Dereference Under the Influence (DUI)  
You Can’t Afford It

Hong Hu, Zheng Leong Chua, Prateek Saxena, Zhenkai Liang

Problem Introduction

- Security-critical components are often protected using isolation mechanisms
  > Interactions via API interfaces
- Attackers can affect the protected component by input to interfaces
  > Data values
  > Memory addresses
- We call memory dereference affected by attackers Dereference Under the Influence (DUI).

DUI Detector: An automatic tool to detect DUI

Types of DUI

- Write DUI: memory writing operation
  ```
  v1 = API_recv();
  v2 = API_recv();
  array[v1] = v2;
  ```
- Read DUI: memory read operation
  ```
  v1 = API_recv();
  data = *(base+v1);
  API_send(data);
  ```
  * API_recv() receives data from outside
  * API_send() sends data to outside

DUI in glibc

Setup the heap region:
```
addr1 = brk(arg1)
addr2 = brk(arg2)
*(addr1 + 4) = addr2 - addr1
```

Corresponding inst.:
```
mov %eax, 0x4(%edi)
```

Detected DUIs

- **Condition**: (brk1 %8 == 0 && brk2 > brk1)
  ```
  address = brk1 + 0x2718;
  data = (brk2 - brk1 - 0 x2718) | 0x1;
  ```
- **Condition**: (brk1 %8 != 0 && brk1 < brk2
  ```
  *(addr1 + 4) = addr2 - addr1
  ```
- **Condition**: (brk1 %8 != 0 && brk1 < brk2
  ```
  *(addr1 + 4) = addr2 - addr1
  ```

mmap2 system call

Map files or devices into memory
```
mov %eax, 0x1ac(%edi)
```

Conclusion

- Attackers can influence memory operations of isolated components through inputs to their public interfaces.
- We present DUI Detector, an automatic tool to detect dereference under the influence (DUI) through memory access patterns in execution traces.

This work is supported by the Ministry of Education, Singapore under Grant No. R-252-000-495-133.

Some pictures courtesy of http://icons8.com/