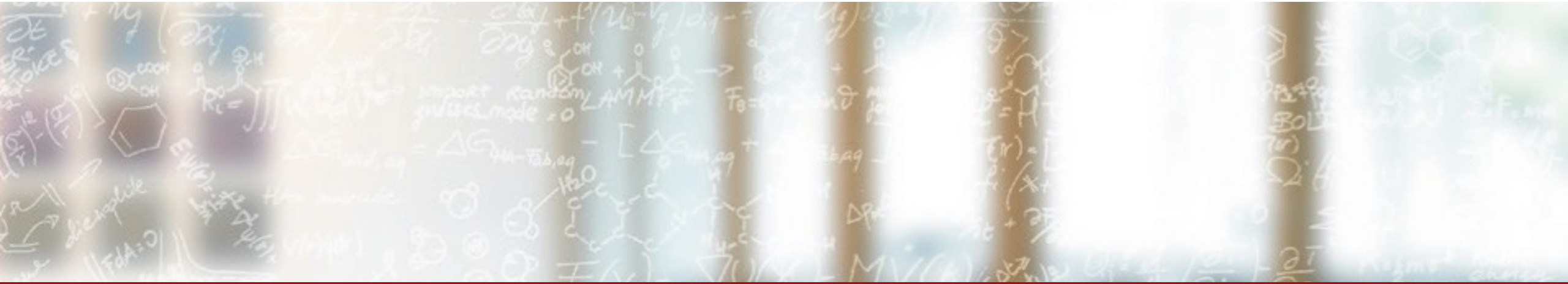




**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH**zürich



# Experiments with eBPF for Security

HPC-ch Forum on HPC Security

Victor Holanda Rusu, CSCS

May (the) 4th (be with you), 2023



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Quick *Census*

---



How many of you have SELinux enabled in your HPC cluster?

SELinux with confined users (e.g. user\_t)?

SELinux with confined users and categories (MCS)?

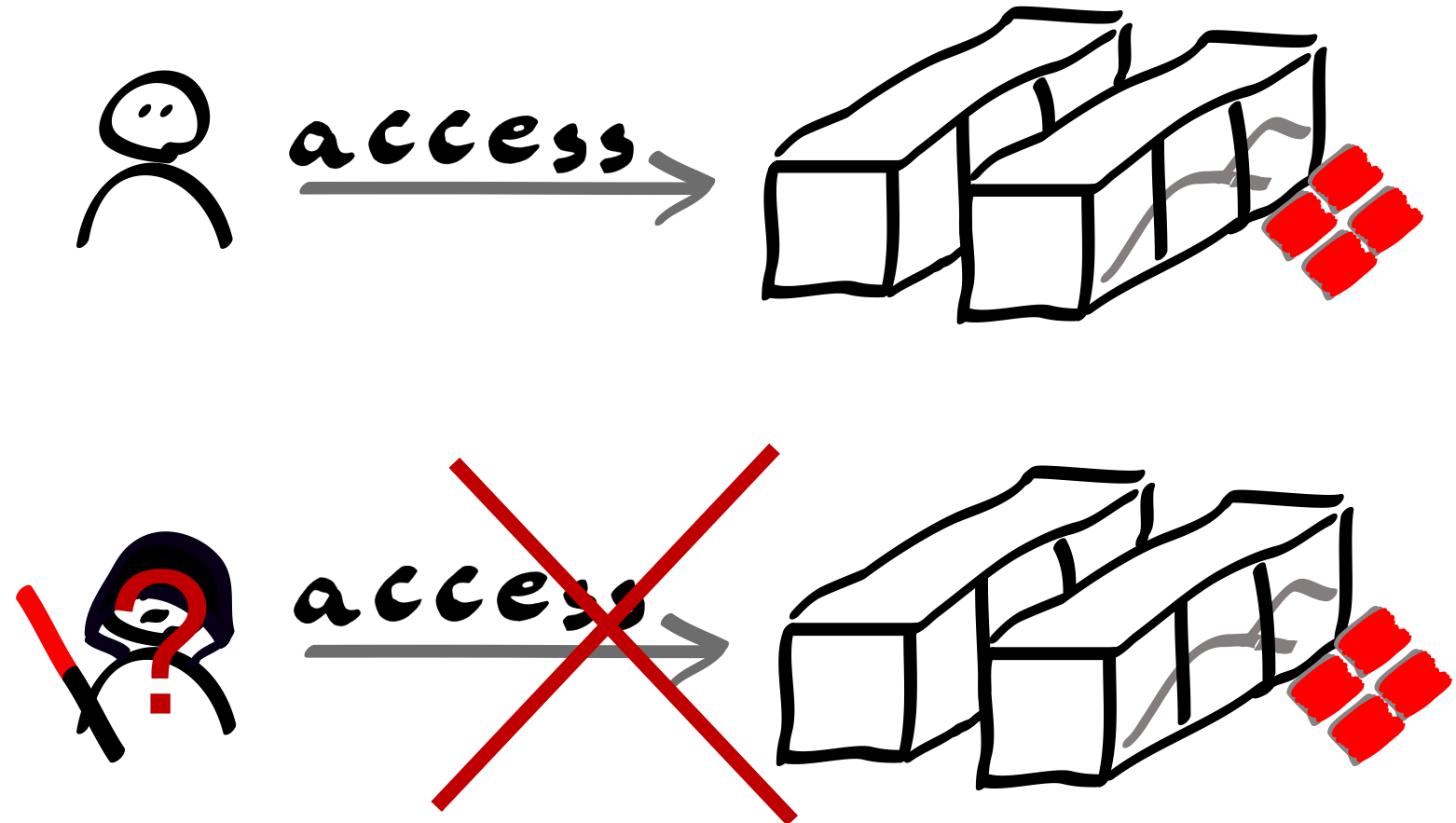
AppArmor?

