CALDERA

Automating Adversary Emulation

Andy Applebaum, Doug Miller

The MITRE Corporation
Bios

Andy Applebaum
Lead Cyber Security Engineer
ATT&CK, AI Research

Doug Miller
Sr. Cyber Security Engineer
ATT&CK, CARET, Red teaming
Adversary Emulation?

About 654 results (0.38 seconds)

PDF adversary emulation - CrowdStrike
https://www.crowdstrike.com/wp-content/.../CS_AdversaryEmulation_datasheet.pdf

ADVERSARY EMULATION. CrowdStrike's premier Red Team service offering helps organizations gauge their readiness to withstand an attack from the most.

References on Adversary Simulations | Strategic Cyber LLC
https://blog.cobaltstrike.com/2015/03/12/references-on-adversary-simulations/
Mar 12, 2015 - I like this article because it discusses why adversary emulation is important, it makes a fair argument about why pen testing [still valuable] isn't a ...
The False Negative Problem

As a defender, you have no idea what you miss
Cue: Adversary

- Introduce a realistic* adversary on your network
  *an emulated adversary

- Now you can determine what happens if an attacker gets on your network
  - Did I detect them?
  - How far did they get?
  - How can I improve my detection and prevention?
Iterative Defensive Cycle

Emulate Adversary

CALDERA

You

Improve Defenses

Introspect
Successful Adversary Emulation

**Make it real:** Use the same techniques, tools, methods and goals of an attacker.

**End-to-End:** Don’t just look for holes or perform small attacks. Start from the initial compromise and go until objectives are accomplished.

**Repeatable:** Be repeatable, so that your detection and prevention improvement (or degradation) can be measured over time.
CALDERA:
CALDERA – Conducting an Operation

1. Load the CALDERA shim onto network hosts
2. Create an adversary by giving it capabilities
3. Launch the operation

- **During the operation:**
  - CALDERA will report its activities, including artifacts created
  - CALDERA will automatically stop if it has exhausted its toolkit

- **After the operation:**
  - CALDERA will provide a report of what it did
  - CALDERA will automatically “reset” infected hosts, removing artifacts and stopping processes
Ingredients for Automated Adversary Emulation

What the adversary can do
• The adversary model
• MITRE ATT&CK™

How the adversary chooses what to do
• CALDERA logic and decision engine

What the adversary needs to do it
• Infrastructure to support real adversary emulation
• Management server; client agents; web interface
The Adversary Model
Choosing an Adversary Model

CALDERA emulates a real adversary after they get into a network
## ATT&CK Matrix™ Tactics and Techniques

### Table: Tactics and Techniques

<table>
<thead>
<tr>
<th>Persistence</th>
<th>Privilege Escalation</th>
<th>Defense Evasion</th>
<th>Credential Access</th>
<th>Discovery</th>
<th>Lateral Movement</th>
<th>Execution</th>
<th>Collection</th>
<th>Exfiltration</th>
<th>Command and Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLL Search Order Hijacking</td>
<td>Brute Force</td>
<td>Application Discovery</td>
<td>Windows Remote Management</td>
<td>Audio Capture</td>
<td>Automated Exfiltration</td>
<td>Commonly Used Port</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Legitimate Credentials</td>
<td>Credential Dumping</td>
<td>Application Window Discovery</td>
<td>Third-party Software</td>
<td>Command-Line</td>
<td>Automated Collection</td>
<td>Communication Through Removable Media</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accessibility Features</td>
<td>Binary Padding</td>
<td>File and Directory Discovery</td>
<td>Data Compressed</td>
<td>Data Staged</td>
<td>Data Transfer Size Limits</td>
<td>Connection Proxy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AppInit DLLs</td>
<td>Credential Manipulation</td>
<td>Exploitation of Vulnerability</td>
<td>Data from Local System</td>
<td>Data from Network Shared Drive</td>
<td>Exfiltration Over Alternative Protocol</td>
<td>Custom Command and Control Protocol</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code Signing</td>
<td>File System Permissions Weakness</td>
<td>Execution through Module Load</td>
<td>Data from Removable Media</td>
<td>Exfiltration Over Command and Control Channel</td>
<td>Custom Cryptographic Protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local Port Monitor</td>
<td>Component Firmware</td>
<td>Malicious Code Injection</td>
<td>Data Encoding</td>
<td>Data Obfuscation</td>
<td>Exfiltration Over Other Network Medium</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Firmware</td>
<td>Credentials in Files</td>
<td>Two-Factor Authentication</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Service</td>
<td>Local Network Configuration Discovery</td>
<td>Interception</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLL Side-Loading</td>
<td>Logon Scripts</td>
<td>Data Staged</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File System Permissions Weakness</td>
<td>Network Sniffing</td>
<td>Data from Removable Media</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Registry Permissions Weakness</td>
<td>Two-Factor Authentication</td>
<td>Exfiltration Over Command and Control Channel</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Control</td>
<td>InstallUtil</td>
<td>Exfiltration Over Physical Medium</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Shell</td>
<td>Indicator Blocking</td>
<td>Multi-Stage Channels</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Shell Injection</td>
<td>Exploitation of Vulnerability</td>
<td>Multi-Stage Channels</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td>Data Obfuscation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Authentication Package

