Operating Systems Security

Asst. Prof. Mihai Chiroiu

SYSTEMS LABORATORY

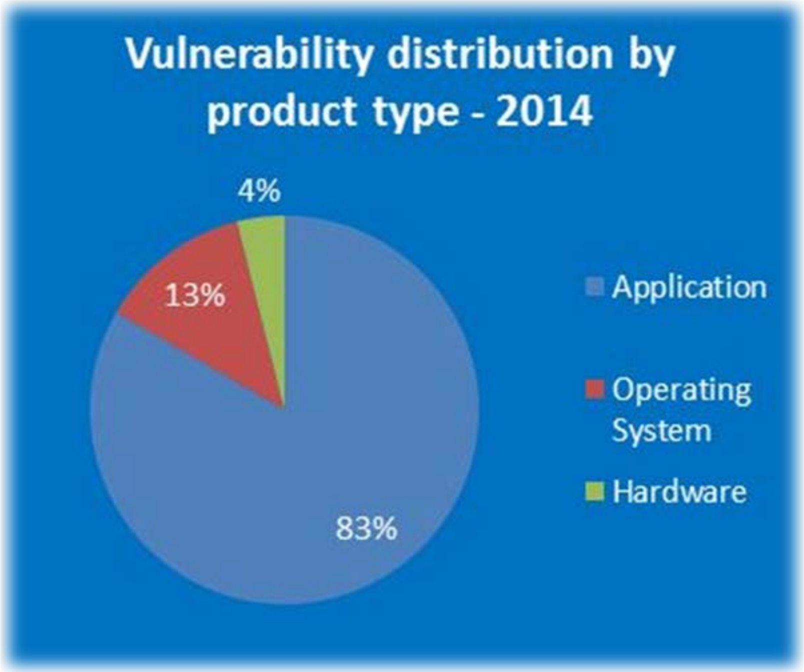
Computer Science & Engineering Department

OS principles

- hardware abstraction
- resource management: accounting, scheduling, and synchronisation
- storage and communication services: file systems, network, inter-process communication (IPC)
- libraries of common functions: libc
- management of user interaction and interface

- More here: http://ocw.cs.pub.ro/courses/so

Stats (2014)



Operating system	# of vulnerabilities	# of HIGH vulnerabilities	# of MEDIUM vulnerabilities	# of LOW vulnerabilities
Apple Mac OS X	147	64	67	16
Apple iOS	127	32	72	23
Linux Kernel	119	24	74	21
Microsoft Windows Server 2008	38	26	12	0
Microsoft Windows 7	36	25	11	0
Microsoft Windows Server 2012	38	24	14	0
Microsoft Windows 8	36	24	12	0
Microsoft Windows 8.1	36	24	12	0
Microsoft Windows Vista	34	23	11	0
Microsoft Windows RT	30	22	8	0

http://www.gfi.com/blog/most-vulnerable-operating-systems-and-applications-in-2014/

What should the OS protect?

- Itself (from users)
- Processes (both services and user's application)
- Files access
- Communication (both IPC and network)