AppArmor using Role-Based Access Control (RBAC)?

# Intro to the problem

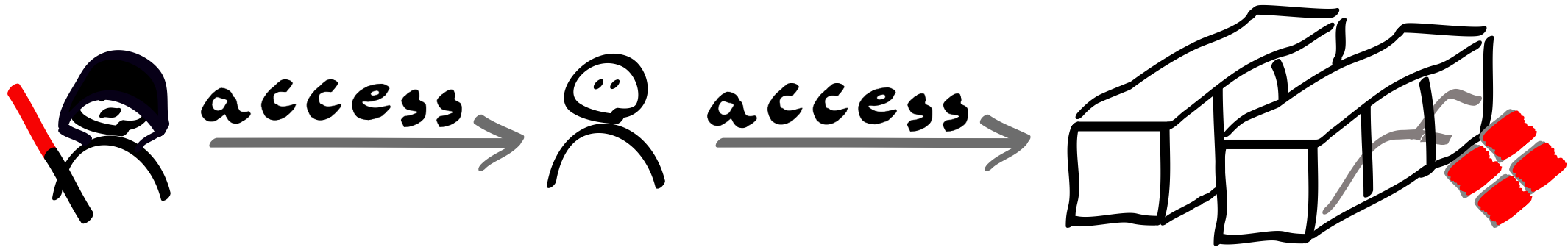
## What are we trying to solve?

- Strong password policies
- Multifactor authentication
- User Behaviour Analytics



# Intro to the problem

Is it game over?



At this point we are at damage control level

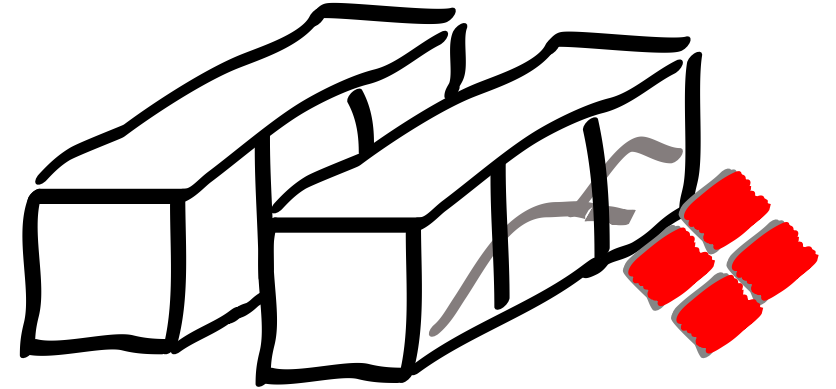
# When the bad guys are in...

