Introduction to Computer Security Lecture Slides

Operating Systems Security

Associate Prof. Mihai Chiroiu

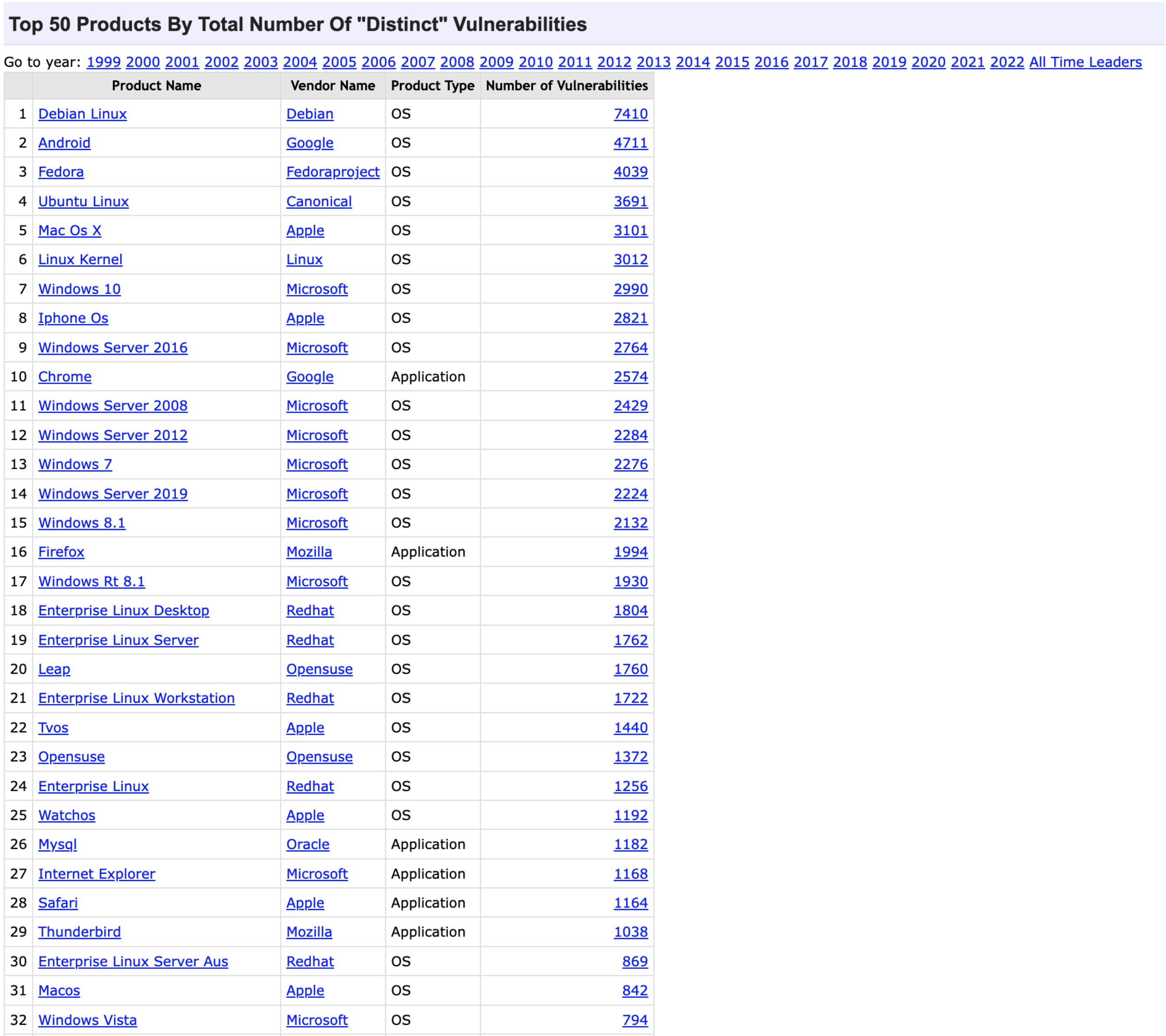
- There was once a young man who, in his youth, professed his desire to become a great writer. When asked to define "Great" he said, "I want to write stuff that the whole world will read, stuff that people will react to on a truly emotional level, stuff that will make them scream, cry, howl in pain and anger!" He now works for Microsoft, writing error messages.

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 (all time)



Top 50 Products By Total Number Of "Distinct" Vulnerabilities

Go to year: 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 All Time Leaders

