RANDORISEC

SOME CRACKS IN THE LINUX FIREWALL

ARTHUR MONGODIN

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

About me

- Arthur Mongodin - *@_Aleknight_*
  - Security researcher at RandoriSec - *@RandoriSec*
  - Passionate about reverse engineering and binary exploitation
  - GreHack organization committee
- End of study internship at RandoriSec
  - Variant analysis on Linux kernel

RANDORISEC

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

Variant analysis

- Vulnerabilities are similar
- Pattern matching
- Some tools
  - *CodeQL*
  - *Joern*
  - *Coccinelle*

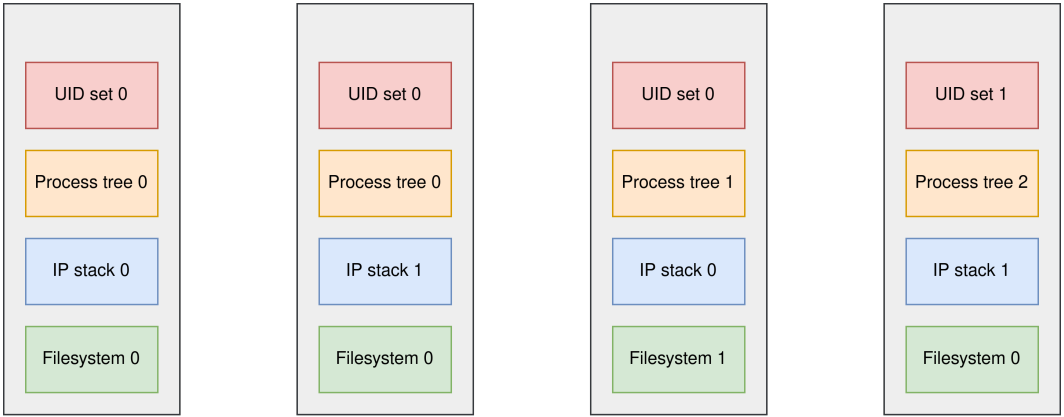
RANDORISEC

Linux namespaces

- Use to create separate environment for applications
- Different kind of namespaces:
  - user
  - network
  - mount
  - PID
  - …
- Some applications
  - Containers
  - Sandbox