## What can they do?

- Data exfiltration
- Lateral movement
- Cryptojacking
- Ransomware
- Denial of Service
- Privilege escalation



*access* →

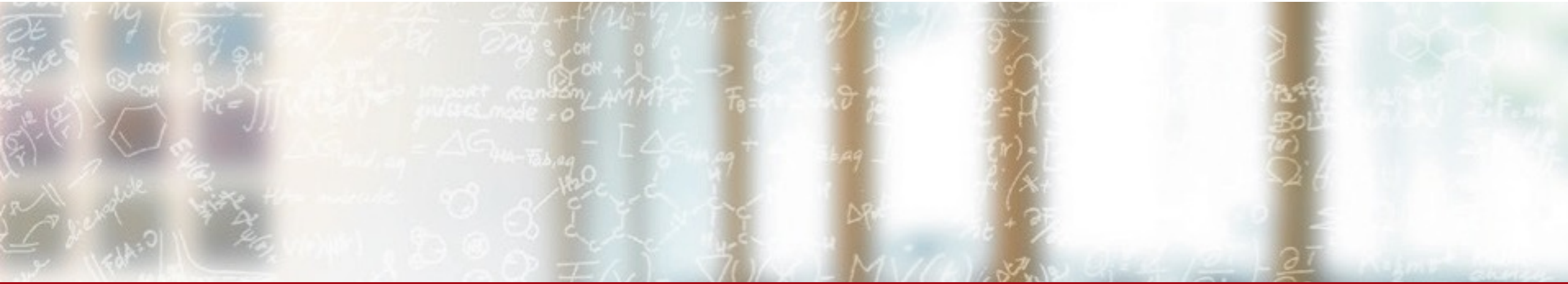




**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH**zürich



# Experiments with eBPF for Security **in the context of privilege escalation**

HPC-ch Forum on HPC Security

Victor Holanda Rusu, CSCS

May (the) 4th (be with you), 2023



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Brief intro to eBPF

---



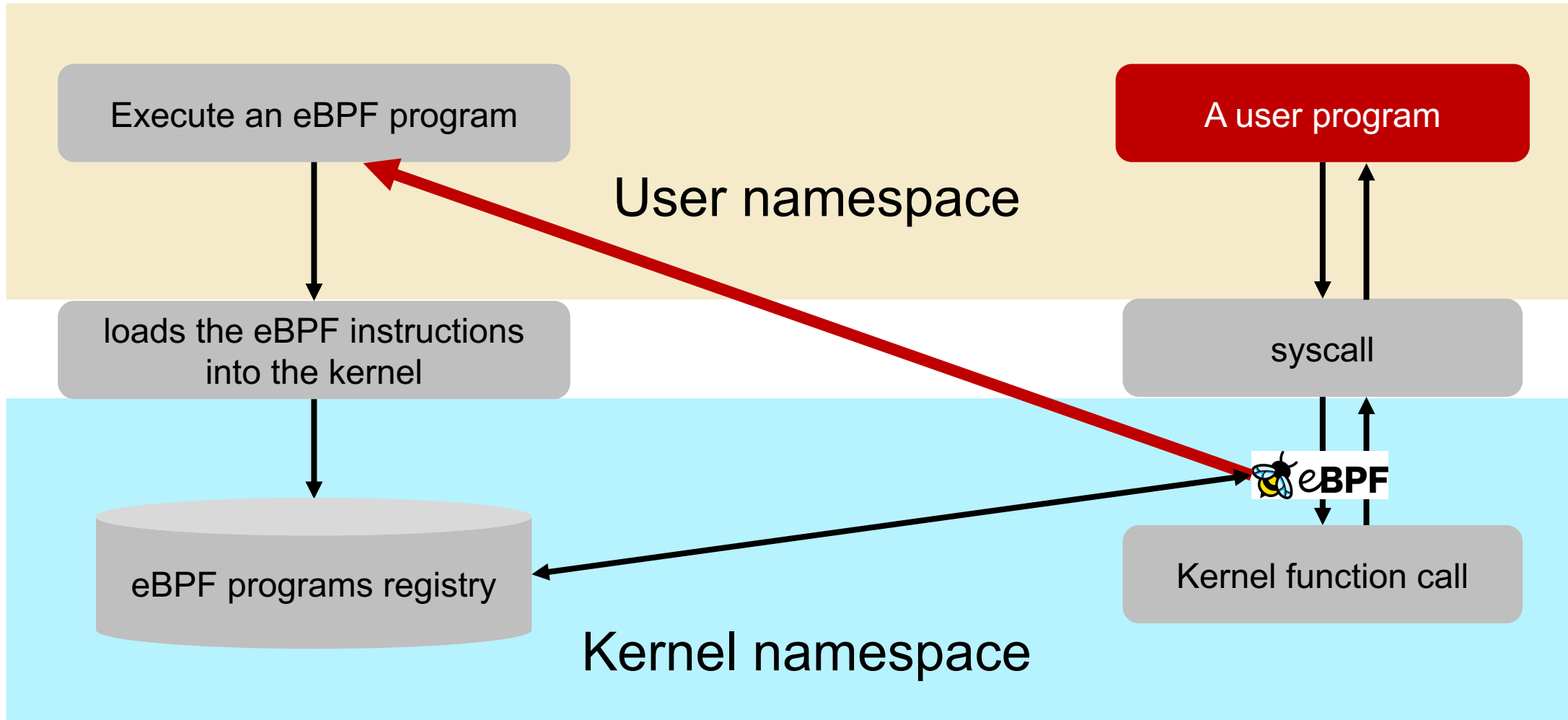
# What's eBPF?

