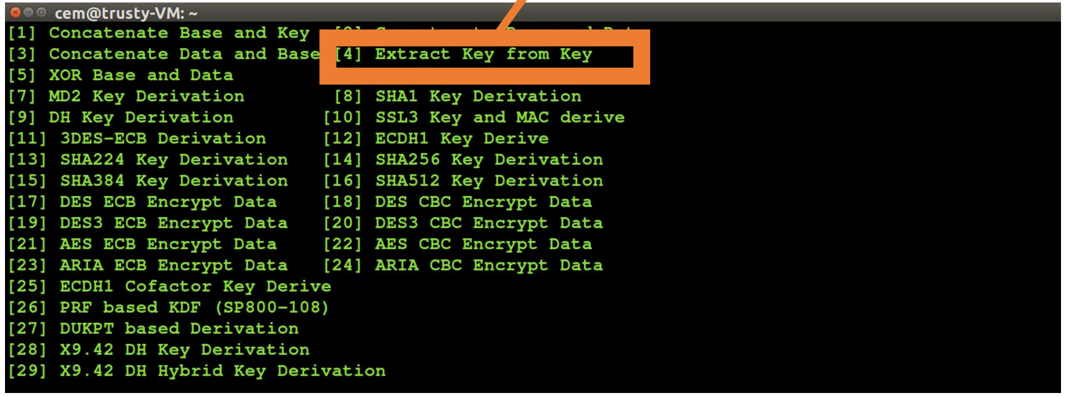
HSMs implement PKCS#11 standard for interoperability

API to create a new key from the secret key:

Given bits_length and offset, it uses bits_length of the secret key from position offset

How would you exploit this function?

Create a new key using a substring of an existing key.



The attack engineering process

Exploiting misidentified assets in the security policy

PKCS#11 considers the full key an asset to protect, but not bytes of the key

Create a new key using a substring of an existing key.

EXAMPLE 1 – EXTRACTING KEYS FROM HARDWARE SECURE MODULES (HSMS)

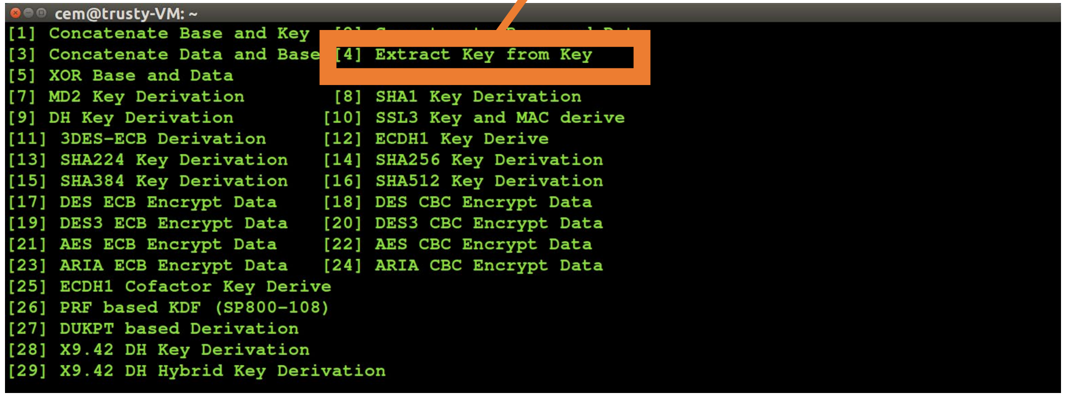
Assume a strong key exists in the HSM

Ask HSM to derive a new key of length 1 byte at offset 0

Use new key to do an operation, say HMAC on a known input (allowed by the HSM)

Brute force the key (input known, output known, key only 1 byte)