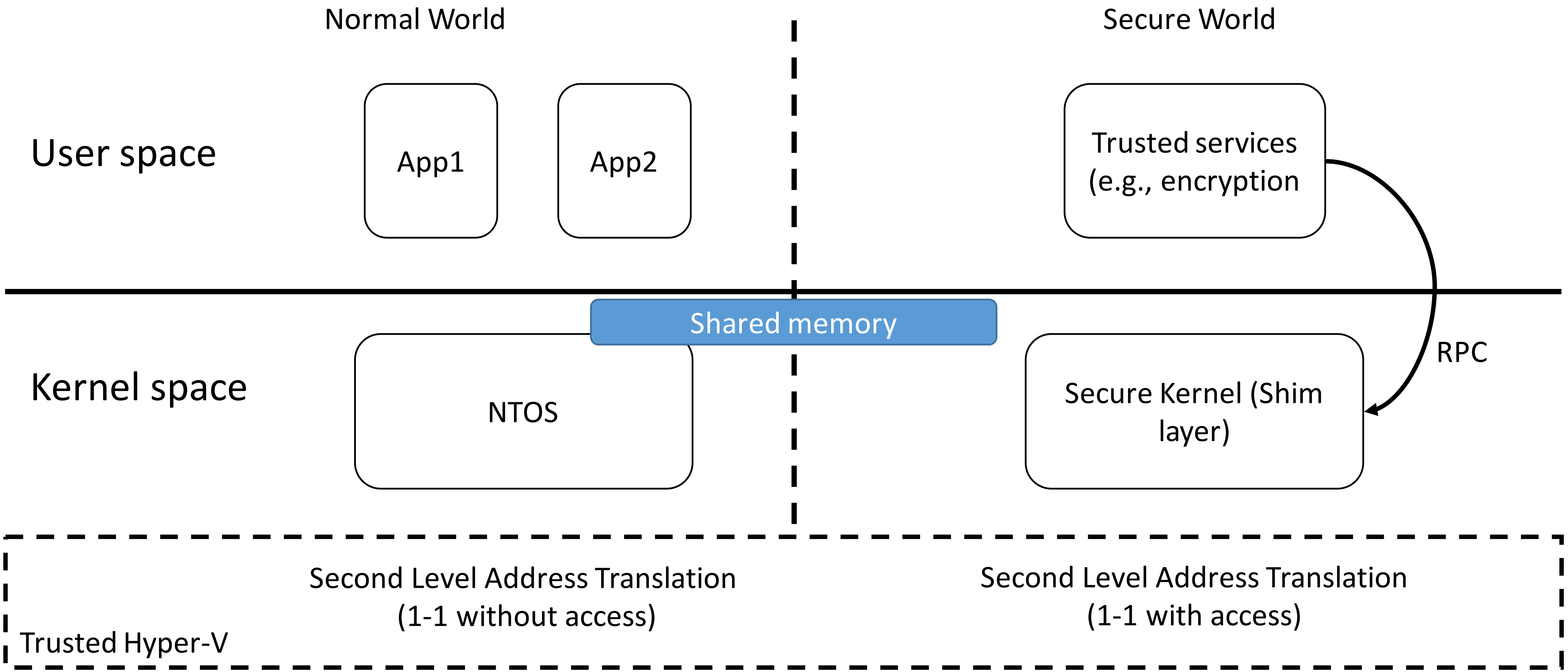
First, authentication

- Most common technique are passwords (i.e., something you know)
  - Stored as hashes typically using a random *salt*
- Tokens (i.e., something you have)
  - Using HSM
  - Often combined with a PIN
- Biometrics (i.e., something you are)
  - Fingerprints, iris scans, etc.

- We will assume that authentication is validated!

Windows 10

Virtualization-based security (VBS)



Code Integrity

- Kernel Mode Code Integrity (KMCI)
  - Validate drivers’ signature
- User Mode Code Integrity (UMCI)
  - Validate apps signature
- AppLocker
  - Policy for what applications can be executed

Protected Processes

- Windows 10 prevents untrusted processes from interacting or tampering with those that have been specially signed.