A brief intro? Nope! It is more like an ultra fast, supersonic one!

- eBPF is a Linux kernel technology
- eBPF allows to run **sandboxed programs in the Linux kernel context**
  - Programs are executed within the Kernel's Virtual Machine
- It aims to **safely and efficiently extend the kernel capabilities** without loading kernel modules or kernel source code changes
  - The programs are **restricted to a subset of operations** (e.g. no memory allocation, no sleeping)
- **It is event-driven** and works when the kernel passes certain hook points
- It can be used to **trace applications** from the kernel point of view
  - Perfect for tracing **containerised applications** (share the kernel with the host system)
- There are several development toolchains
  - libbpf (C/C++) – CORE (**Compile Once Run Everywhere**) approach
  - bcc (python)
  - Bpftrace (standalone language similar to awk)
  - ebpf Go lib

# What's eBPF?

## How an eBPF program works?



# How to use eBPF for security?

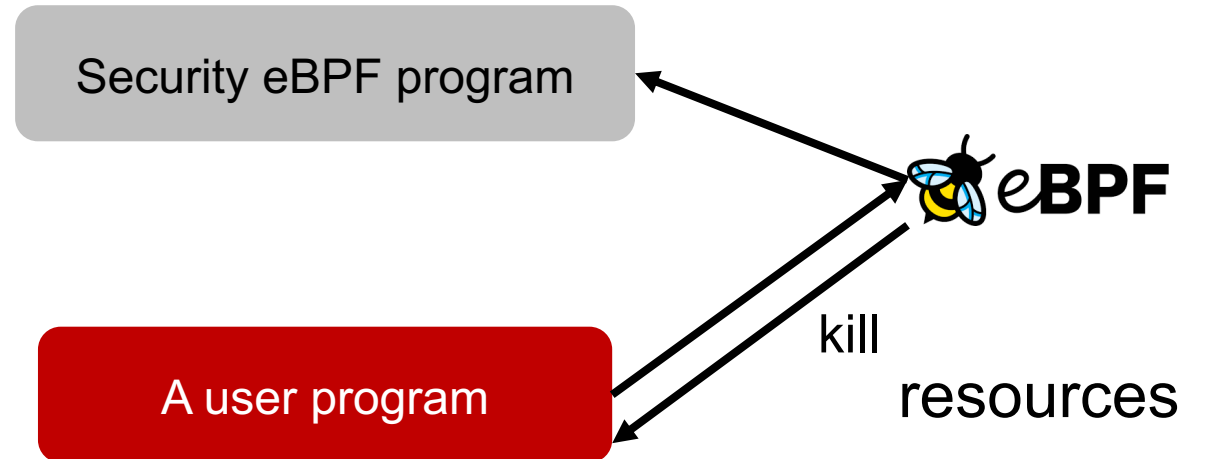
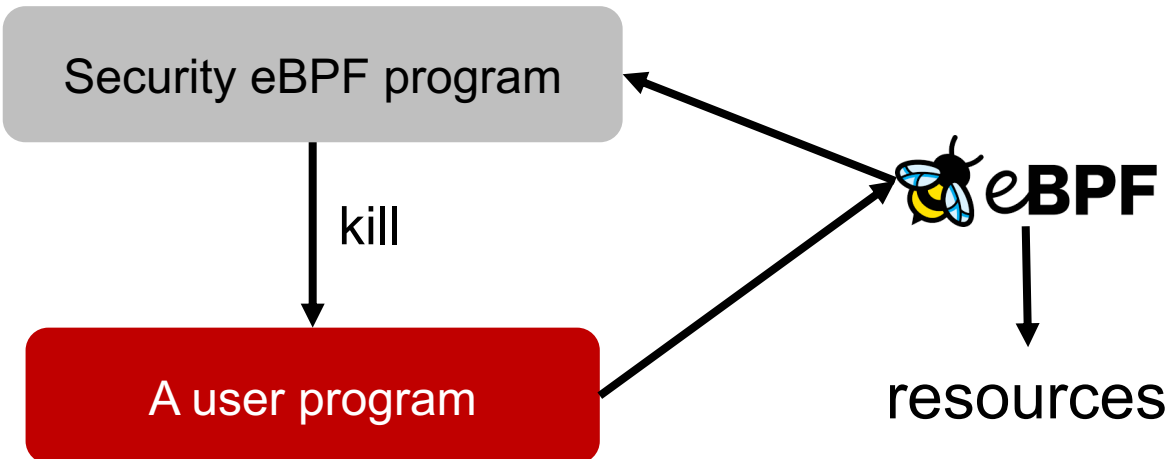
## The two types of control