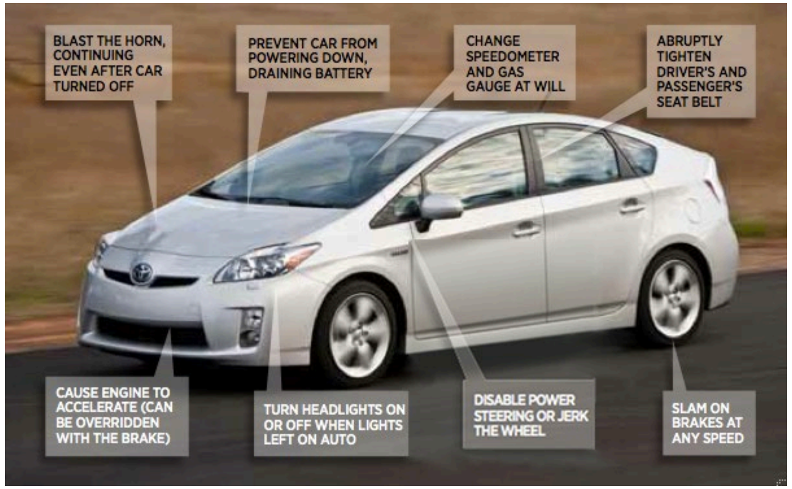
Repeat with keys at different offsets ⟶ Full key recovery!



The attack engineering process

Exploiting unforeseen access capabilities

In both cases the adversary had remote access to functionality that was not foreseen by the threat model



EXAMPLE 2 – FROM CABLE TO THE AIR

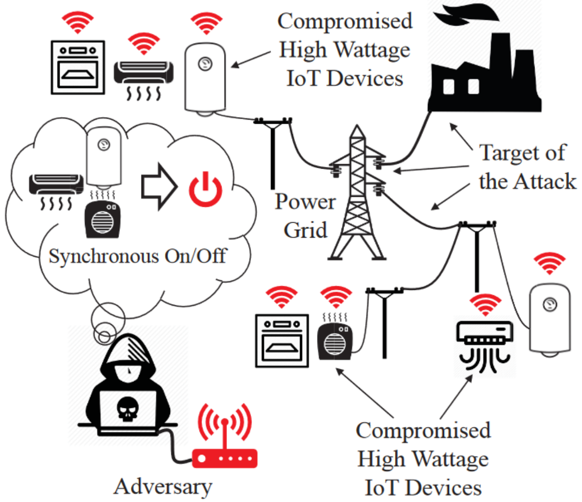
Engine Control Units (ECU) control the vehicle

ECU connected to GSM/WiFi give a remote adversary access to the CAN bus and all the (safety) functions of the vehicle

EXAMPLE 3 – IOT DEVICES ARE A WEAK LINK

IoT weakly protected devices connected to internet

MadIoT - manipulation of demand via IoT **(Princeton U.) – hackers can compromise the Smart Grid with ~100K devices**



https://www.forbes.com/sites/andygreenberg/2013/07/24/hackers-reveal-nasty-new-car-attacks-with-me-behind-the-wheel-video/#4b536af4228c
https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/
https://www.ft.com/content/2c17ff5e-4f02-11e8-ac41-759eee1efb74

The attack engineering process

Exploiting unforeseen capabilities

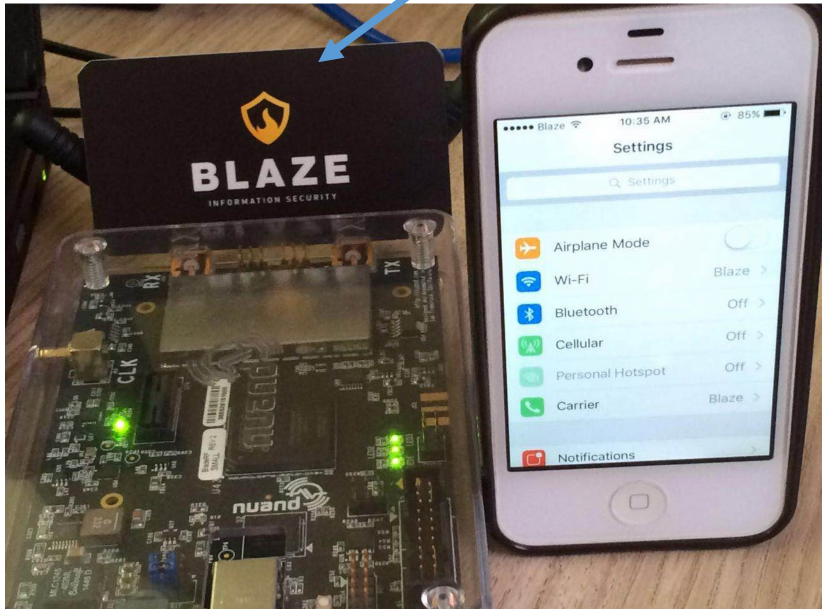
EXAMPLE 3 – UNILATERAL USER AUTHENTICATION IN GSM

When GSM was designed antennas (Base Transceiver Stations - BTS) were difficult to implement and expensive to build.