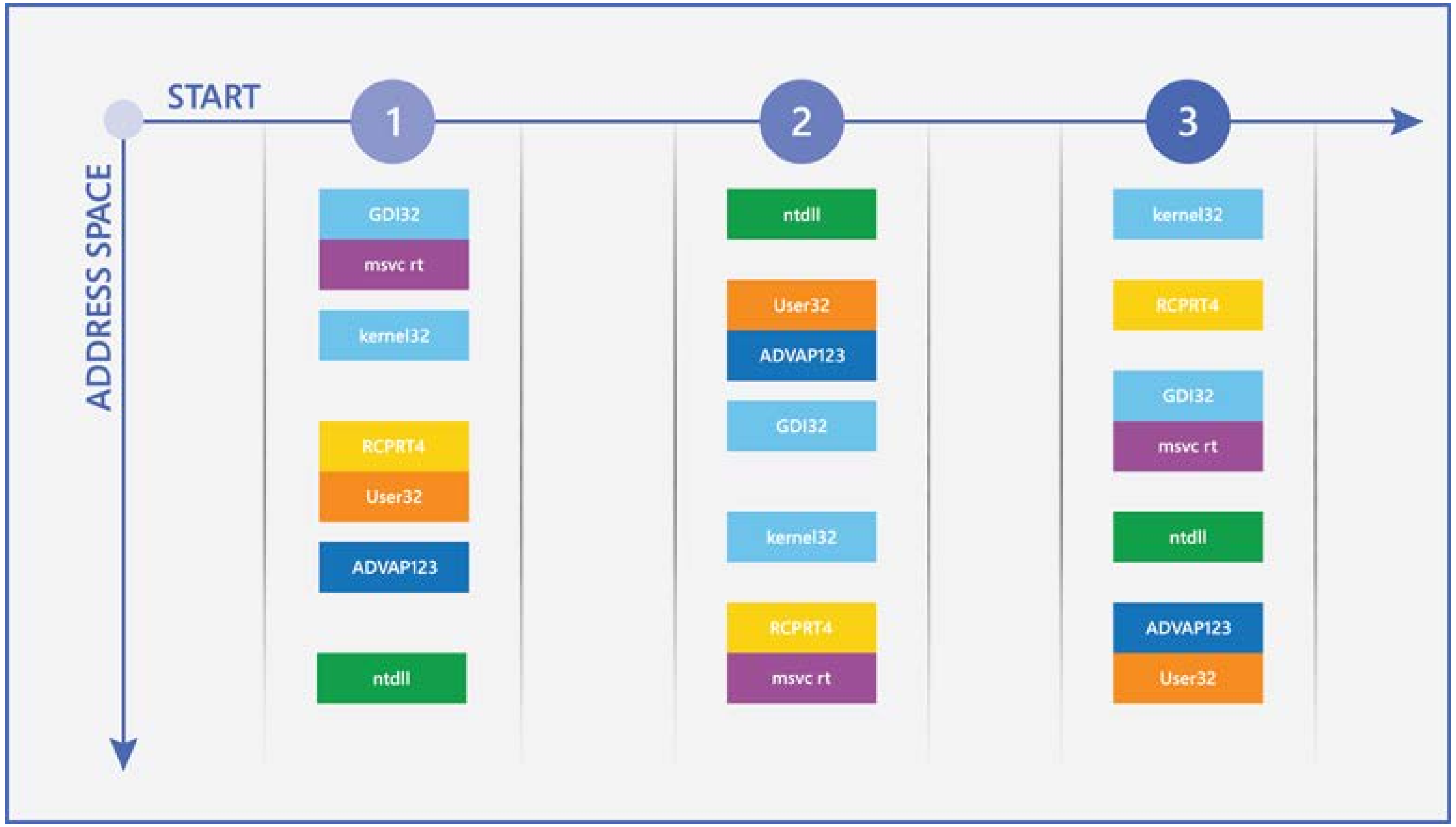
- Protected Processes defines levels of trust for processes.

- Less trusted processes are prevented from interacting with and therefore attacking more trusted processes.

Address Space Layout Randomization (ASLR)

- Present in most OSes
- Not a real solution

(part of a complex one) [1]



ASLR implementation

- On Windows, ASLR does not affect runtime performance, but it can slow down the initial loading of modules.
  - ASLR also randomizes heap and stack memory
- On Linux, ASLR imposes 26% [9]
- On Android, ASLR bases for all others and the bases remain constant across executions [10]
- On iOS, dyld_shared_cache (libraries) load address is randomized (at boot time) [11]
- ASLR cannot be force-enabled for applications on Linux (they must be compiled with PIE), as EMET can do on Windows.

Data Execution Prevention (DEP)

- DEP uses the No eXecute bit on modern CPUs
- Available on all major Oses
- Not real use if you can access mprotect/VirtualProtect/etc.

TrueCrypt - Full-disk encryption (3rd party)

- Password used to encrypt/decrypt when mounting the partition.
- Supports plausible deniability
  - can be configured to hide even the existence of encrypted data.
  - Unused space on an encrypted partition is initialized with random data, encrypted volume is indistinguishable from such random data.

BitLocker – Full-disk encryption

- Encrypting entire hard drives
- Support for Self-Encrypting Drives (SED) for offloading encryption
- Uses Trusted Platform Module (TPM) v1.2 to validate pre-OS components

Where's the Encryption Key?

1. SRK (Storage Root Key) contained in TPM
2. SRK encrypts FVEK (Full Volume Encryption Key) protected by TPM/PIN/USB Storage Device
3. FVEK stored (encrypted by SRK) on hard drive in the OS Volume

File permissions

- Stored as an ACE in a discretionary access control list (DACL) that is part of the object's security descriptor.

- Permissions can also be explicitly denied.

- Inherited permissions are those that are propagated to a child object from a parent object.