- Bypass User Account Control
- Exfiltration Over Command and Control Channel

### BookKit

- DLL Injection
- Component Object Model Hijacking
- Indicator Removal from Tools

### Basic Input/Output System

- Component Object Model Hijacking
- Indicator Removal on Host

### Change Default File Association

- Install Root Certificate
- InstallUtil

### Component Firmware

- Masquerading
- Modify Registry

### External Remote Services

- Network Share Removal
- NTFS Extended Attributes

### Hypervisor

- NTFS Extended Attributes
- Obfuscated Files or Information

### Logon Scripts

- Process Hollowing
- Redundant Access

### Modify Existing Service

- Regsvr32
- Rundll32

### Netsh Helper DLL

- Scripting
- Software Packing

### Redundant Access

- TimeStamp

### Registry Run Keys / Start Folder

- Regsvr32
- Rundll32

### Security Support Provider

- Scripting
- Software Packing

### Shortcut Modification

- TaintStamp

### Windows Management Instrumentation Event Subscription

- TaintStamp

### Winlogon Helper DLL

- TaintStamp

---

[https://attack.mitre.org](https://attack.mitre.org)
Supported Adversary Actions

**Persistence**
- Registry autorun keys
- Scheduled Task
- Services

**Privilege Escalation**
- Weak service perms
- Weak service file perms
- Unquoted paths (Path interception)

**Defense Evasion**
- Scripting
- Timestomping

**Credential Access**
- Credential Dumping

**Lateral Movement**
- Remote File Copy
- Windows Admin shares
- Pass the Hash
- PsExec

**Discovery**
- Remote System Discovery
- Local Network config
- Registry
- Account
- System information
- Processes/services
- System Owner
- Permission Group
- Files

**Execution**
- PowerShell
- Scheduled Task
- WMI
- SC (service controller)

**Exfiltration**
- HTTP/S
Decision Making for Automated Adversary Emulation
Early CALDERA

- **First version**
  - Finite-state machine (FSM) approach
  - Successfully tested within MITRE

- **Hard to extend to new techniques**
  - Action needs to be coded into FSM
  - FSM logic needs to be recomputed
  - Inflexible in operation; hard to configure
Early CALDERA

- **First version**
  - Finite-state machine (FSM) approach
  - Successfully tested within MITRE

- **Hard to extend to new techniques**
  - Action needs to be coded into FSM
  - FSM logic needs to be recomputed
  - Inflexible in operation; hard to configure

- **Predictability during execution**
  - Easy to spot and identify

Visualized with the MITRE CyGraph tool
Designing an Adversary Decision Engine

- Typical engagements have *human* operators dictating and controlling the assessment
  - Targeting, TTP selection, domain inference…
  - … all needs to be fully automated!

- Ideally, our automated adversary will:
  - Make intelligent choices
  - Achieve tactical objectives
  - Easily incorporate new techniques
  - Work in new and unknown environments
  - Vary operations to test the defense
  - Chain weaknesses for maximum effect
Example Scenario

- Start with code execution and a RAT on Host 1
- Identified sensitive files on Host 2
- Goal: exfiltrate sensitive data from Host 2
Example Scenario

Host 1
- Exploit Vuln
- Remote Desktop
- Remote Execution
- Copy File

Host 2
- Mount Share
- Dump Credentials

Exfiltrate Data
Analyzing Copying Over a File

What do we need to do to copy a RAT over?
- Working RAT on source host
- Mounted file share from target onto source host
- Write access to file share

What happens after copying a RAT over?
- There will be a new file on the target host
- That file will contain the RAT