RANDORISEC

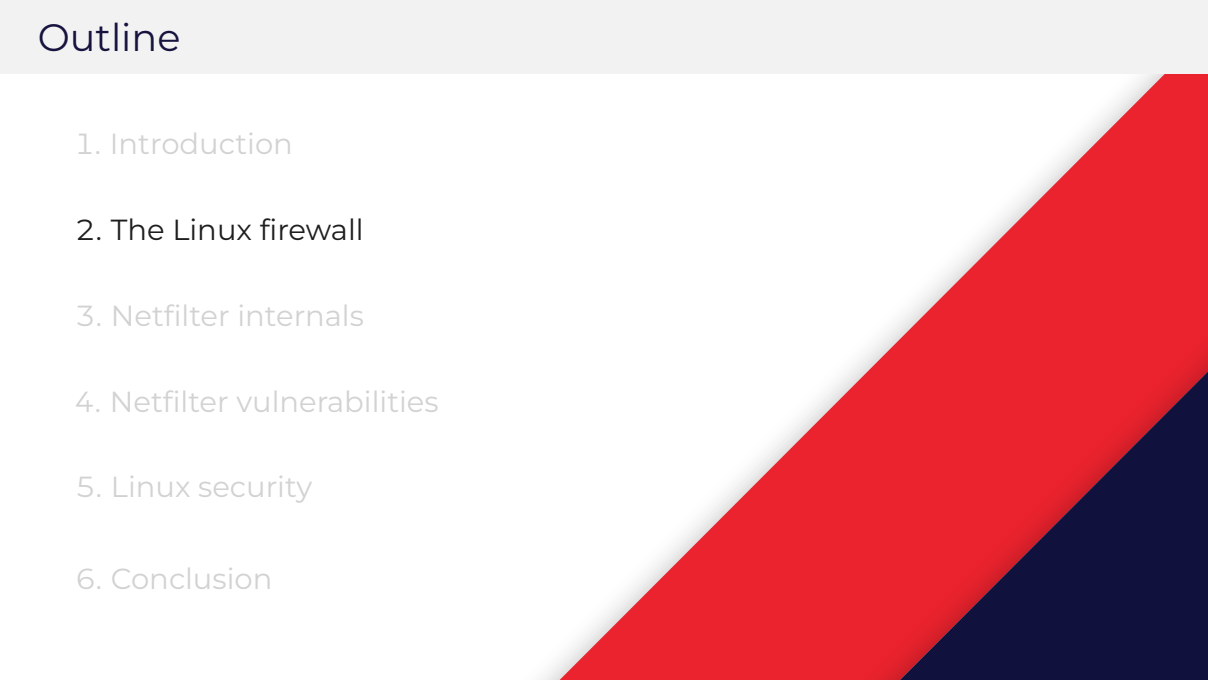
Linux namespaces



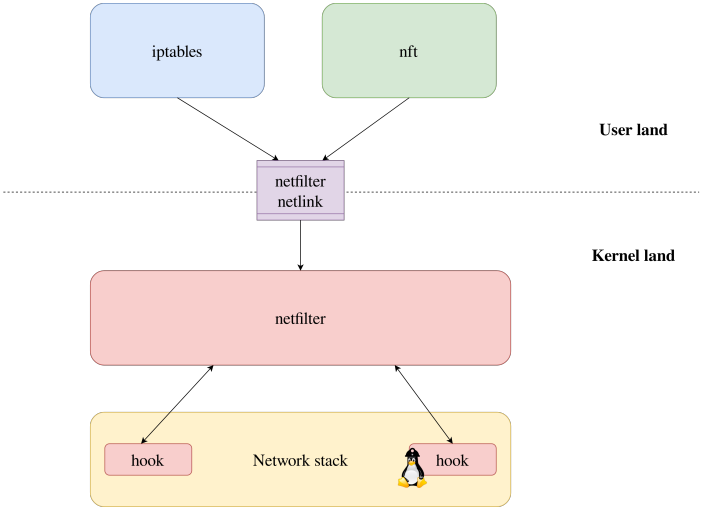
RANDORISEC

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion



Global organization



RANDORISEC

Operations

- Interactive shell nft
- Different target object
  - IP addresses
  - Port number
- Different actions
  - Drop
  - Accept

RANDORISEC

Outline

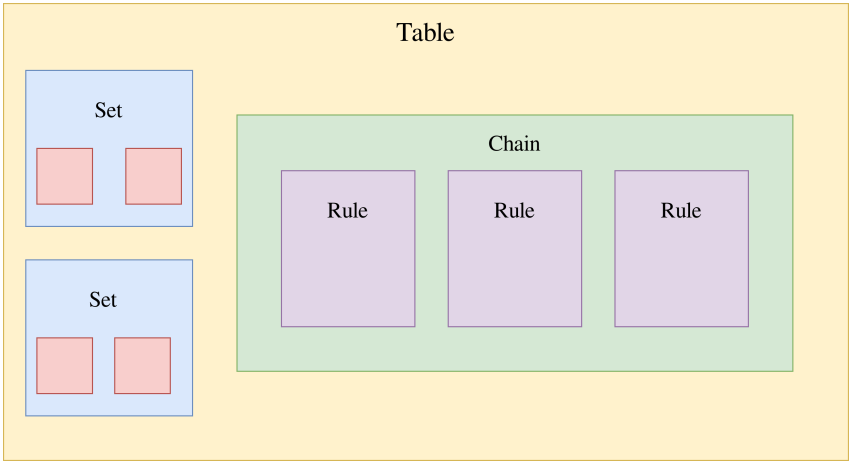
1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

User interface

- Reachable with unprivileged user namespaces
- Netlink socket (sendmsg, recvmsg)
- Message encapsulation: Netlink attributes

RANDORISEC

Object encapsulation



RANDORISEC

Expressions

- Used to match packets
- More than 40 types of expressions
- Linked to several kind objects
  - Sets
  - Maps
  - Rules

RANDORISEC

Verdict statements

- Instruction on packets
  - accept
  - drop
  - queue
  - continue
  - return
  - goto *chain*
  - jump *chain*

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

CVE-2022-1015 - Root Cause Analysis

- Found by D. Bouman
- Integer overflow in input validation check

```
1  static int nft_validate_register_load(
2                              enum nft_registers reg,
3                              unsigned int len) {
4      ...
5
6      if (reg * NFT_REG32_SIZE + len >
7          sizeof_field(struct nft_regs, data))
8          return -ERANGE;
9
10     return 0;
11 }
```

CVE-2022-1015 - Let's find some variants

- Use of *CodeQL*
- `if` statement
  - Multiplication in condition expression
  - Returns an error