Network access

- Per application firewall

Microsoft Bounty Programs

- Online Services Bug Bounty (Microsoft Azure services additions: 22nd April 2015)
  - $500 USD up to $15,000 USD.
- Mitigation Bypass Bounty (Windows 10)
  - up to $100,000 USD
- Bounty for Defense (Windows 10)
  - up to $100,000 USD

- https://technet.microsoft.com/en-US/security/dn425036

Linux

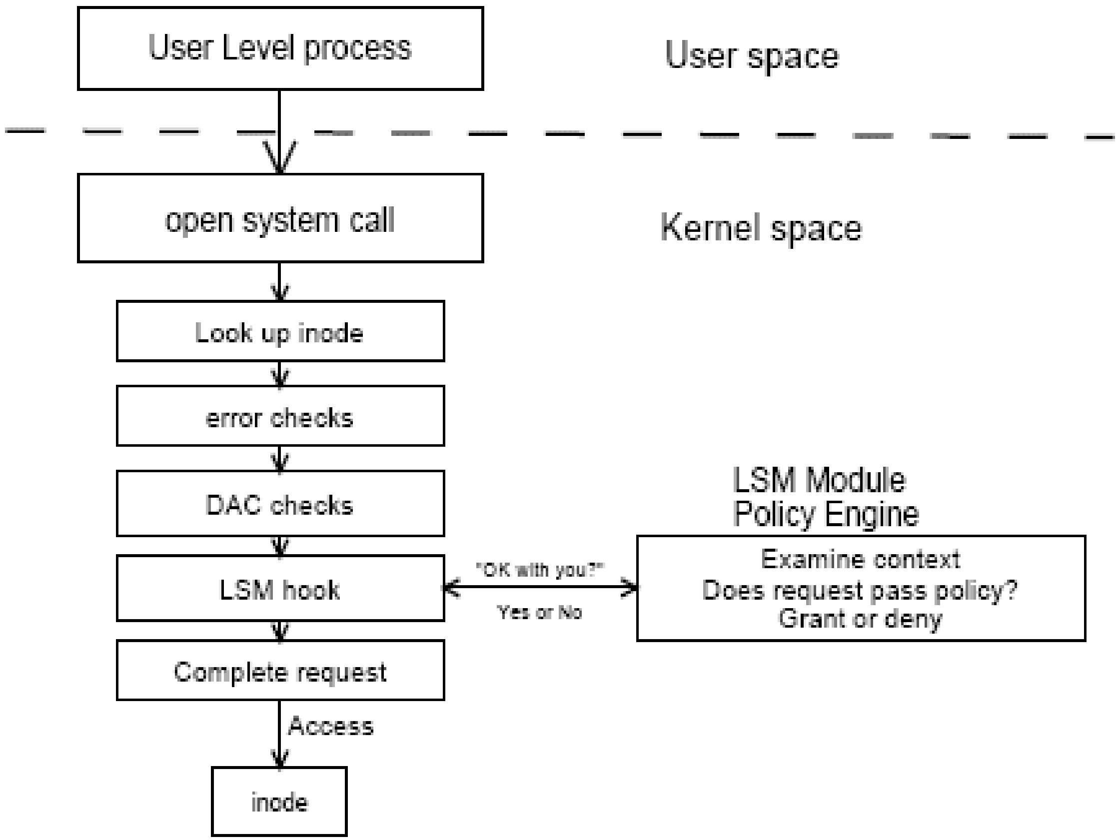
Linux - *setuid*

- Sometimes we want to specify that a file can only be modified by a certain program.
- Thus, we want to control access on a per-program, rather than a per-user basis.
- We can achieve this by creating a new user, representing the role of a modifier for these files.
- Mark the program, as *setuid* to this user.
- This means, no matter who started the program, it will run under the user id of this new user.

LUKS – Full-disk encryption [3]

- A master key is generated by the system (used to encrypt/decrypt data on disk)
- Protected using the user’s password
- Several master keys are stored, one for each user

Linux Security Modules (2002) [6]



- IPC Hooks
- Filesystem Hooks
- Network Hooks

SELinux

- Mandatory Access Control system for Linux
- Implement Flask architecture [7]

- A process (a daemon or a running program) is called a *subject*.
- A role defines which users can access that process.
- An *object* in SELinux is anything that can be acted upon
- A file's context is called its *type* in SELinux lingo

