Cybersecurity – EC521
Unix Security

Manuel Egele
PHO 337
megele@bu.edu
Boston University

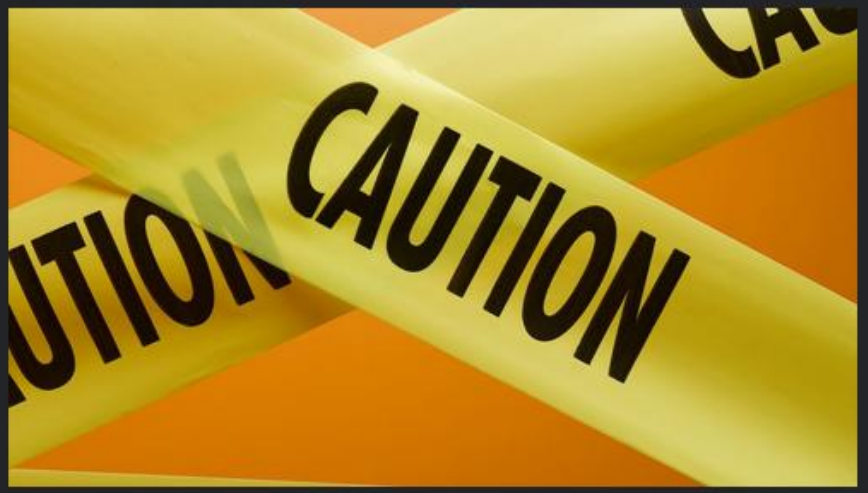
News From the Field …

GET YOUR PATCHING ON

SAP warns of high-severity vulnerabilities in multiple products

Users of SAP's S/4HANA and NetWeaver products are at risk and should patch soon.

DAN GOODIN – SEP 9, 2025 3:55 PM | 0



Credit: Getty Images

Project Mechanics: Resources

- Academic security conferences (e.g., Oakland, Usenix, …)
- Applied security conferences (e.g., Defcon, …)
- Security-focused blogs (r/netsec)
- Popular media coverage of Security issues
- Past MITRE eCTF competitions

- **Goal:** Become knowledgeable enough in two topics to propose a meaningful project.

- Note: This will take effort from your part that goes beyond reading blogs and news. (i.e., you must be able to identify and ***grok*** the root-cause of the problem before you can propose a good project).

Cybersecurity – EC521
Unix Security

Manuel Egele
PHO 337
megele@bu.edu
Boston University

Unix

- Multi-user operating system
- Operating system functionality
  - Process management
  - Memory management
  - File system
  - Input/Output
- Kernel-User separation
  - Privileged OS kernel
  - Unprivileged user-space programs (daemons, applications, shell, etc.)

Unix Kernel

- Abstracts hardware implementation details (e.g., different network cards, etc.)
- Complete access to all (physical) resources
- Trusted computing base
- Provides unified services via *system calls*

System Calls

- Precisely defined interface that allows user-mode code to request kernel services
- Transitions from (unprivileged) user-mode to (privileged) kernel-mode
- Crosses the border between two security domains
- Implemented with hardware (CPU) support
  - Software interrupts
  - Call gates
  - Special instructions (SYSENTER/SYSCALL)

Unix Kernel Vulnerabilities

Vulnerability

- Usually complete system compromise
- Attacks performed via *system calls*

Solaris/NetBSD call gate creation input validation

- Malicious input when creating a LDT (x86 local descriptor table)
- Used in 2001 by Last Stage of Delirium to win Argus Pitbull Competition

Kernel Integer Overflows

- FreeBSD `procfs()` code (September 2003)
- Linux `brk()` used to compromise debian.org (Dec. 2003)
- Linux `setsockopt()` (May 2004)
- Linux `vmsplice()` (Feb. 2008)

Syzkaller – Fuzzing the Linux Kernel

https://syzkaller.appspot.com/upstream

open (755):