Thus, operators decided that the network did not need to authenticate!

Nowadays, commodity hardware can be used to fake a base station and perform a man in the middle (eavesdrop, impersonate,…)!

Fake BTS!



https://wildfire.blazeinfosec.com/practical-attacks-against-gsm-networks-part-1/

The attack engineering process

Exploiting unforeseen computational/algorithmic capabilities

EXAMPLE 4 – THE MACHINE LEARNING REVOLUTION

The power of inference at your fingertips!

Apparently irrelevant information becomes critical for the security of the system

Learn to break better and faster!

Machine learning eases attacks, as it simplifies their implementation through substituting complex modeling tasks by data collection

Jul 6, 2017, 10:10am

Help! Hackers Stole My Password Just By Listening To Me Type On Skype!

Thomas Brewster Forbes Staff
Security
I cover crime, privacy and security in digital and physical forms.

For many, everyday life involves sitting in front of a computer typing endless emails, presentation documents and reports. Then there's the frequent typing of passwords just to get access to those files. But

The Real First Class? Inferring Confidential Corporate Mergers and Government Relations from Air Traffic Communication

Martin Strohmeier*, Matthew Smith*, Vincent Lenders†, Ivan Martinovic*

*University of Oxford, UK †armasuisse, Switzerland

Abstract—**This paper exploits publicly available aircraft meta data in conjunction with unfiltered air traffic communication gathered from a global collaborative sensor network to study the privacy impact of large-scale aircraft tracking on governments and public corporations.**

First, we use movement data of 542 verified aircraft used by 113 different governments to identify events and relationships ...ral clustering ...blic meetings

through ADS-B tracking [11] or revelations on the personal use of corporate aircraft by top management [18, 7].

To go beyond such anecdotes and provide a more complete picture, we analyze the impact that large-scale and long-term collection of aircraft communication data has on the privacy of aviation users. The difficulty of obtaining flight movement data has considerably decreased with the advent of affordable software-defined radios (SDRs), which make the reception of ADS-B messages (and thus the po-

Using deep learning to break a Captcha system

Using Torch code to break simplecaptcha with 92% accuracy

Captcha is used as a common tactic to stop bots from entering a website. Any visitor to a website is presented with a text image containing some letters which can be read by a human and not by automated bots. They are quite frequently used to stop automated password hacking or automated login to websites etc. The following is taken from the wikipedia page[1] of captcha. As captchas are usually used to deter software programs they can be usually very hard to read and human accuracy can be around 93% [2]. It also takes something like 10 secs to read a captcha. As can be seen this takes quite a toll on the user experience.

Breaking Captchas

There are a few approaches to defeating CAPTCHAs: using cheap human labor to

The attack engineering process

Exploiting unforeseen computational/algorithmic capabilities

Jul 6, 2017, 10:10am

EXAMPLE 3

The powe

Learn to b

THE MACHINE LEARNING REVOLUTION: ALSO WORKS FOR THE GOOD GUYS!!

Improved malware detection

Predicting zero days (unknown vulnerabilities)

Identifying vulnerable devices

Automated log analysis

ential Corporate Mergers and Government
Traffic Communication

h*, Vincent Lenders†, Ivan Martinovic*

†armasuisse, Switzerland

through ADS-B tracking [11] or revelations on the personal use of corporate aircraft by top management [18, 7].
To go beyond such anecdotes and provide a more complete picture, we analyze the impact that large-scale and long-term collection of aircraft communication data has on the privacy of aviation users. The difficulty of obtaining flight movement data has considerably decreased with the advent of affordable software-defined radios (SDRs), which make the reception of ADS-B messages (and thus the po-

a human and not by automated bots. They are quite frequently used to stop automated password hacking or automated login to websites etc. The following is taken from the wikipedia page[1] of captcha. As captchas are usually used to deter software programs they can be usually very hard to read and human accuracy can be around 93% [2]. It also takes something like 10 secs to read a captcha. As can be seen this takes quite a toll on the user experience.