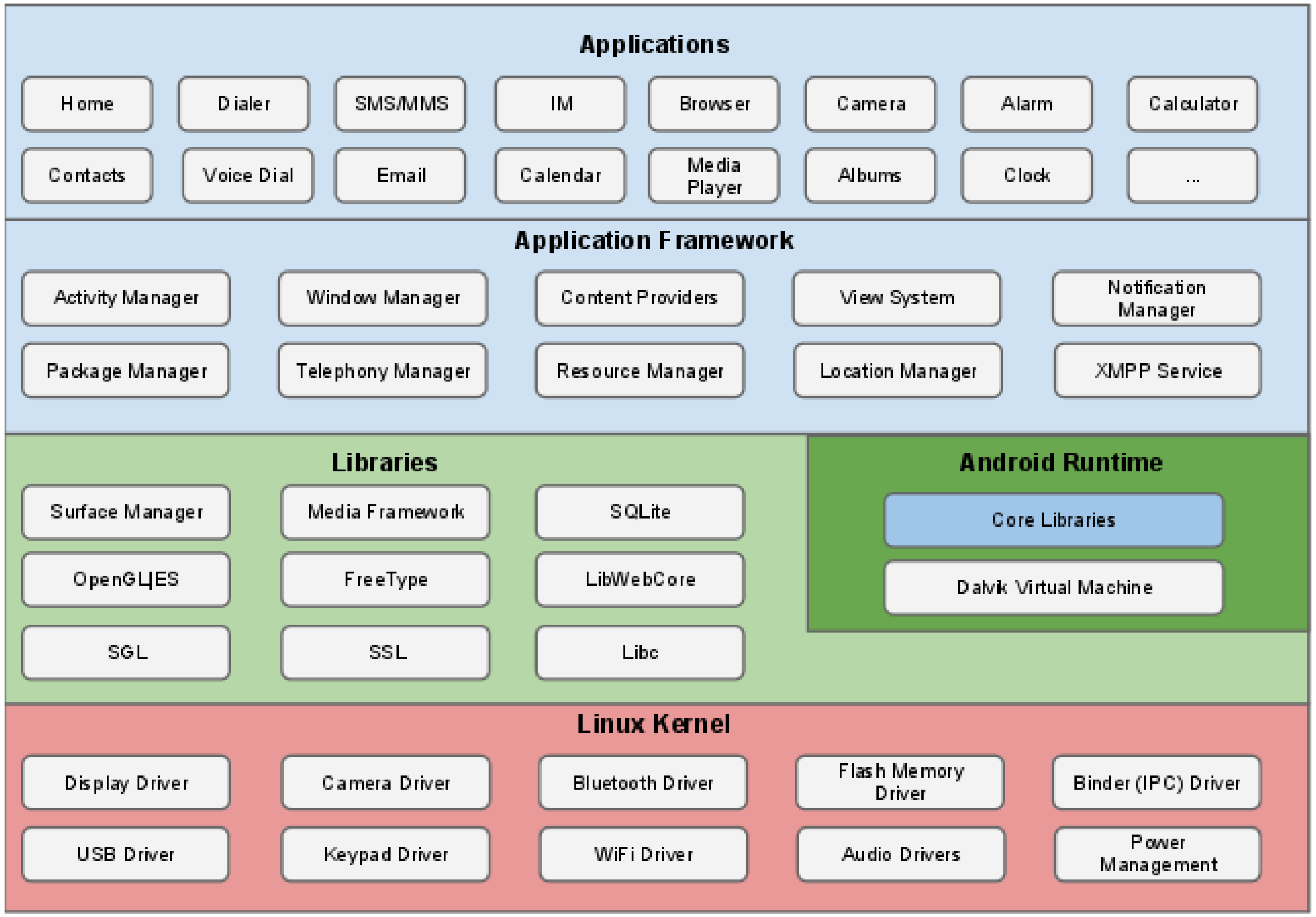
SELinux

- An SELinux policy defines user access to roles, role access to domains, and domain access to types.
- Possible modes are Enforcing, Permissive, or Disabled

- `-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/index.html`
- `system_u:system_r:httpd_t:s0      7126 ? 00:00:00 httpd`
- `sesearch --allow --source httpd_t --target httpd_sys_content_t --class file`
  - `allow httpd_t httpd_sys_content_t : file { ioctl read getattr lock open } ;`

Android

Android Architecture



Package (APK) integrity

- Components of applications
  - Activity: User interface
  - Service: Background service
  - Content Provider: SQL-like database
  - Broadcast receiver: Mailbox for broadcasted messages
- META-INF contains the application certificate and package manifest
- Certified by developer
- Used for: application upgrade; application modularity (two apps from same developer can collude);