### Asynchronous control

- Attach tracepoints or probes to kernel functions

### Synchronous control

- Use the LSM (Linux Security Module) hooks - requires kernel 5.7+
- Same behavior as SELinux and AppArmor





**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

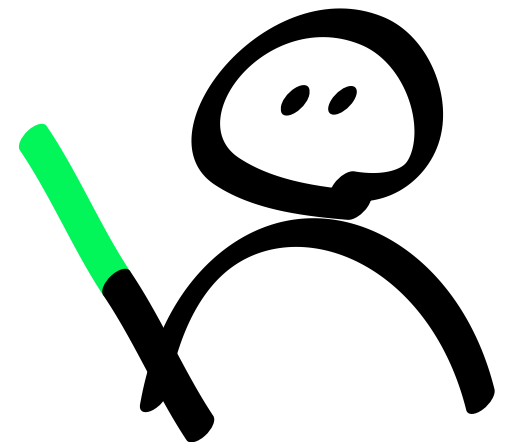
# Our Experiments – **the eBPF tool**

---

# Our Experiments

## The eBPF tools we developed

- Named project rpam (Rough Privilege Access Management)
- Wrote the **deny\_root** eBPF program with the intention to be used as a HIPS
  - deny\_unshare (blocks unshare, clone and clone3 syscalls)
  - deny\_ldapsearch (blocks connections to the ldap server – still in development)
- It uses libbpf in C producing a CORE binary
- It contains 18 files, including Makefile and clang-format
- Developed using DevSecOps principles
- Implemented Static Application Security Testing (SAST) controls
  - flawfinder, semgrep, and cppcheck
- Implemented Software Composition Analyses (SCA) control
  - aquasec trivy
  - Plan to also include Jfrog's XRAY
- It is based on the synchronous control (LSM eBPF hook)



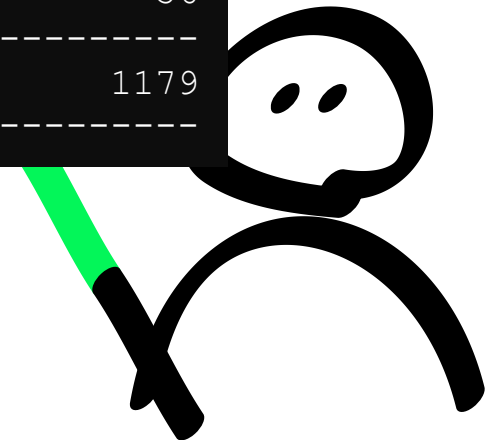
# Our Experiments

## The eBPF tools we developed

- Named `cloc`
- Wrote the tool
  - deny\_
  - deny\_
- It uses `ls`
- It contains
- Developed
- Implemented
  - flawfinder
- Implemented
  - aquasec trivy
  - Plan to also include Jfrog's XRAY
- It is based on the synchronous control (LSM eBPF hook)

```
$ cloc .
 18 text files.
 18 unique files.
  0 files ignored.

github.com/AlDanial/cloc v 1.97 T=0.05 s (341.0 files/s, 29833.5 lines/s)
-----
Language             files      blank      comment      code
-----
C                     8          160         122          934
C/C++ Header         8           46           4          101
YAML                  1            4           31           88
make                  1            20           9           56
-----
SUM:                  18          230         166         1179
-----
```



