Integrating DMA attacks in exploitation frameworks

Rory Breuk Albert Spruyt

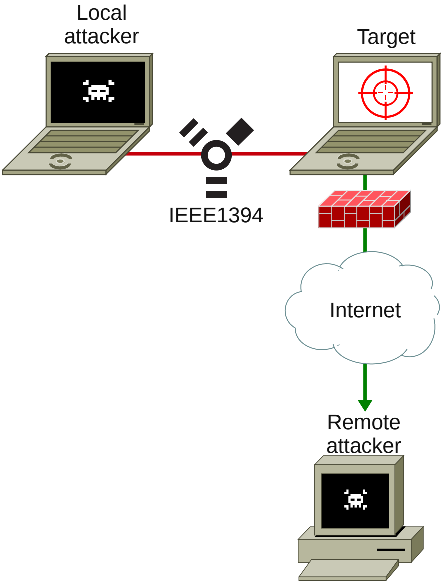
University of Amsterdam

February 7, 2012

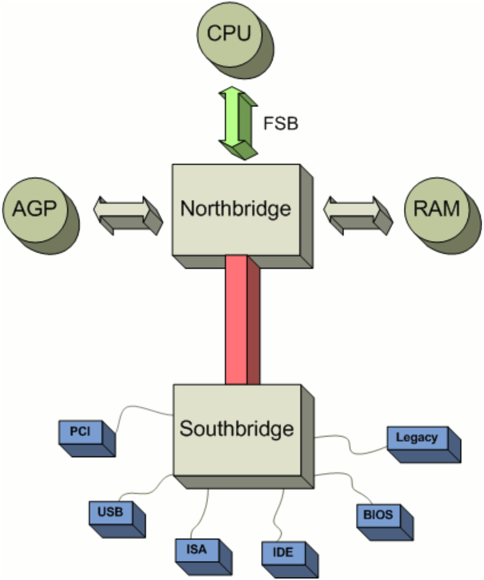
Introduction

- Research Question:
  How can DMA attacks be integrated into an exploitation framework?
- Previous work
  - FTWAutopwn
  - `libforensic1394`
  - Payloads
- Why?
  - Huge potential, but under utilized
  - Widespread awareness is lacking
  - Making it easy
  - Different from buffer overflows
  - Lots of possibilities

Usecase



Computer architecture



DMA - protocol analysis

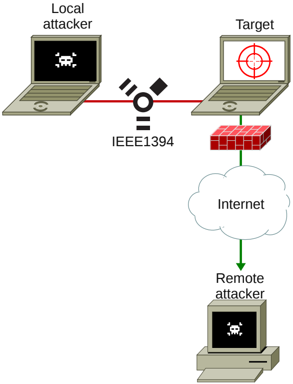
- FireWire
- eSATA
- USB - On The Go
- Thunderbolt
- PCMCIA

Exploitation frameworks

- Core Impact
- Metasploit Framework
- CANVAS
- Volatility

Metasploit concepts

- Exploits
- Payloads
- Sessions

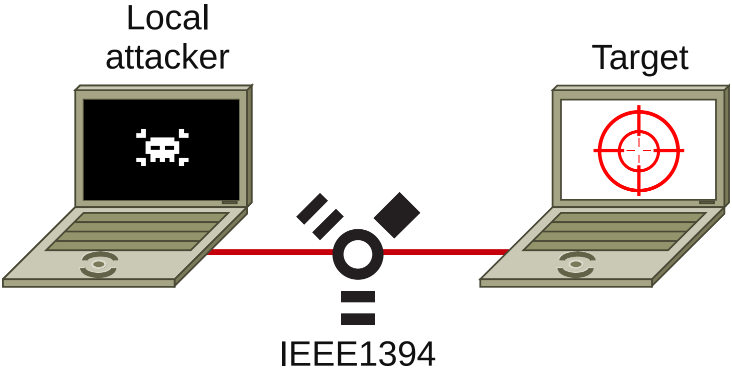


Integration

- `libforensic1394`
- Inserting code
- Metasploit reverse shell
- Cleaning up
- FireWire data connection

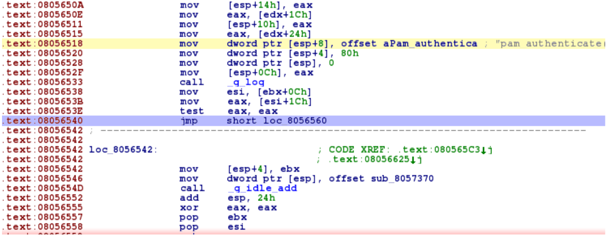
Userspace FireWire data connection - DEMO