	Product Name	Vendor Name	Product Type	Number of Vulnerabilities
1	Debian Linux	Debian	OS	7410
2	Android	Google	OS	4711
3	Fedora	Fedoraproject	OS	4039
4	Ubuntu Linux	Canonical	OS	3691
5	Mac Os X	Apple	OS	3101
6	Linux Kernel	Linux	OS	3012
7	Windows 10	Microsoft	OS	2990
8	Iphone Os	Apple	OS	2821
9	Windows Server 2016	Microsoft	OS	2764
10	Chrome	Google	Application	2574
11	Windows Server 2008	Microsoft	OS	2429
12	Windows Server 2012	Microsoft	OS	2284
13	Windows 7	Microsoft	OS	2276
14	Windows Server 2019	Microsoft	OS	2224
15	Windows 8.1	Microsoft	OS	2132
16	Firefox	Mozilla	Application	1994
17	Windows Rt 8.1	Microsoft	OS	1930
18	Enterprise Linux Desktop	Redhat	OS	1804
19	Enterprise Linux Server	Redhat	OS	1762
20	Leap	Opensuse	OS	1760
21	Enterprise Linux Workstation	Redhat	OS	1722
22	Tvos	Apple	OS	1440
23	Opensuse	Opensuse	OS	1372
24	Enterprise Linux	Redhat	OS	1256
25	Watchos	Apple	OS	1192
26	Mysql	Oracle	Application	1182
27	Internet Explorer	Microsoft	Application	1168
28	Safari	Apple	Application	1164
29	Thunderbird	Mozilla	Application	1038
30	Enterprise Linux Server Aus	Redhat	OS	869
31	Macos	Apple	OS	842
32	Windows Vista	Microsoft	OS	794

https://www.cvedetails.com/top-50-products.php

What should the OS protect?

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

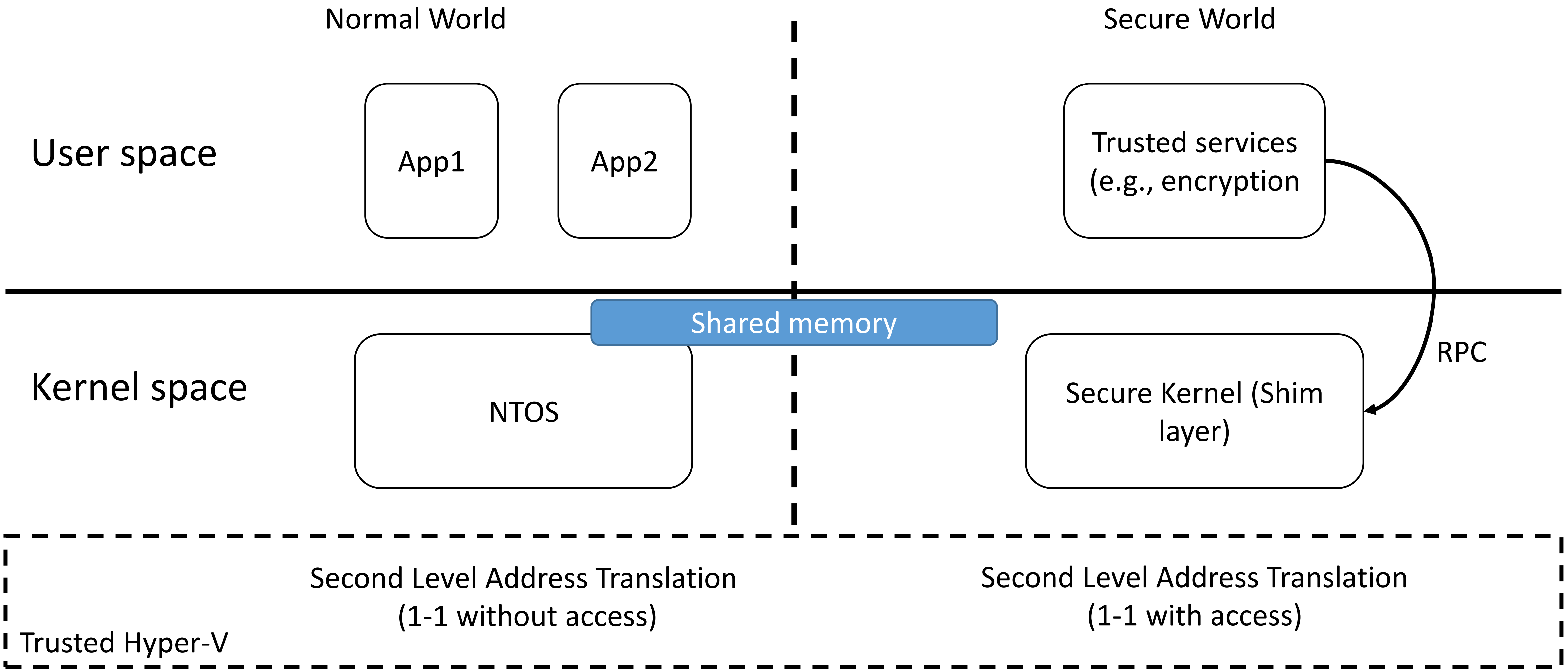
First, authentication

- 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.