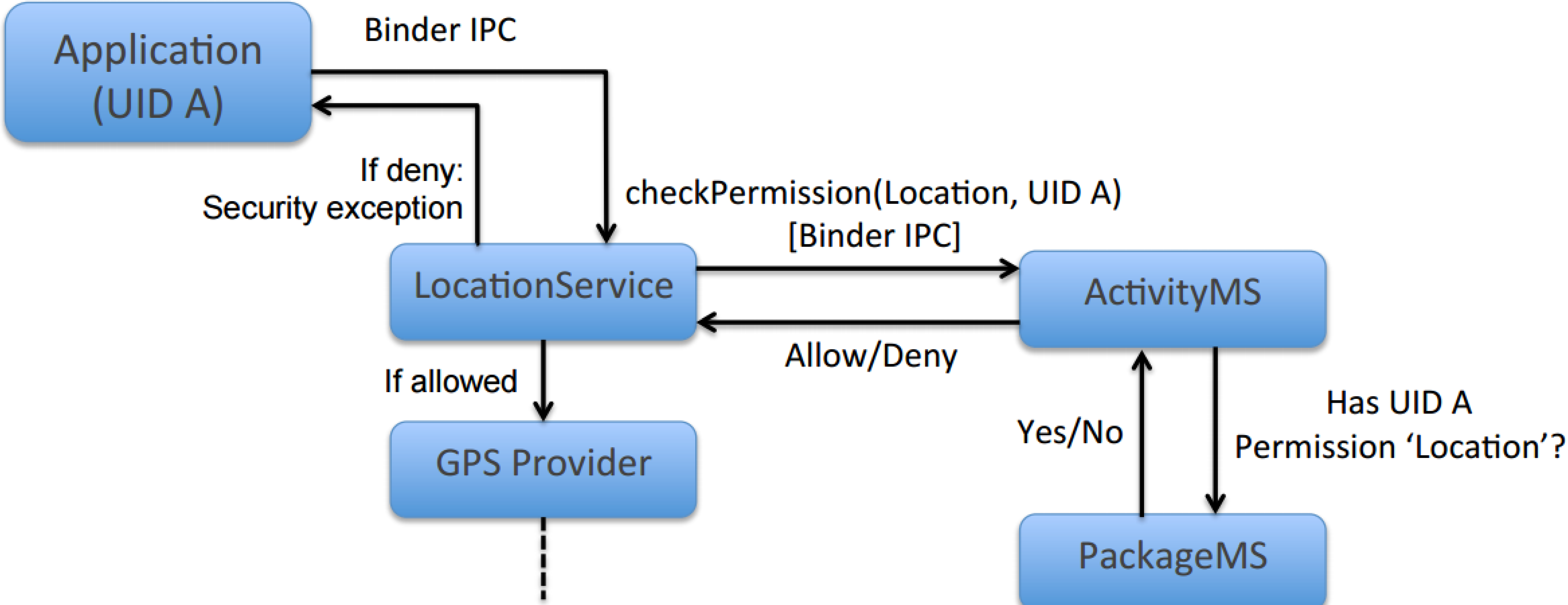
Android Security Basics

- Applications, by default, have no permissions
- Applications statically declare the permissions they require
  - Android system prompts the user for consent at the time the application is installed
  - No mechanism for granting permissions dynamically (at run-time)
  - In AndroidManifest.xml, add one or more [<uses-permission>]() tags
  - e.g., <uses-permission android:name= "android.permission.RECEIVE_SMS" />

Android Sandbox

- Each application is isolated in its own sandbox
  - Applications can access only its own resources
  - Access to sensitive resources depends on the application's rights
- Enforced by underlying Linux Kernel (SELinux) and middleware
- Each App is assigned a unique UserID during installation and runs in separate process