RANDORISEC

CVE-2022-1015 - Let's find some variants

CodeQL class definition

```
class InputCheck extends IfStmt {
    InputCheck() {
        exists(ReturnStmt rs |
            this.getThen() = rs.getEnclosingStmt()
            and rs.getExpr().getValue().toInt() < 0)
    }
}
```

RANDORISEC

CVE-2022-1015 - Let's find some variants

CodeQL query

```
from InputCheck ic,
    RelationalOperation ro,
    MulExpr me
where ic.getCondition() = ro
    and ro.getGreaterOperand.getAChild*() = me
    and not me.isConstant()
select ic
```

RANDORISEC

CVE-2022-1015 - A variant

```
1  static int nft_set_desc_concat_parse(...)
2      u32 len;
3
4      ...
5
6      len = ntohl(nla_get_be32(tb[NFTA_SET_FIELD_LEN]));
7
8      if (len * BITS_PER_BYTE / 32 > NFT_REG32_COUNT)
9          return -E2BIG;
10
11     desc->field_len[desc->field_count++] = len;
12     return 0;
13 }
```

RANDORISEC

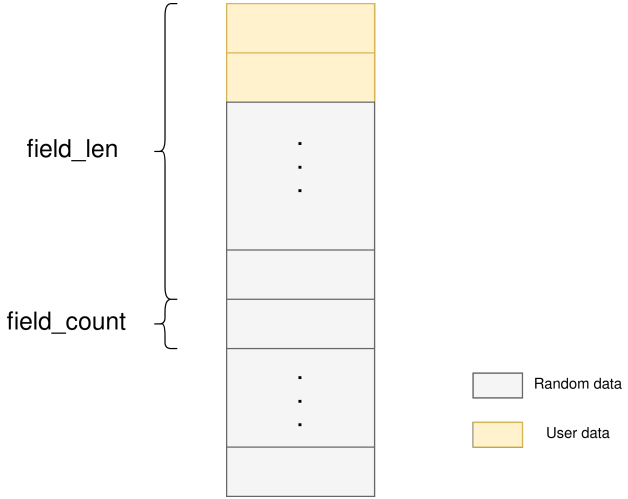
CVE-2022-2078 - The feature

- Set concatenation
  - Introduced in Linux 4.1
  - Better performance

```
1 set myset {
2     typeof ip saddr . tcp dport
3     elements = {
4         192.168.0.3 . 22,
5         192.168.0.4 . 80,
6     }
7 }
```