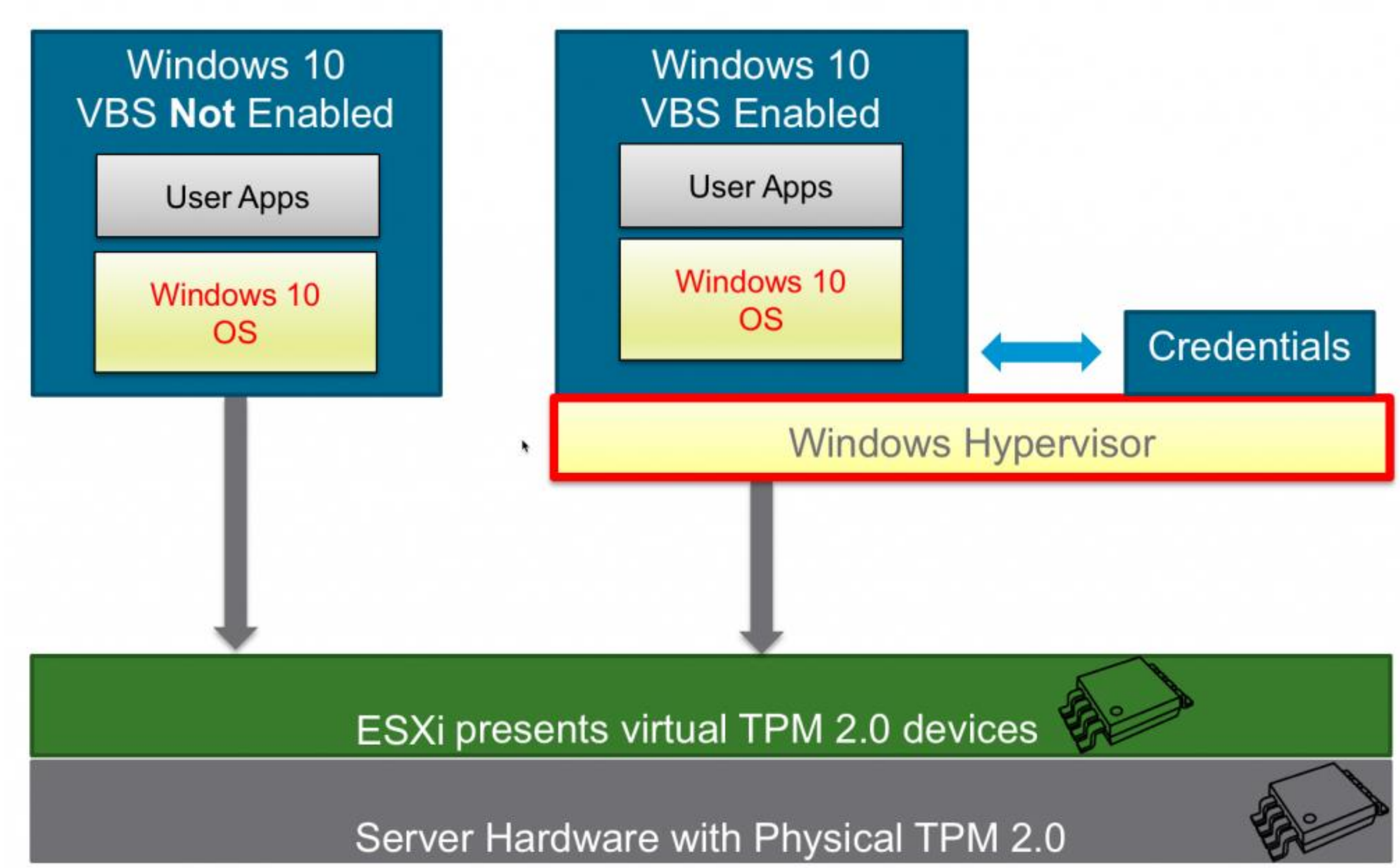
- We will assume that authentication is validated!

Windows 10

Virtualization-based security (VBS)


Normal World
Secure World
User space
App1
App2
Trusted services (e.g., encryption
Kernel space
Shared memory
NTOS
RPC
Secure Kernel (Shim layer)
Second Level Address Translation (1-1 without access)
Second Level Address Translation (1-1 with access)
Trusted Hyper-V