# How does deny\_root work?

## It blocks setresuid

```
[myuser@centos ~]$ sudo -i -u anotheruser
sudo: PERM_ROOT: setresuid(0, -1, -1): Cannot allocate memory
sudo: error initializing audit plugin sudoers_audit
```

## deny\_root enabled

```
[myuser@centos ~]$ getenforce
Enforcing
```

```
[myuser@centos ~]$ sudo -i
sudo: PERM_ROOT: setresuid(0, -1, -1): Cannot allocate memory
sudo: error initializing audit plugin sudoers_audit
```

## deny\_root disabled

```
[myuser@centos ~]$ getenforce
Enforcing
```

```
[myuser@centos ~]$ sudo -i
sudo: PERM_SUDOERS: setresuid(-1, 1, -1): Operation not permitted
sudo: no valid sudoers sources found, quitting
sudo: setresuid() [0, 0, 0] -> [1002, -1, -1]: Operation not permitted
sudo: error initializing audit plugin sudoers_audit
```

# Static Application Security Tests

## The SAST results

```
$ flawfinder deny_root
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
...
ANALYSIS SUMMARY:

No hits found.
```

```
$ semgrep scan --config auto deny_root

Scan Status

...

Ran 1072 rules on 106 files: 0 findings.
```



# Static Application Security Tests

## The SAST results

```
$ /centos/cppcheck/build/bin/cppcheck --inline-suppr --enable=all --inconclusive deny_root
Checking deny_root/deny_root.bpf.c ...
deny_root/deny_root.bpf.c:11:0: information: Include file: "vmlinux.h" not found. [missingInclude]

^
1/8 files checked 38% done
Checking deny_root/deny_root.c ...
deny_root/deny_root.c:20:0: information: Include file: "deny_root.skel.h" not found. [missingInclude]

^
2/8 files checked 65% done
Checking deny_root/deny_root_config.c ...
3/8 files checked 72% done
Checking deny_root/shamerock_conf_file.c ...
4/8 files checked 86% done
Checking deny_root/shamerock_error.c ...
5/8 files checked 86% done
Checking deny_root/shamerock_types.c ...
6/8 files checked 94% done
Checking deny_root/shamerock_util.c ...
7/8 files checked 96% done
Checking deny_root/strncpy.c ...
8/8 files checked 100% done
```

# Software Composition Analysis

## The SCA results

```
$ /centos/trivy fs /centos/rpam/deny_root/ --scanners vuln,config,secret,license
2023-05-04T01:33:10.886+0200 INFO Vulnerability scanning is enabled
2023-05-04T01:33:10.886+0200 INFO Misconfiguration scanning is enabled
2023-05-04T01:33:10.887+0200 INFO Secret scanning is enabled
2023-05-04T01:33:10.887+0200 INFO If your scanning is slow, please try '--scanners vuln' to
disable secret scanning
2023-05-04T01:33:10.887+0200 INFO Please see also
https://aquasecurity.github.io/trivy/v0.40/docs/secret/scanning/#recommendation for faster secret
detection
2023-05-04T01:33:10.887+0200 INFO License scanning is enabled
2023-05-04T01:33:11.284+0200 INFO Number of language-specific files: 0
2023-05-04T01:33:11.285+0200 INFO Detected config files: 0
```



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Our Experiments – **the Lab setup**

---

# Our Lab

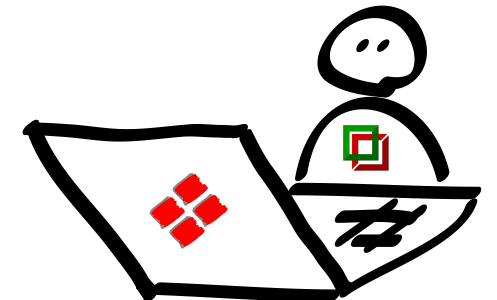
## The vulnerable systems – Polkit exploit - CVE 2021-4034