Breaking Captchas

There are a few approaches to defeating CAPTCHAs: using cheap human labor to

The attack engineering process

“inverse” approach – exploits flaws in the security engineering process

1. **Define a security policy (principals, assets, properties) and a threat model.**

   Adversary can exploit

   Misidentified principals, assets, or properties

   Capabilities beyond what is considered in threat model

   (access or computational/algorithmic)

2. **Define security mechanisms that support the policy given the threat model.**

   Adversary can exploit

   Design weaknesses/flaws in the security mechanisms

3. **Build an implementation that supports / embodies the mechanisms.**

The attack engineering process

Exploiting security mechanisms design weaknesses

In both cases the algorithms were secret, but researchers reverse engineered them. Once the algorithms were known researchers identified vulnerabilities that allowed them to decrypt and read messages, and even recover the key.

EXAMPLE 1 – WEAK CRYPTOGRAPHIC PRIMITIVES

Tesla – Key Fob algorithm to start the car allows to recover key in seconds (with pre-computation)

**GSM – A5/1 and A5/2 weak allow ciphertext only attacks
Can be real time by FPGA parallel computation!**

Security by obscurity is a bad idea <- Open design principle!

https://web.archive.org/web/20090821163913/http://reflextor.com/trac/a51/wiki
https://www.securityweek.com/hackers-can-clone-tesla-key-fobs-seconds/
https://motherboard.vice.com/amp/en_us/article/43ee8m/watch-hackers-steal-a-tesla-model-s-key-fob-hack

The attack engineering process

“inverse” approach – exploits flaws in the security engineering process

1. **Define a security policy (principals, assets, properties) and a threat model.**

   Adversary can exploit

   Misidentified principals, assets, or properties

   Capabilities beyond what is considered in threat model

   (access or computational/algorithmic)

2. **Define security mechanisms that support the policy given the threat model.**

   Adversary can exploit

   Design weaknesses/flaws in the security mechanisms

3. **Build an implementation that supports / embodies the mechanisms.**

   Adversary can exploit

   Implementation or operation problems that allow you to subvert the mechanisms

The attack engineering process

Exploiting bad operation decisions to subvert security mechanisms

EXAMPLE 1 – WEP BAD USE OF RC4

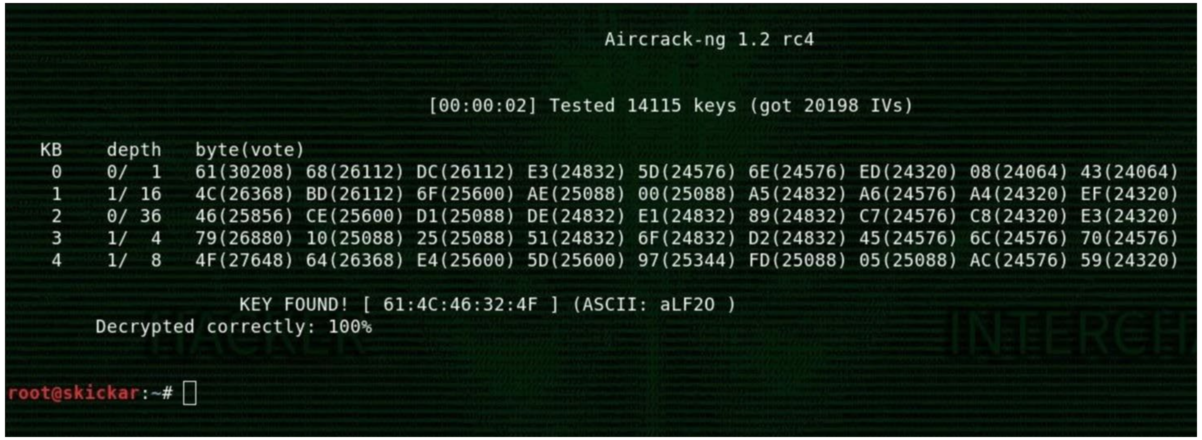
WEP uses RC4, a secure stream cipher when the IV is random.

When the IV is repeated, the stream produced by RC4 that is XORed with messages is repeated. This effectively is a repeated One Time Pad, and thus allows to recover messages. Because of some particularities of how RC4 is constructed, one can even recover the secret key.

In WEP the IV is defined to have 24 bit. The implementation uses this 24 bits in such a way that the IV is repeated every 5000 / 6000 frames!