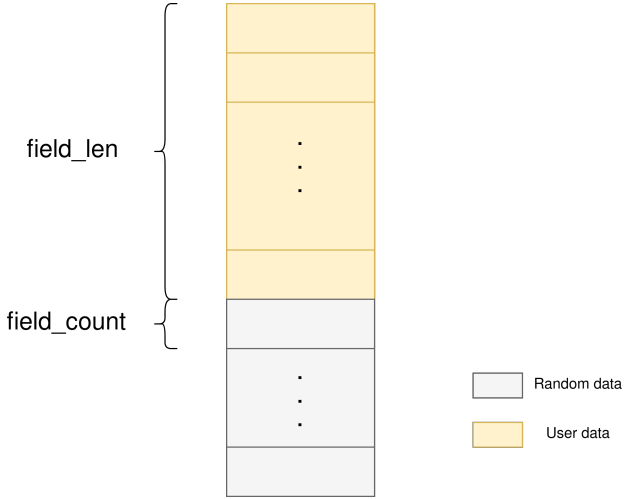
RANDORISEC

CVE-2022-2078 - Illustration



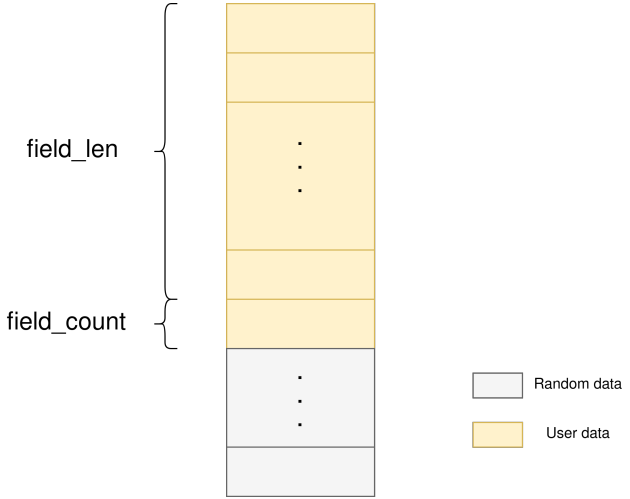
RANDORISEC

CVE-2022-2078 - Illustration



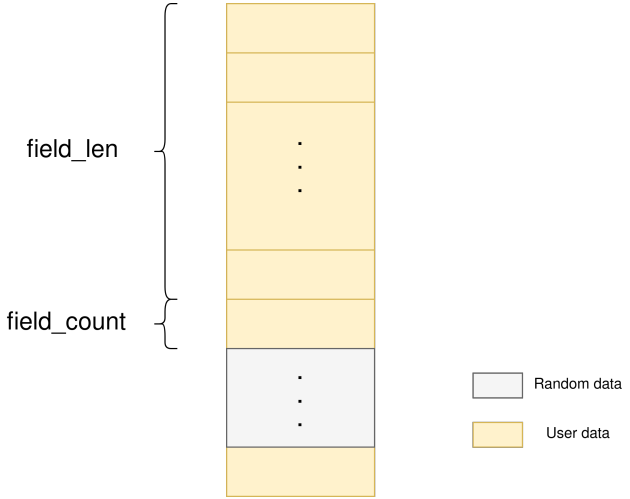
RANDORISEC

CVE-2022-2078 - Illustration



RANDORISEC

CVE-2022-2078 - Illustration



RANDORISEC

CVE-2022-2078 - Technical details

- Stack buffer overflow
- Stack Out-of-bound primitives
- Bad primitive
  - len lower than 64
  - In practice len lower than 67

RANDORISEC

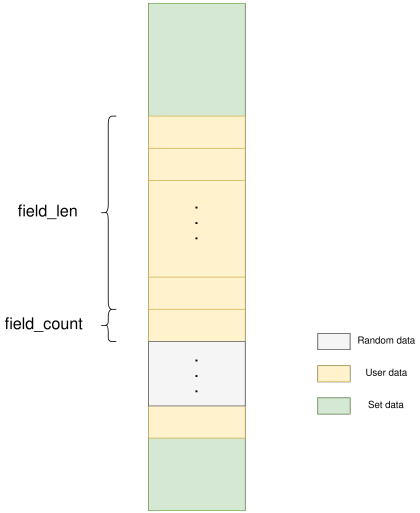
CVE-2022-2078 - Technical details

- Buffer copied within a nft_set structure

```
1  static int nf_tables_newset(...){
2      ...
3      set->field_count = desc.field_count;
4      for (i = 0; i < desc.field_count; i++)
5          set->field_len[i] = desc.field_len[i];
6      ...
7  }
```

RANDORISEC

CVE-2022-2078 - Infoleak



RANDORISEC

CVE-2022-2078 - Infoleak

Demo time !

RANDORISEC

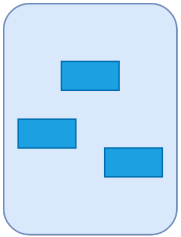
CVE-2022-34918 - The feature

- Verdict maps

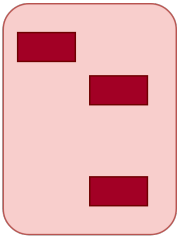
```
table ip mytable {
    set myset {
        type ipv4_addr
        elements = { 192.168.0.3, 192.168.0.4 }
    }
    map mymap {
        type ipv4_addr : verdict
        elements = { 192.168.0.3 : drop, 192.168.0.4 : accept }
    }
}
```