Requirements, or *preconditions*

Consequences, or *postconditions*
Making a Plan to Copy a File

Sequence of actions, or plan

- **Dump Credentials**
  - Need: Elevated RAT
  - Get: Credentials

- **Mount Share**
  - Need: Credentials
  - Get: Mounted Share

- **Copy File**
  - Need: Mounted Share
  - Get: File on Target
The Core CALDERA Idea

- Move from an explicit, prescribed decision model towards a dynamic, model-based one

- Tag actions with semantic execution information:
  - Preconditions specify the requirements that must be true to execute a technique
  - Postconditions specify the consequences that will be true after executing a technique

- No longer need to be explicitly told what to do!
  - Instead, compare the current state to the available actions to determine which are valid

- Added bonus: planning for the future
  - If I dump credentials now, that can help me execute lateral movement in the future!
Fun With Preconditions

- **Preconditions tell you what you can do **now**
  - In chess: can tell you which moves are valid
  - Taken further: can tell you which moves are legal

- **In the emulation sense:** given an escalated foothold on a host, we can:
  - Dump credentials
  - Add/modify registry keys
  - Setup scheduled tasks
  - …
Fun With Postconditions

- **Postconditions tell you what will be true after**
  - With preconditions, can *chain actions together* to plan for the future
  - Can evaluate *potential futures* to select the best immediate action
Fun With Postconditions

- Postconditions tell you what will be true after
  - With preconditions, can *chain actions together* to plan for the future
  - Can evaluate *potential futures* to select the best immediate action

- In the emulation sense: given an escalated foothold on a host, we can:
  - Dump credentials and then laterally move
  - Add/modify registry keys and then dump credentials
  - Setup scheduled tasks and then add/modify registry keys
  - …
Making Progress

Host 1
- Exploit Vuln
- Remote Desktop
- Remote Execution

Mount Share

Copy File

Dump Credentials

Host 2

Exfiltrate Data
Building Plans: Copying a File

Host 1

Dump Credentials
Mount Share
Copy File
Remote Execution
Exfiltrate Data

Host 2
Building Plans: Exploiting a Vulnerability

1. Dump Credentials
2. Mount Share
3. Copy File
4. Remote Execution
5. Exfiltrate Data

Host 1

Host 2

© 2017 The MITRE Corporation. All rights reserved. Approved for Public Release. Distribution Unlimited. Case Number 17-4315
Building Plans: Remote Desktop Protocol
Selecting the Right Plan

Host 1

Dump Credentials  Mount Share  Copy File  Remote Execution  Exfiltrate Data

Exploit Vulnerability  Exfiltrate Data

Dump Credentials  Remote Desktop  Exfiltrate Data

Host 2

© 2017 The MITRE Corporation. All rights reserved. Approved for Public Release. Distribution Unlimited. Case Number 17-4315
Selecting the Right Plan – The CALDERA Heuristic

- Assign each action a reward
- Score plans on summed weighted rewards

<table>
<thead>
<tr>
<th>Action</th>
<th>Reward</th>
<th>Action</th>
<th>Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exfiltrate Data</td>
<td>100</td>
<td>Copy File</td>
<td>5</td>
</tr>
<tr>
<td>Dump Credentials</td>
<td>50</td>
<td>Others</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ S(p) = \sum_{i=1}^{n} \frac{R(a_i)}{i} \]
The Language of Pre/Postconditions: The Data Model

- Need a way to logically encode what the pre and postconditions of techniques are
  - Can specify requirements/consequences by specifying facts over a data model

- CALDERA’s language: objects and typed fields
  - Objects reference commonly used Windows components
  - Fields specify properties of objects, restricted by type
    - Constructed by default, some fields may not be defined
      - (this is important later!)
An Example Host Object

Object

Field

Example

fqdn
pc1234.test.org

admins
[[“andy”, “doug”]]

hostname
pc1234
test.org

local_profiles
[[“andy”]]

system_info
...

processes
[[“512”, “133”, “415”, …]]

os_version
“Windows 7”