VBS in the (private) cloud



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

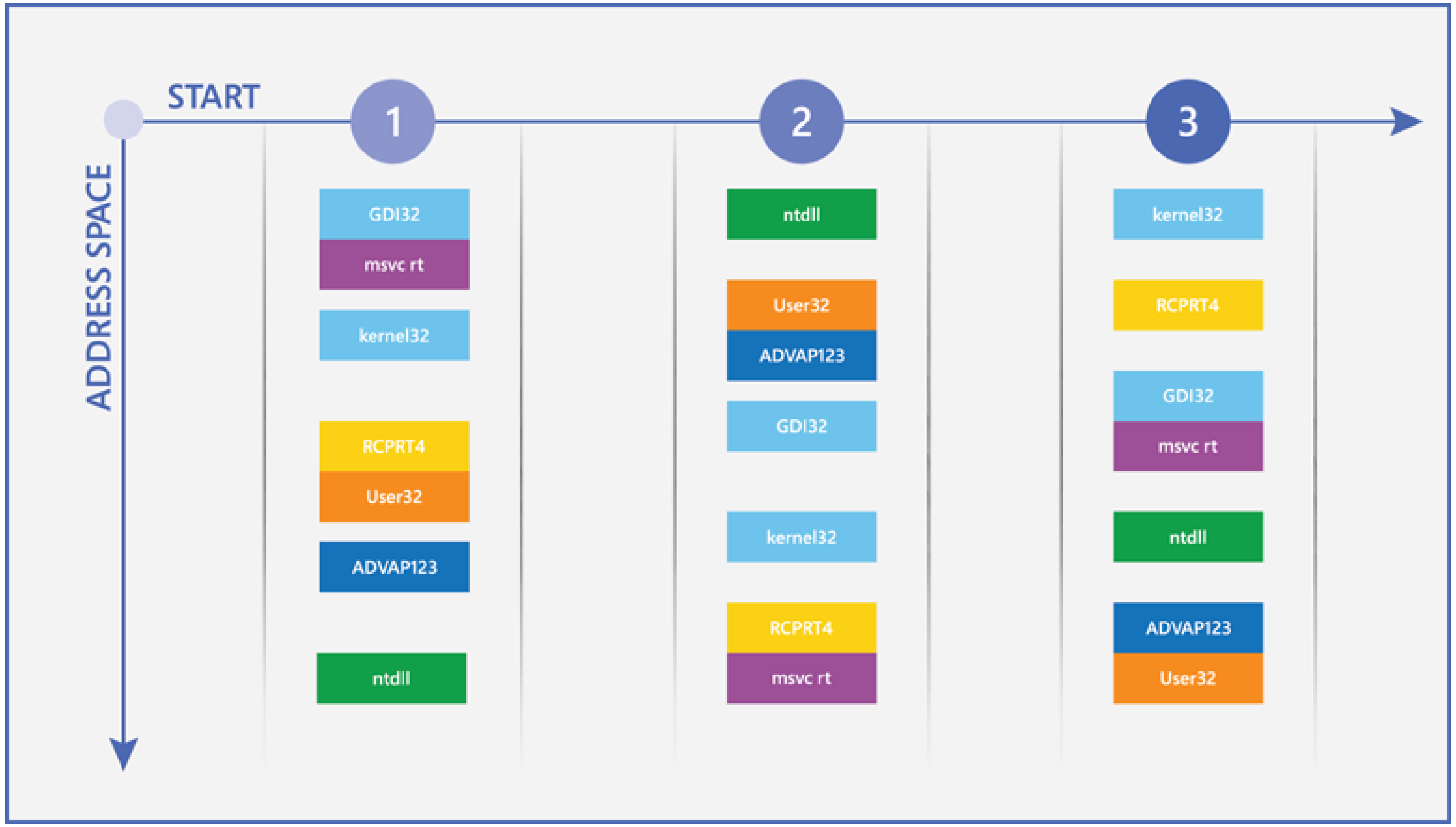
Protected Processes

- 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)

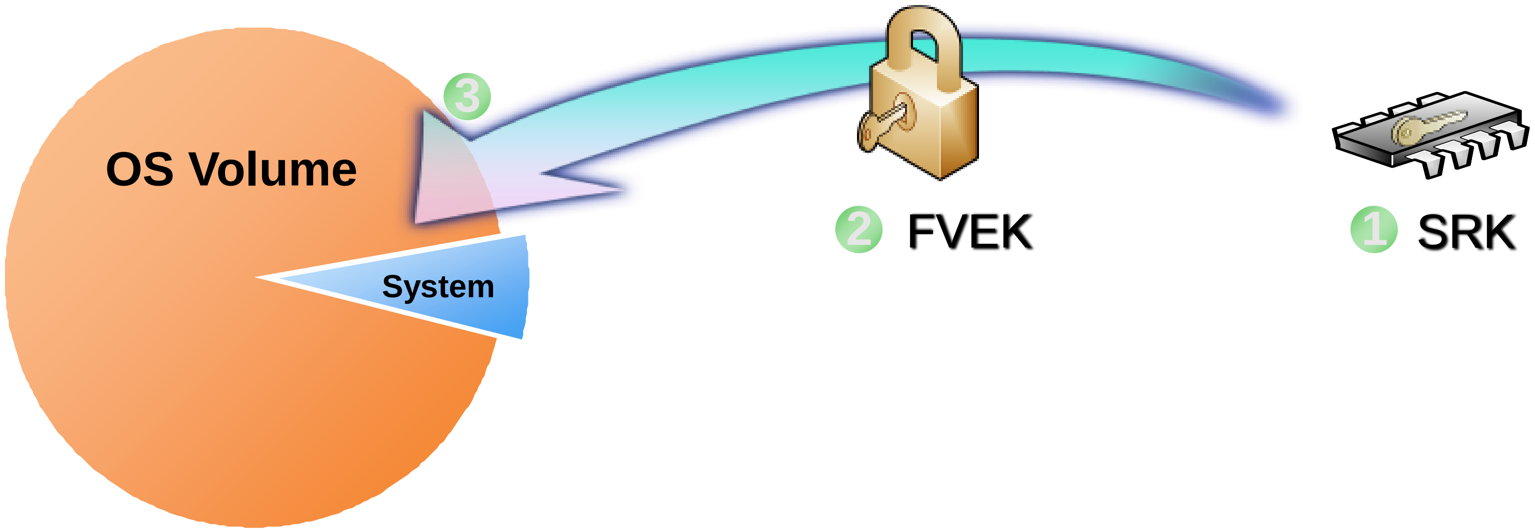
- DEP uses the No eXecute bit on modern CPUs
- Available on all major Oses
- Not real use if you can access mprotect/VirtualProtect/etc.

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.