- Runs in userspace
- Injectable
- Cache coherency



Payloads

What to patch

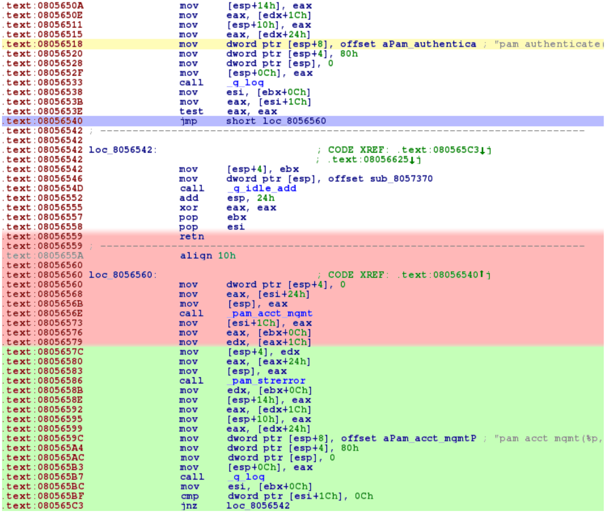


```
.text:0805650A                 mov     [esp+14h], eax
.text:0805650E                 mov     eax, [edx+1Ch]
.text:08056511                 mov     [esp+10h], eax
.text:08056515                 mov     eax, [edx+24h]
.text:08056518                 mov     dword ptr [esp+8], offset aPam_authentica ; "pam authenticate
.text:08056520                 mov     dword ptr [esp+4], 80h
.text:08056528                 mov     dword ptr [esp], 0
.text:0805652F                 mov     [esp+0Ch], eax
.text:08056533                 call    _g_log
.text:08056538                 mov     esi, [ebx+0Ch]
.text:0805653B                 mov     eax, [esi+1Ch]
.text:0805653E                 test    eax, eax
.text:08056540                 jmp     short loc_8056560
.text:08056542 ; ---------------------------------------------------------------------------
.text:08056542
.text:08056542 loc_8056542:                            ; CODE XREF: .text:080565C3↓j
.text:08056542                                         ; .text:08056625↓j
.text:08056542                 mov     [esp+4], ebx
.text:08056546                 mov     dword ptr [esp], offset sub_8057370
.text:0805654D                 call    _g_idle_add
.text:08056552                 add     esp, 24h
.text:08056555                 xor     eax, eax
.text:08056557                 pop     ebx
.text:08056558                 pop     esi
```

Library call

Patch

Clean up - Act normal



```
.text:0805650A                 mov     [esp+14h], eax
.text:0805650E                 mov     eax, [edx+1Ch]
.text:08056511                 mov     [esp+10h], eax
.text:08056515                 mov     eax, [edx+24h]
.text:08056518                 mov     dword ptr [esp+8], offset aPam_authentica ; "pam_authenticate
.text:08056520                 mov     dword ptr [esp+4], 80h
.text:08056528                 mov     dword ptr [esp], 0
.text:0805652F                 mov     [esp+0Ch], eax
.text:08056533                 call    _g_log
.text:08056538                 mov     esi, [ebx+0Ch]
.text:0805653B                 mov     eax, [esi+1Ch]
.text:0805653E                 test    eax, eax
.text:08056540                 jmp     short loc_8056560
.text:08056542 ; ---------------------------------------------------------------------------
.text:08056542
.text:08056542 loc_8056542:                            ; CODE XREF: .text:080565C3↓j
.text:08056542                                         ; .text:08056625↓j
.text:08056542                 mov     [esp+4], ebx
.text:08056546                 mov     dword ptr [esp], offset sub_8057370
.text:0805654D                 call    _g_idle_add
.text:08056552                 add     esp, 24h
.text:08056555                 xor     eax, eax
.text:08056557                 pop     ebx
.text:08056558                 pop     esi
.text:08056559                 retn
.text:08056559 ; ---------------------------------------------------------------------------
.text:0805655A                 align 10h
.text:08056560
.text:08056560 loc_8056560:                            ; CODE XREF: .text:08056540↑j
.text:08056560                 mov     dword ptr [esp+4], 0
.text:08056568                 mov     eax, [esi+24h]
.text:0805656B                 mov     [esp], eax
.text:0805656E                 call    _pam_acct_mgmt
.text:08056573                 mov     [esi+1Ch], eax
.text:08056576                 mov     eax, [ebx+0Ch]
.text:08056579                 mov     edx, [eax+1Ch]
.text:0805657C                 mov     [esp+4], edx
.text:08056580                 mov     eax, [eax+24h]
.text:08056583                 mov     [esp], eax
.text:08056586                 call    _pam_strerror
.text:0805658B                 mov     edx, [ebx+0Ch]
.text:0805658E                 mov     [esp+14h], eax
.text:08056592                 mov     eax, [edx+1Ch]
.text:08056595                 mov     [esp+10h], eax
.text:08056599                 mov     eax, [edx+24h]
.text:0805659C                 mov     dword ptr [esp+8], offset aPam_acct_mgmtP ; "pam_acct_mgmt(%p,
.text:080565A4                 mov     dword ptr [esp+4], 80h
.text:080565AC                 mov     dword ptr [esp], 0
.text:080565B3                 mov     [esp+0Ch], eax
.text:080565B7                 call    _g_log
.text:080565BC                 mov     esi, [ebx+0Ch]
.text:080565BF                 cmp     dword ptr [esi+1Ch], 0Ch
.text:080565C3                 jnz     loc_8056542
```

Library call

Patch

Fork

Payload

Metasploit demo

- Choose exploit and payload
- Change the settings for the modules
- Run exploit
  - Load payload into target
  - Depending on payload: achieve session between target and attacker

Mitigation

- Mitigation for end-users
- Don't buy them
- Destroy them / glue them
- Disable them
- Deny physical access

Conclusion

- Achievements:
    - Show DMA vulnerabilities exist on different ports
    - Port `libforensic1394` bindings to Ruby
    - Integrate FireWire exploit into Metasploit
    - Clean payload execution
    - Proof of concept FireWire data session

Questions?