Title	Repro	Cause bisect	Fix bisect	Count	Last	Reported	Discussions
KASAN: slab-use-after-free Read in pvr2_context_set_notify ...	C			10	38m	20h13m	0 [20h13m]
WARNING in tcp_disconnect (2) net				1	4d11h	22h39m	0 [22h39m]
WARNING: ODEBUG bug in __init_work (4) rdma				1	5d16h	1d02h	0 [1d02h]
possible deadlock in blocking_notifier_call_chain kernel				1	1d08h	1d04h	0 [1d04h]
BUG: unable to handle kernel NULL pointer dereference in ...	C	done		4	3d13h	2d00h	PATCH [1d23h]
kernel BUG in validate_mm (3) mm	C	error		3	2d02h	2d02h	2 [1d01h]
KMSAN: uninit-value in nci_rsp_packet net nfc				1	2d18h	2d04h	0 [2d04h]
WARNING in cfg802154_switch_netns wpan				1	2d22h	2d04h	0 [2d04h]
INFO: task hung in migrate_pages_batch ext4 nilfs	C			20	6d20h	2d20h	1 [1d19h]
INFO: task hung in hci_conn_failed bluetooth	C	done		1	7d13h	3d13h	12 [1h48m]
BUG: unable to handle kernel paging request in bpf_probe_...	C	done		4	8d05h	4d12h	0 [4d12h]
INFO: task hung in ntfs_evict_big_inode ntfs	syz			4	8d19h	4d18h	0 [4d18h]
kernel BUG in resv_map_release mm	C	done		34	2d13h	4d20h	0 [4d20h]
general protection fault in jbd2__journal_start xfs ext4	C	error		2	22h25m	5d04h	PATCH [1h13m]
possible deadlock in __unmap_hugepage_range mm	C	done		31	2d12h	5d04h	4 [5d03h]
WARNING: refcount bug in p9_req_put (3) net v9fs				2	3d22h	5d04h	2 [20h36m]
INFO: task hung in blk_trace_remove (2) block trace	C	done		4	8d19h	5d17h	14 [37m]

User Management

Code running in user mode is always linked to a certain identity

- Security checks and access control decisions are based on user identity

Unix is user-centric

- No roles
- Users Identified by user id (uid) & group id (gid)
- Authenticated by password (stored salted & hashed)

User root

- Superuser, system administrator
- Special privileges (access resources, configure the OS)
- Cannot “decrypt” user passwords

Process Management

Process

- Implements user-activity
- Entity that executes a given piece of code
- Own stack, memory pages, and file descriptors
- Separated from other processes using virtual memory

Thread

- Separate stack and program counter
- Shares memory and file descriptors with other threads of the same process

Process Management

Process Attributes

- Process id (PID)
  - Uniquely identifies a process
  - PIDs are reused
- User id (UID)
  - ID of owner of process
- Effective user id (EUID)
  - ID used for permission checks / access control
- Saved user id (SUID)
  - To temporarily drop and restore privileges
- Lots of management information
  - Scheduling
  - Memory management
  - Resource management

User Authentication

How does a process get a user ID?

- Authentication via `login`

cf. *identification* & *authentication* from Lecture 2

Passwords

- User passwords used as key for `crypt()` function
- Public salt (prevent pw collisions)
- Repeatedly apply encryption/hash

Password cracking

- Dictionary attacks
- Crack, JohnTheRipper

User Authentication

File /etc/passwd

- Maps user names to user ids (many applications legitimately need this)
- No legitimate need for encrypted passwords

File /etc/shadow

- Contains salted & hashed passwords
- Account information (last change, expiration)
- Readable only by superuser and privileged programs
- Different hash algorithms supported
  - DES
  - MD5
  - SHA-{256,512}

DEMO
passwd vs. shadow

Unix Groups

Users belong to one or more groups

- Primary group (stored in `/etc/passwd`)
- Additional groups (stored in `/etc/group`)
- Possibility to set group password
- Become group member with `newgrp`

File /etc/group

```
groupname : password : group id : additional users
root:x:0:root
bin:x:1:root,bin,daemon
users:x:1000:pizzaman
```

Special group wheel

- Group for users that can call `su`

DEMO
id

File System

Directory tree

- Primary repository of information
- Hierarchical set of directories
- Directories contain file system objects (FSO)
- Root is denoted as "/"

File system objects (FSO)

- Files, directories, symlinks, sockets, device files
- FSOs Have names but are really referenced by inode (index node)

File System

- Access Control
  - Permission bits
  - chmod, chown, chgrp, umask
  - File listing

```
     -          rwx     rwx          rwx
(file type) (user) (group) (other/world)
```

Type	r	w	x	s	t
File	Read access	Write access	Execute	suid / sgid inherit id	sticky bit
Directory	List files	Add and remove files	Stat / execute files, chdir	New files have dir-gid	Files only deletable by owner

SUID Programs

Each process has *real* and *effective* user / group id

- Usually identical
- Real id
  - Determined by current user
  - `login, su`
- Effective ids
  - Determine access rights of a process
  - System calls (e.g., `setuid(), setgid(),` etc.)
- `suid/sgid` bits
  - Start process with effective ID different from real ID
  - Attractive targets for attacker

Why does `login` need to be suid root?

No SUID shell scripts anymore

DEMO
suid program
setgid directory