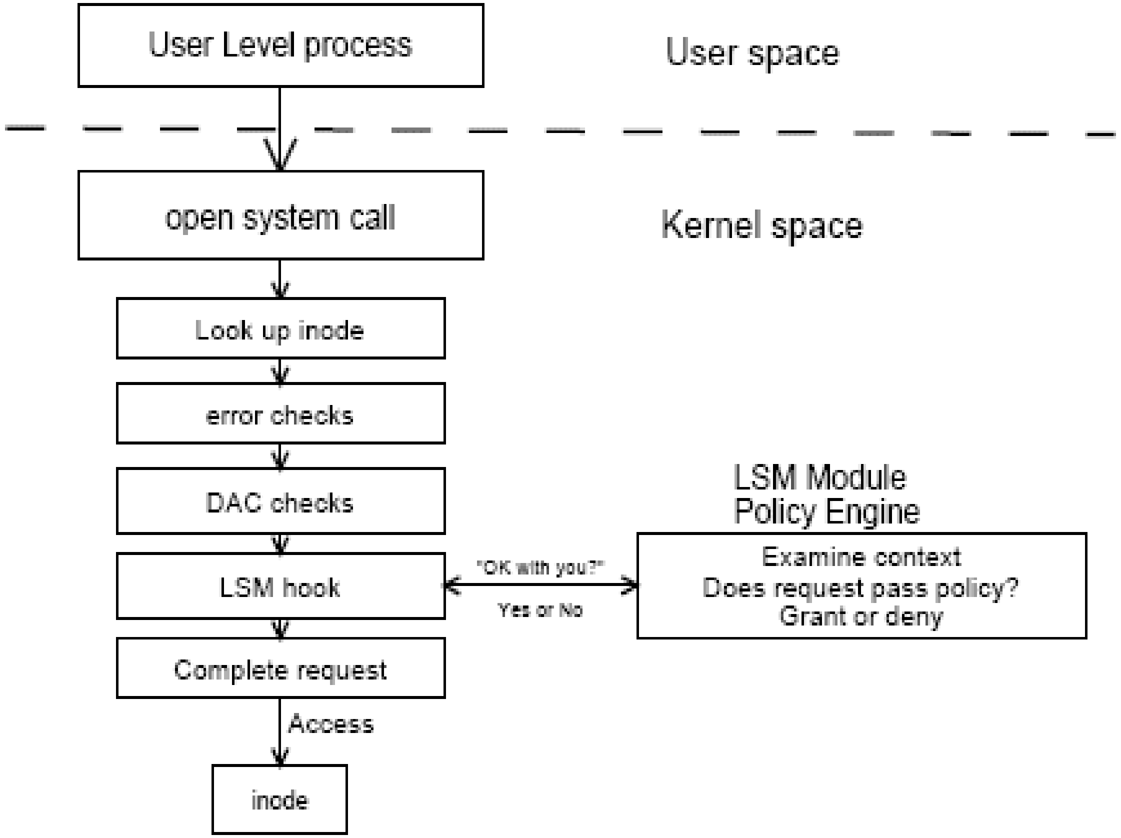
LUKS – Full-disk encryption [3]

- 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
- Labels are in the format user:role:type:level (level is optional)

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 } ;
    ```

Apparmor

- Mandatory Access Control (MAC)
- Per path profile
- Enforcement and complain mode

Apparmor

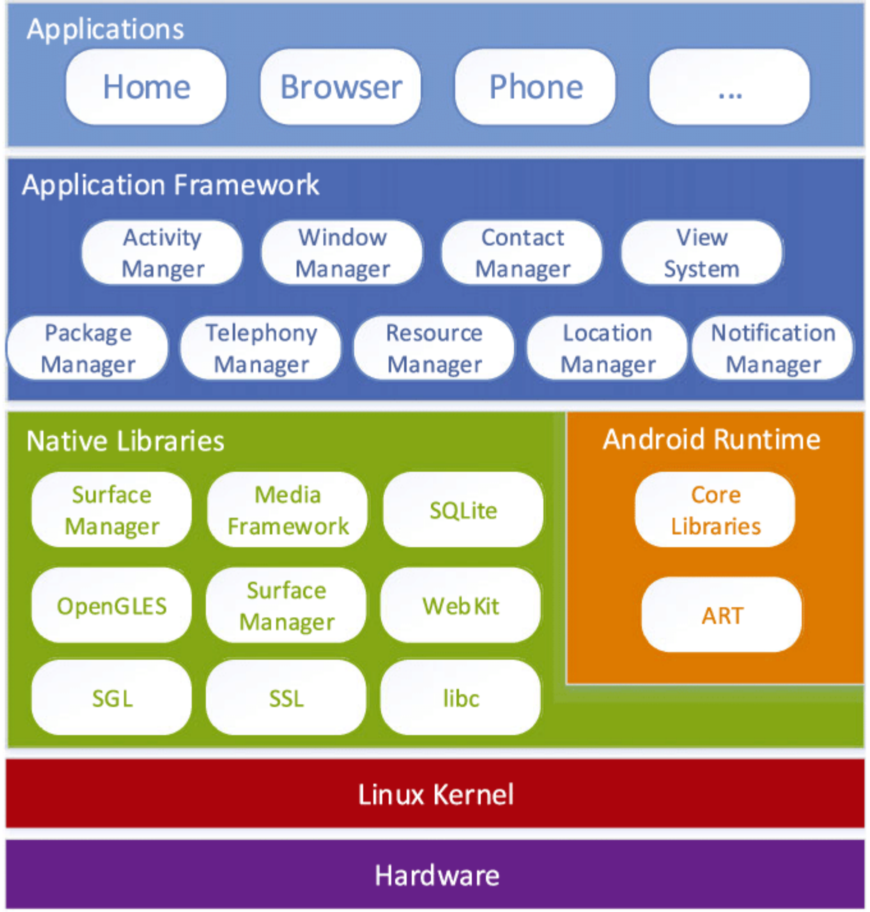
From /etc/apparmor.d/usr.sbin.tcpdump on Ubuntu 9.04:

```
/usr/sbin/tcpdump {
  #include <abstractions/user-tmp>
  capability setuid,
  network raw,
  network packet,
  @{PROC}/bus/usb/ r,
  @{PROC}/bus/usb/** r,

  audit deny @{HOME}/bin/ rw,
  audit deny @{HOME}/bin/** mrwkl,
  @{HOME}/ r,
  /usr/sbin/tcpdump r,
}
```