Adversary can accelerate the attack by spoofing MAC addresses to ask for more frames

Can be also seen as the WEP protocol is a flawed design



https://null-byte.wonderhowto.com/how-to/hack-wi-fi-cracking-wep-passwords-with-aircrack-ng-0147340/
https://asecuritysite.com/encryption/rc4_wep

The attack engineering process

Exploiting implementation flaws to subvert security mechanisms

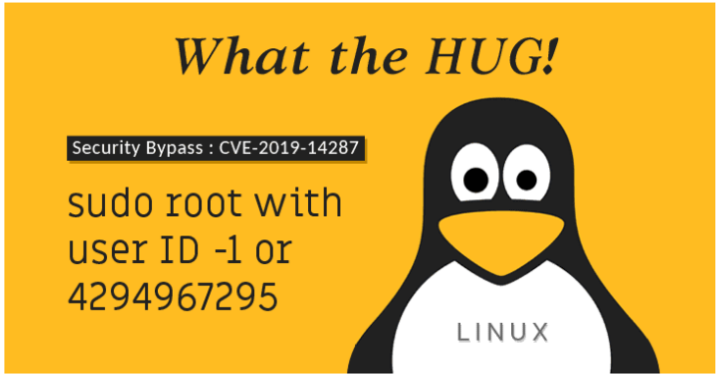
Example 2 – Bugs, bugs and more bugs

Programmers make mistakes:

- They forget checks, or check the wrong things
- They do not sanitize, or do not sanitize correctly
- They forget to protect what needs to be protected
- They get confused about origin or reliability of data / variables (Ambient authority & confused deputy)

Sudo Flaw Lets Linux Users Run Commands As Root Even When They're Restricted

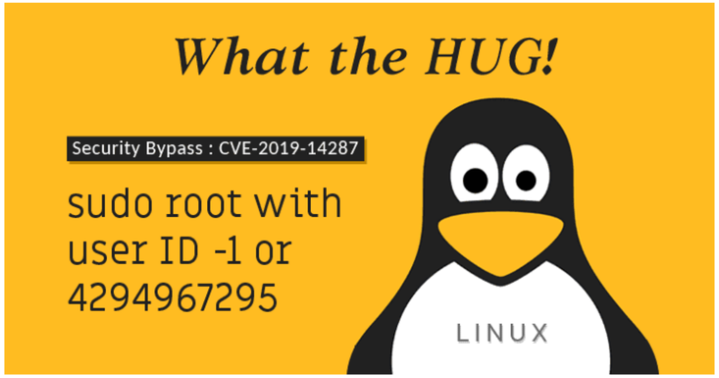
October 14, 2019 Mohit Kumar



Why does this work?

Sudo Flaw Lets Linux Users Run Commands As Root Even When They're Restricted

October 14, 2019 Mohit Kumar



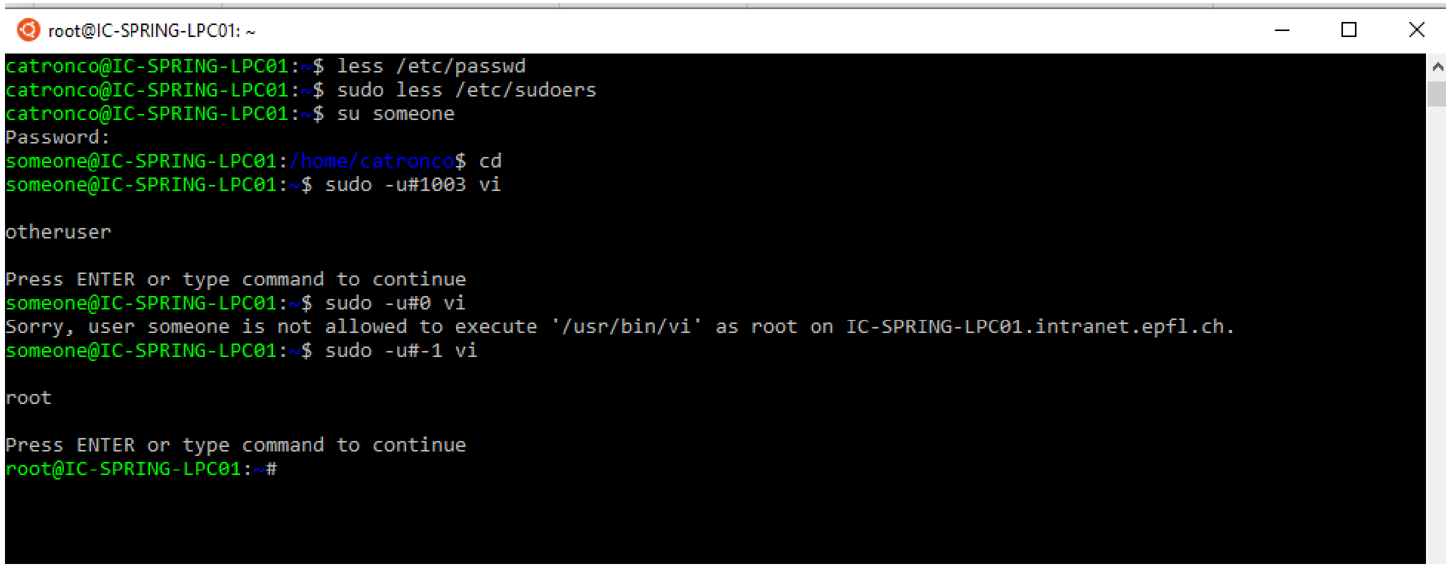
Sudo program uses two routines, one of them does the change in UID (Remember UID is what determines the permissions of the program).