Android Sandbox



Android Sandbox

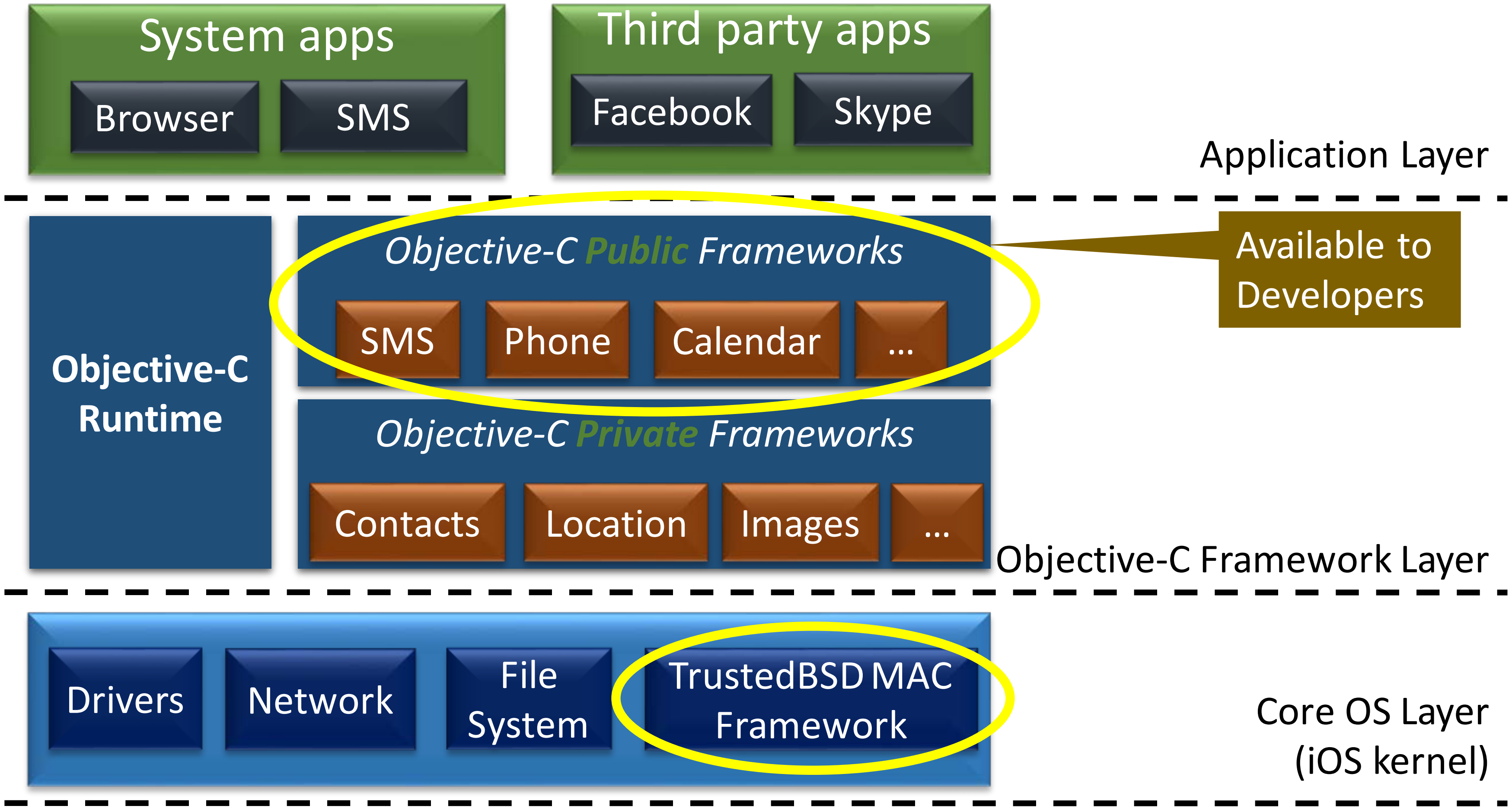
- App UID must be member of a Linux group to have access to sockets, etc.
- UID of an app with corresponding permission is added to group during install
- Kernel access errors translated into Java security exceptions by core libraries

Isolated Processes

- Security-aware application developer can declare in application manifest that a Service component should be executed as an isolated process
  - Component executed on separate process with UID nobody
  - Nobody is a UID with no privileges
    - All permission checks will return deny
    - No file system access
  - only communication with it is through the Service API
- Allows compartmentalization of the app

iOS

iOS Architecture



iOS Protection Mechanisms

- Encrypted file system
- Applications signing
- Vetting processs (app reviewing)
  - 700 - 1000 apps are submitted each day [Apple]
- Address Space Layout Randomization (ASLR)
- Non-executable memory security model (with code signing on memory pages)

Sandboxing

- Enforcement at the Objective-C runtime layer
  - That could be bypassed
- Enforcement by the TrustedBSD kernel module
  - Based on a generic profile that forces application containment (for IPC and files)
- Custom rules added by users are allowed