- CentOS Stream 9
- Updated all packages, except polkit
- The system has SELinux enabled
- Setup confined user myuser (user\_t)
- Setup confined user centos (unconfined\_t)



- Ubuntu 20.04
- Updated all packages, except polkit
- The system has AppArmor enabled
- No AppArmor RBAC implemented





**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Our Experiments – **CentOS** results

---



# CentOS machine results

## System without deny\_root

### SELinux disabled

```
[myuser@centos]$ getenforce
Permissive

[myuser@centos]$ ./cve-2021-4034
sh-5.1# id
uid=0(root) gid=0(root) ...
sh-5.1#
```



# CentOS machine results

## System without deny\_root

### SELinux enabled without confined user

```
[centos@centos]$ getenforce
Enforcing

[centos@centos]$ ./cve-2021-4034
sh-5.1# id
uid=0(root) gid=0(root) ...
sh-5.1#
```



### SELinux enabled with confined user

```
[myuser@centos]$ getenforce
Enforcing

[myuser@centos]$ ./cve-2021-4034
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"
The value for the SHELL variable was not found the /etc/shells file

This incident has been reported.
```



# CentOS machine results

## System with deny\_root

### SELinux disabled

```
[myuser@centos]$ getenforce  
Permissive
```

```
[myuser@centos]$ ./cve-2021-4034  
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"  
Cannot run program pwnkit.so:.: No such file or directory
```





# CentOS machine results

## System with deny\_root

### SELinux enabled without confined user

```
[centos@centos]$ getenforce  
Enforcing
```

```
[centos@centos]$ ./cve-2021-4034  
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"  
Cannot run program pwnkit.so:.: No such file or directory
```



### SELinux enabled with confined user

```
[myuser@centos]$ getenforce  
Enforcing
```

```
[myuser@centos]$ ./cve-2021-4034  
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"  
Cannot run program pwnkit.so:.: No such file or directory
```



# CentOS machine results

## Can the systems be exploited?

Test conditions	Without deny_root	With deny_root
SELinux enabled without confined user	Exploited	Not exploited
SELinux enabled with confined user	Not exploited	Not exploited
SELinux disabled	Exploited	Not exploited

**deny\_root is able to block the privilege escalation**



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

## Our Experiments – **Ubuntu** results

---



ubuntu

# Ubuntu machine results

## System without deny\_root

### Default AppArmor enabled

```
[myuser@ubuntu]$ sudo aa-status
apparmor module is loaded.
33 profiles are loaded.
... # no pkexec profile

[myuser@ubuntu]$ ./cve-2021-4034
sh-5.1# id
uid=0(root) gid=0(root) ...
sh-5.1#
```



# Ubuntu machine results

## System without deny\_root

### AppArmor enabled but with custom pkexec rule

```
[myuser@ubuntu]$ sudo aa-status  
apparmor module is loaded.  
4096 profiles are loaded.  
...  
/usr/bin/pkexec  
...
```



```
[myuser@ubuntu]$ ./cve-2021-4034  
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"  
Cannot run program pwnkit.so:.: No such file or directory
```

# Ubuntu machine results

## System without deny\_root

AppArmor enabled but with custom pkexec rule

Copy pkexec to local storage

```
[myuser@ubuntu]$ sudo aa-status
apparmor module is loaded.
4096 profiles are loaded.
...
/usr/bin/pkexec
...

[myuser@ubuntu]$ cp /usr/bin/pkexec .
[myuser@ubuntu]$ ./cve-2021-4034-local-pkexec
sh-5.1# id
uid=0(root) gid=0(root) ...
sh-5.1#
```



# Ubuntu machine results

## System without deny\_root


AppArmor enabled but with custom pkexec rule

Copy pkexec to local storage

Add nosuid to mount point

```
[myuser@ubuntu]$ sudo aa-status
apparmor module is loaded.
4096 profiles are loaded.
...
/usr/bin/pkexec
...

[myuser@ubuntu]$ cp /usr/bin/pkexec .
[myuser@ubuntu]$ ./cve-2021-4034-local-pkexec
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"
Cannot run program pwnkit.so:.: No such file or directory
```



# Ubuntu machine results

## System with deny\_root

### Default AppArmor enabled

```
[myuser@ubuntu]$ sudo aa-status  
apparmor module is loaded.  
33 profiles are loaded.  
... # no pkexec profile
```

```
[myuser@ubuntu]$ ./cve-2021-4034  
GLib: Cannot convert message: Could not open converter from "UTF-8" to "PWNKIT"  
Cannot run program pwnkit.so:.: No such file or directory
```



Skipping the rest of the results for brevity



# Ubuntu machine results

Can the systems be exploited?