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);

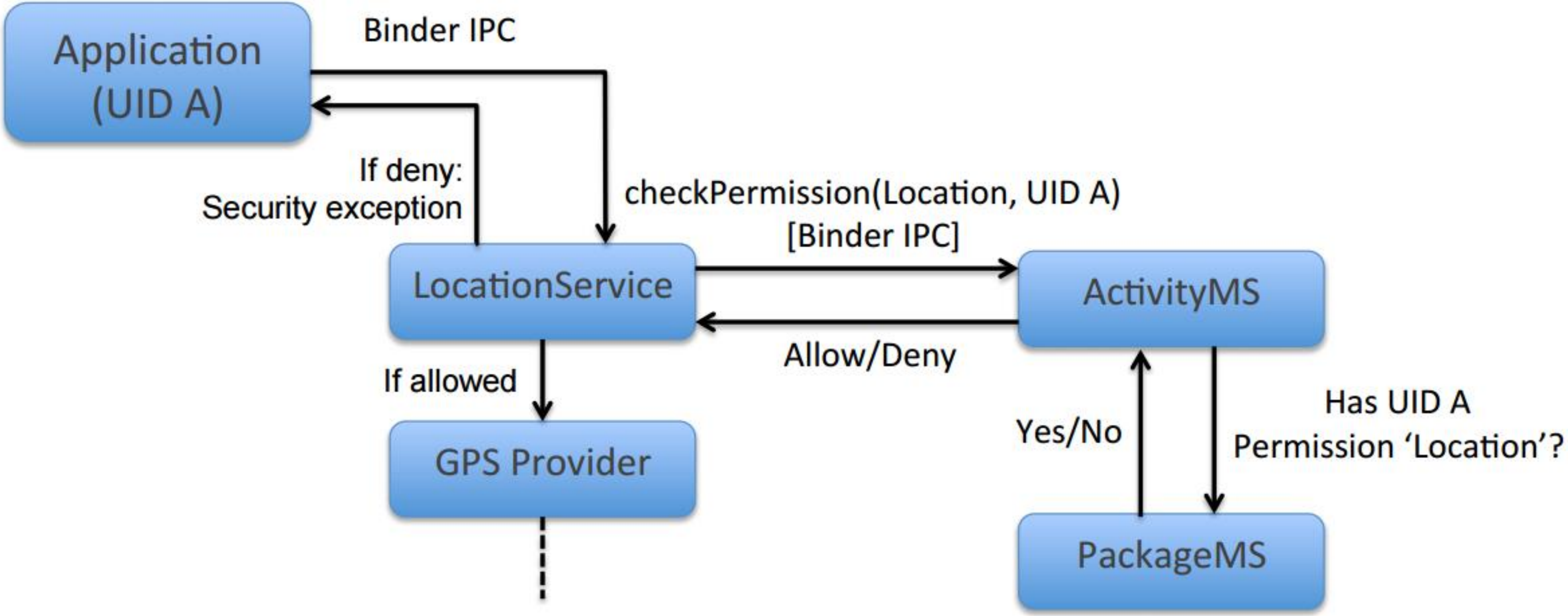
Android Security Basics

- 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

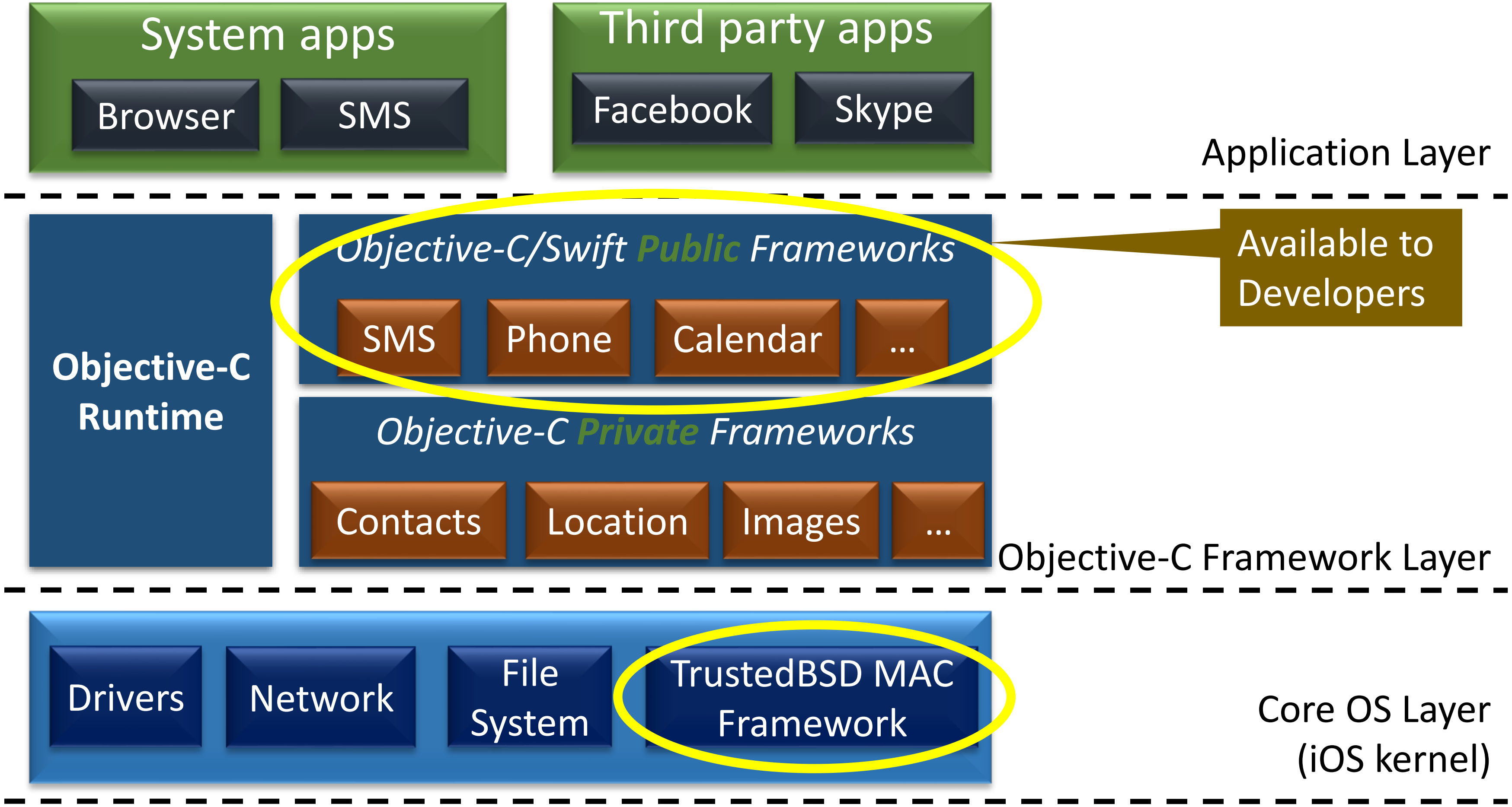
- App UID must be member of a Linux group to have access to sockets, etc.
- UID of an app with corresponding permission is added to group during install
- Kernel access errors translated into Java security exceptions by core libraries

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)