Xen VMM

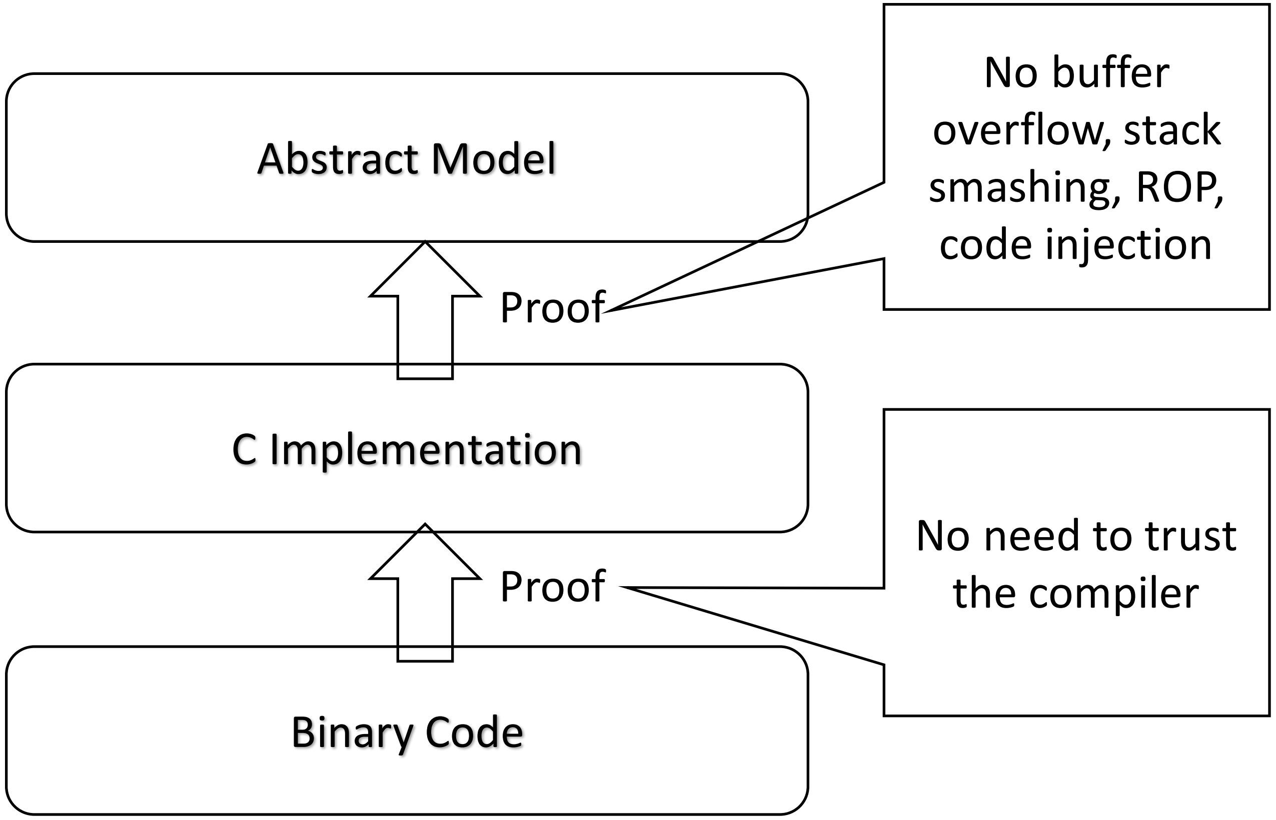
Security possibilities

- VM introspection
- Dom0 dissagregation
  - Driver domains
- Xen Security Module (same as LSM)
  - Restricts hypercalls to those needed by a particular guest

Formally verified security kernel

seL4 [4]

- Based on a minimal L4 kernel (drivers are outside kernel, user-mode processes)
- A refinement proof establishes a correspondence between a high-level (abstract) and a low-level (concrete, or refined) representation of a system.



References

- [1] https://www.trust.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_TRUST/PubsPDF/jit-rop.pdf
- [2] https://technet.microsoft.com/en-us/library/mt601297(v=vs.85).aspx
- [3] https://gitlab.com/cryptsetup/cryptsetup/wikis/LUKS-standard/on-disk-format.pdf
- [4] http://web1.cs.columbia.edu/~junfeng/09fa-e6998/papers/sel4.pdf

References

- [5] https://opensource.com/business/13/11/selinux-policy-guide
- [6] https://www.usenix.org/legacy/event/sec02/full_papers/wright/wright.pdf
- [7] https://www.nsa.gov/research/_files/publications/flask.pdf
- [8] http://css.csail.mit.edu/6.858/2012/readings/android.pdf
- [9] http://nebelwelt.net/publications/files/12TRpie.pdf

References

- [10] https://copperhead.co/blog/2015/05/11/aslr-android-zygote
- [11] http://antid0te.com/CSW2012_StefanEsser_iOS5_An_Exploitation_Nightmare_FINAL.pdf