RANDORISEC

CVE-2022-34918



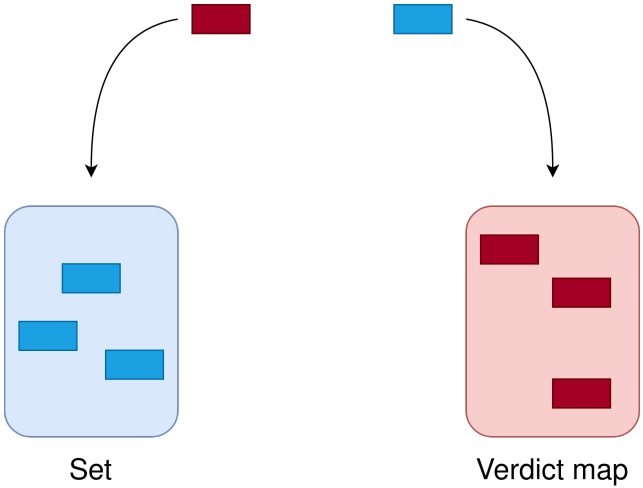
Set



Verdict map

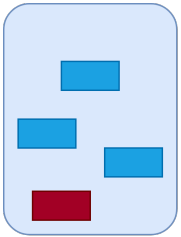
RANDORISEC

CVE-2022-34918

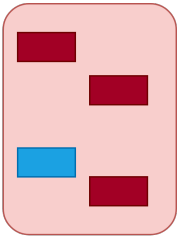


RANDORISEC

CVE-2022-34918



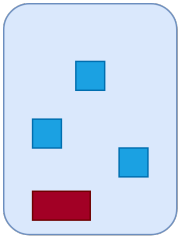
Set



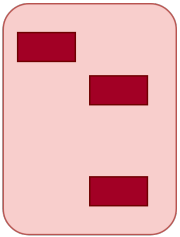
Verdict map

RANDORISEC

CVE-2022-34918



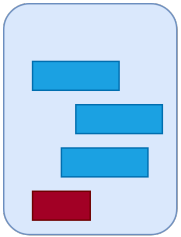
Set



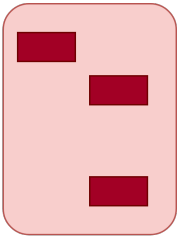
Verdict map

RANDORISEC

CVE-2022-34918



Set



Verdict map

RANDORISEC

CVE-2022-34918 - The overflow

- Root cause
  - Allocation size based on the element size
  - Copy size based on set settings
- Controlled with an uninitialized variable
- Several caches
- Can be extended up to 48 bytes

RANDORISEC

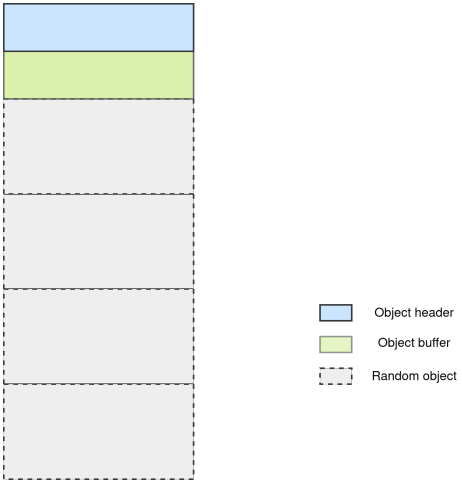
CVE-2022-34918 - infoleak

- addkey syscall

```
1 struct user_key_payload {
2     struct rcu_head rcu;        /* RCU destructor */
3     unsigned short  datalen;    /* length of this data */
4     char        data[] __aligned(__alignof__(u64)); /* actual
          data */
5 };
```

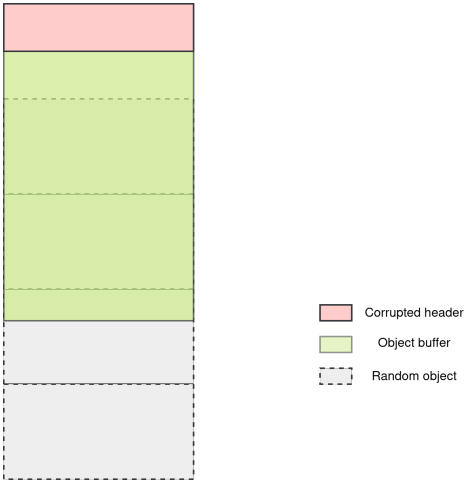
RANDORISEC

CVE-2022-34918 - infoleak



RANDORISEC

CVE-2022-34918 - infoleak



RANDORISEC

CVE-2022-34918 - Write primitive

- Unlinking attack
  - Write-up from L. J. Rong
  - Use of list_del
- Edit modprobe_path to gain root privileges

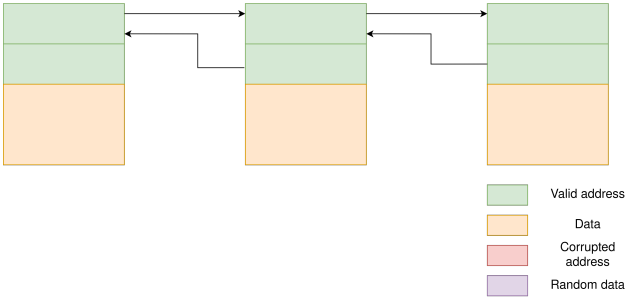
RANDORISEC

CVE-2022-34918 - Write primitive

```
1 static inline void __list_del(struct list_head * prev,
    struct list_head * next)
2 {
3     next->prev = prev;
4     WRITE_ONCE(prev->next, next);
5 }
```