That routine understands "-1" as "do nothing".

Because the routine is called inside sudo, which is being executed as root (UID = 0), then the program comes out without changing, and stays with the same UID.

Bright side, only exploitable under certain configurations in which users can execute sudo on potential dangerous programs for some users except for root:

```
someone ALL=(ALL, !root) /usr/bin/vi
```



```
root@IC-SPRING-LPC01: ~
catronco@IC-SPRING-LPC01:~$ less /etc/passwd
catronco@IC-SPRING-LPC01:~$ sudo less /etc/sudoers
catronco@IC-SPRING-LPC01:~$ su someone
Password:
someone@IC-SPRING-LPC01:/home/catronco$ cd
someone@IC-SPRING-LPC01:~$ sudo -u#1003 vi

otheruser

Press ENTER or type command to continue
someone@IC-SPRING-LPC01:~$ sudo -u#0 vi
Sorry, user someone is not allowed to execute '/usr/bin/vi' as root on IC-SPRING-LPC01.intranet.epfl.ch.
someone@IC-SPRING-LPC01:~$ sudo -u#-1 vi

root

Press ENTER or type command to continue
root@IC-SPRING-LPC01:~#
```

Computer Security (COM-301)
Adversarial thinking
Reasoning as a defender - I

Carmela Troncoso
SPRING Lab
carmela.troncoso@epfl.ch

Some slides/ideas adapted from: Emiliano de Cristofaro, Gianluca Stringhini, George Danezis

Reasoning about attacks
Threat modelling methodologies

IDEA: help security engineers reason about threats to a system - **“What can go wrong?"**

Threat modelling

Process to identify potential threats and unprotected resources with the goal of prioritizing problems to implement security mechanisms.

Systematic analysis:

What are the most relevant threats?

What kind of problems can threats result on?

Where should I put more effort to protect?

Reasoning about attacks
Threat modelling methodologies

IDEA: help security engineers reason about threats to a system - **“What can go wrong?"**

Attack trees

The attack goal is the root, and the ways to achieve this goal are represented by the branches. The leafs are the weak resources.

STRIDE

Identify system entities, events, and the boundaries of the system.

Reason about threats enumerating the type of threats that can be embodied by the adversary

P.A.S.T.A.

Start from business goals, processes, and use cases.

Find threats within business model, assess impact, and prioritize based on risk

Many more!

Reasoning about attacks
STRIDE (by Microsoft)

Model the target system, with entities, assets, and flows. Then reason about:

Threat	Property threatened	Example
Spoofing	Authenticity	A member of the council of Ricks convinces Morty that he is the real Rick
Tampering	Integrity	The bad minion modifies the plan message send by Gru to our favorite minion Bob
Repudiation	Non-repudiability	Summer denies having told Morty that Rick was waiting for him
Information disclosure	Confidentiality	Summer learns about the secret plans of Rick and Morty
Denial of Service	Availability	The minions flood Dr. Nefario's lab with bananas and he cannot receive the latest weapons
Elevation of Privilege	Authorization	Bob the minion gains access to the system with Gru's credentials

Reasoning about attacks
Brainstorming using cards

http://securitycards.cs.washington.edu/index.html