Test conditions	Without deny_root	With deny_root
Default Apparmor	Exploited	Not exploited
Additional pkexec rules	Not exploited	Not exploited
Additional pkexec rules, copy binary to local storage	Exploited	Not exploited
Additional pkexec rules, copy binary to local storage, mount local storage using nosuid	Not exploited	Not exploited

**deny\_root is able to block the privilege escalation**

# What are the disadvantages?

There is no free lunch!

- The eBPF code and the user namespace application have to be maintained
  - It may break in future kernel, libbpf, and bpftool releases
- Does it have performance impact?
  - We don't know yet. The tests will be performed in the next weeks
- There could be exploits in the user namespace part of the code
  - Even though we implement SAST and SCA controls
  - We will do human source code analyses in the next weeks

# Take home message

## What have we touched?

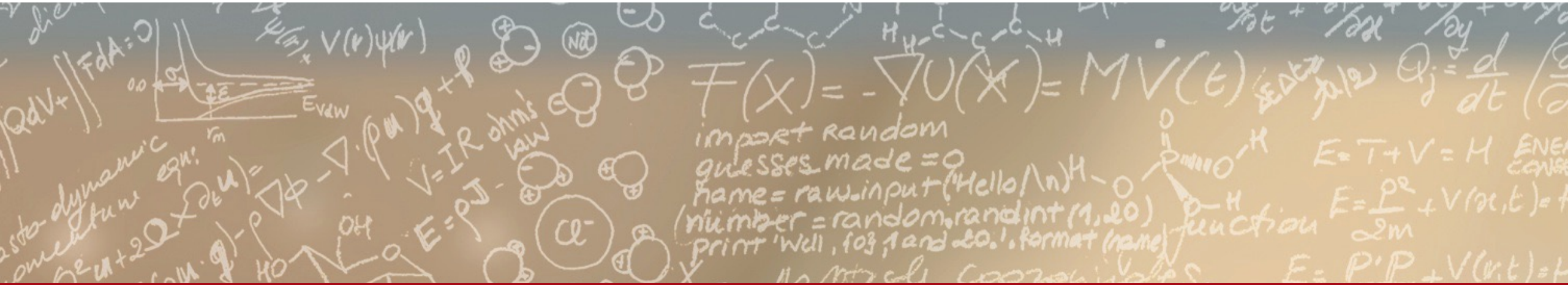
- eBPF can be used for implementing HIPS and HIDS systems
- deny\_root is a HIPS for privilege escalation
- deny\_root is able to block attempts to become root, or any other user
- It has been developed following DevSecOps practices
- It is distro independent
- The impact on performance is unknown at this point in time
- Should mount all filesystems that users can write as nosuid
- One needs to implement SELinux and AppArmor properly
  - SELinux – confined user
  - AppArmor – implement rules for all binaries in the system and have nosuid filesystems



CSCS

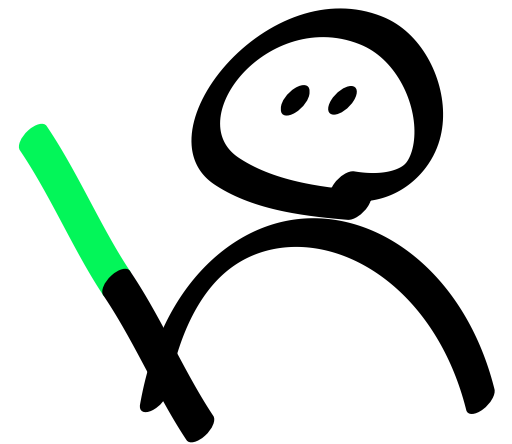
Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

ETH zürich



Thank you for your attention.

Questions?



May the 4th be with you