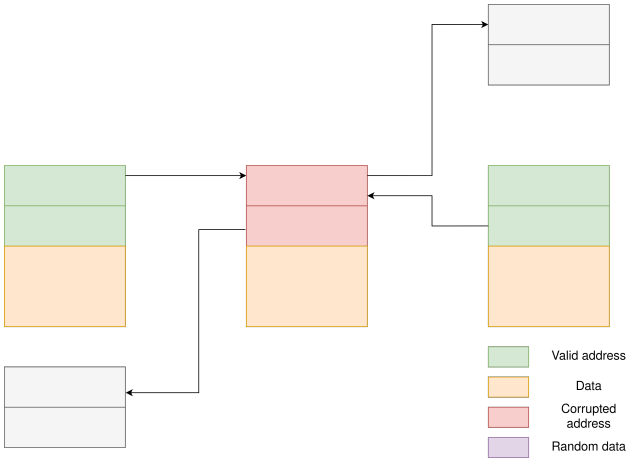
RANDORISEC

CVE-2022-34918 - Unlinking attack



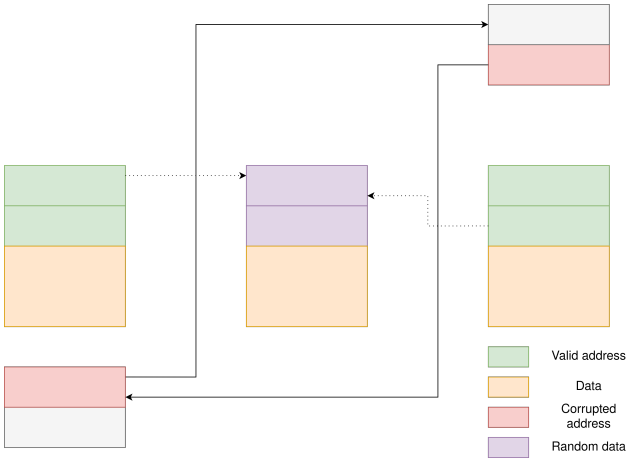
RANDORISEC

CVE-2022-34918 - Unlinking attack



RANDORISEC

CVE-2022-34918 - Unlinking attack



RANDORISEC

CVE-2022-34918 - Final result

Demo time !

RANDORISEC

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

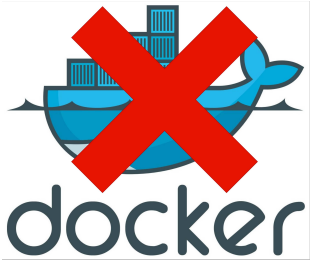
Hard time for Netfilter tables

- CVE-2022-1015: Integer overflow (D. Bouman)
- CVE-2022-1016: Uninitialized variable (D. Bouman)
- CVE-2022-2078: Buffer overflow (Ant Group Light-Year Security Lab, Sea Security and me ;-))
- CVE-2022-2586: Use-after-free (Sea Security)
- CVE-2022-25636: Heap out-of-bound write (N. Gregory)
- CVE-2022-32250: Use-after-free (NCC Group)
- CVE-2022-34918: Type confusion (Also me ;-))
- CVE-2022-39190: Uncontrolled Resource Consumption (G. Jung)

RANDORISEC

System hardening

- Disable unprivileged user namespaces
- Do not use containters
- Use *LKRG*
- Test your configuration with *kconfig-hardened-check*



RANDORISEC

Outline

1. Introduction
2. The Linux firewall
3. Netfilter internals
4. Netfilter vulnerabilities
5. Linux security
6. Conclusion

Conclusion

- Variant analysis efficient to find bugs
- Unprivileged user namespaces are increasing kernel attack surface
- Netfilter great attack surface
- Metasploit module (by *@red0xff*)
- Technical write-ups on our website
  - *CVE-2022-2078*
  - *CVE-2022-34918*
- Exploit sources available on GitHub
  - *CVE-2022-2078*
  - *CVE-2022-34918*

RANDORISEC

RANDORISEC

THANKS!

DO YOU HAVE ANY QUESTIONS?