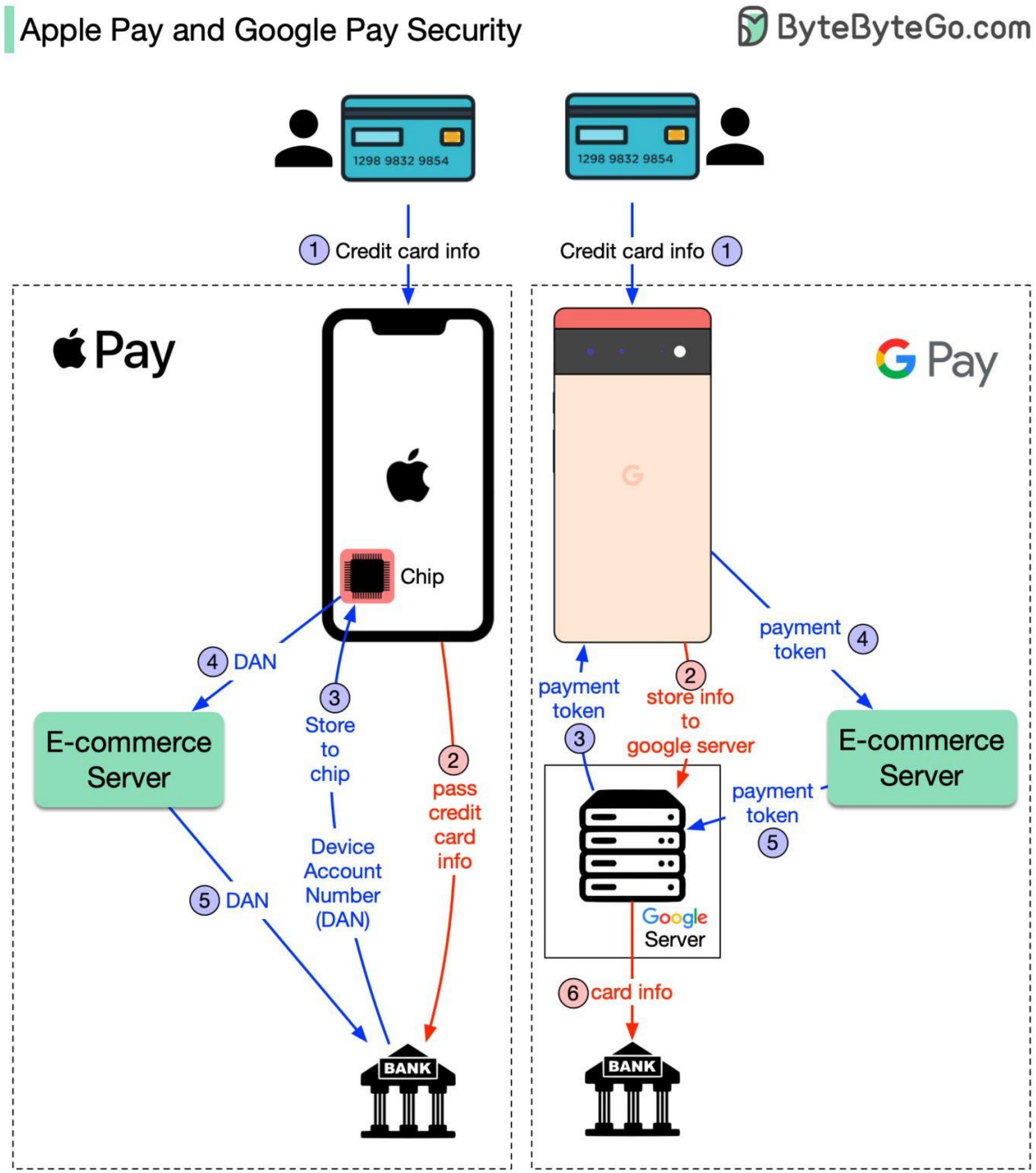
Sandboxing

- 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

Apple Pay and Google Pay Security



https://blog.bytebytego.com/i/74750876/how-do-apple-pay-and-google-pay-handle-sensitive-card-info

Hypervisor security

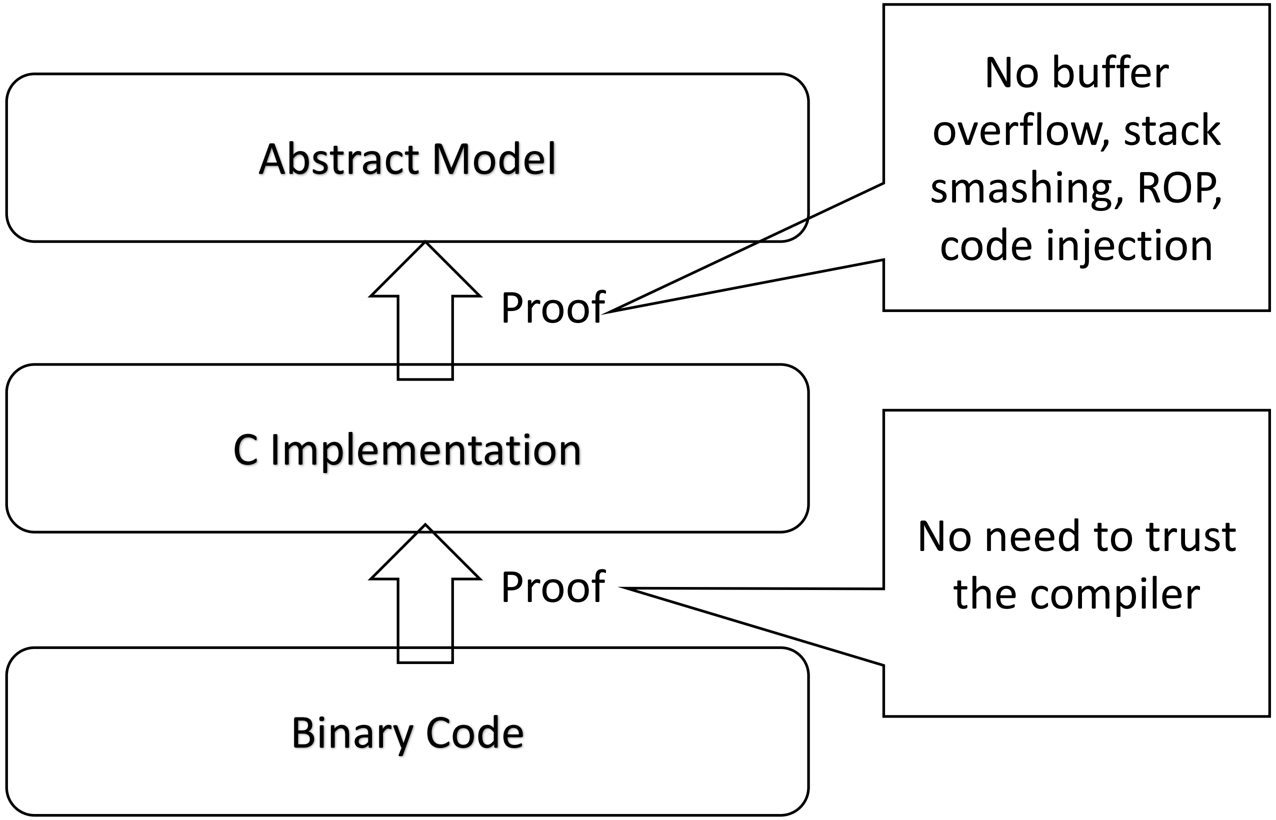
Security possibilities

- 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.


Abstract Model
Proof
C Implementation
Proof
Binary Code
No buffer overflow, stack smashing, ROP, code injection
No need to trust the compiler

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
- [12] https://doi.org/10.1002/cpe.4180