Host

String

Object

Integer

DateTime

Boolean

Reference

List
## Diving into the Data Model

<table>
<thead>
<tr>
<th>Schtask</th>
<th>Service</th>
<th>TimeDelta</th>
<th>Rat</th>
<th>Process</th>
<th>Persistence</th>
<th>RegKey</th>
<th>Credential</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>name</td>
<td>seconds</td>
<td>host</td>
<td>host</td>
<td>host</td>
<td>host</td>
<td>found_on_host</td>
</tr>
<tr>
<td>host</td>
<td>host</td>
<td>microseconds</td>
<td>elevation</td>
<td>elevated</td>
<td>image_name</td>
<td>user_context</td>
<td>key</td>
</tr>
<tr>
<td>status</td>
<td>start_type</td>
<td>host</td>
<td>executable</td>
<td>host</td>
<td>pid</td>
<td>elevated</td>
<td>path_to_file</td>
</tr>
<tr>
<td>cred</td>
<td>error_control</td>
<td>host</td>
<td>username</td>
<td>tid</td>
<td>pid</td>
<td>+11</td>
<td>value</td>
</tr>
<tr>
<td>+5</td>
<td>+7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
<th>OSVersion</th>
<th>Host</th>
<th>File</th>
<th>Share</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>windows_domain</td>
<td>os_name</td>
<td>fqdn</td>
<td>host</td>
<td>share_name</td>
<td>username</td>
</tr>
<tr>
<td>dns_domain</td>
<td>major_version</td>
<td>admins</td>
<td>path</td>
<td>dest_host</td>
<td>host</td>
</tr>
<tr>
<td></td>
<td>minor_version</td>
<td>hostname</td>
<td>src_host</td>
<td>share_path</td>
<td>is_group</td>
</tr>
<tr>
<td></td>
<td>build_number</td>
<td>os_version</td>
<td>src_path</td>
<td>src_host</td>
<td>domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+4</td>
<td></td>
<td>+8</td>
<td></td>
</tr>
</tbody>
</table>

- **Object**
- **String**
- **Integer**
- **DateTime**
- **Boolean**
- **Reference**
- **List**
Declaring Actions

- **CALDERA provides a syntax to declare actions**
  - *Preconditions* specify the requirements
  - *Postconditions* specify the effects
  - *Not_equal* specifies inequality invariants
  - *Preproperties* specify that certain fields must be *defined* but not necessarily a specific value
  - *Postproperties* specify that certain fields will be defined after execution
CALDERA provides a syntax to declare actions

```python
class NetUse(Step):
    value = 0
    preconditions = [("rat", OPRat),
                     ("host", OPHost),
                     ("cred", OPCredential({'$in': {'user': OPVar("hostadmins")}})),
                     ("user", OPUser(OPVar("cred.user"))),
                     ("domain", OPDomain(OPVar("user.domain")))]

    postconditions = [("share_g", OPShare({'src_host': OPVar("rat.host"),
                                             "dest_host": OPVar("host"),
                                             "share_name": 'C$'}))]

    not_equal = [("host", 'rat.host')]

    preproperties = ['domain.windows_domain', 'cred.password', 'host.fqdn', 'user.username']
    postproperties = ['share_g.share_path', "share_g.mount_point"]

    deterministic = True
```

Easy, right?
CALDERA’s Planning Algorithm

1. Update the world state
2. Figure out all valid actions to execute
3. Construct plans that lead off with those actions
   - Chain actions together by leveraging model
4. Run heuristic to determine best plan
5. Execute the first action in the best plan
6. Repeat
And we’re done! …Right?

- Using pre and postconditions we can dynamically construct plans and choose actions

- Meets our goals:
  - Can construct plans to make intelligent choices
  - Can easily incorporate new techniques by defining pre/postconditions
  - Can modify rewards/included actions to vary operations
  - Chains weaknesses to achieve goals
  - Functions in new environments*

- Wait – functions in new environments?
  - Actually, maybe not: when constructing plans, there is a significant amount of uncertainty!
  - Consider dumping credentials:
    - Sometimes they’re great creds
    - Sometimes there’s none

- In reality – handling uncertainty is a very hard problem!
And we’re done! …Right?

- Using pre and postconditions we can dynamically construct plans and choose actions.
- Meets our goals:
  - Can construct plans to make intelligent choices.
  - Can easily incorporate new techniques by defining pre/postconditions.
  - Can modify rewards/included actions to vary operations.
  - Chains weaknesses to achieve goals.
  - Functions in new environments.*
- Wait – functions in new environments?
  - Actually, maybe not: when constructing plans, there is a significant amount of uncertainty!
  - Consider dumping credentials:
    - Sometimes they’re great creds
    - Sometimes there’s none
- In reality – handling uncertainty is a very hard problem!
And we’re done! …Right?

- Using pre and postconditions we can dynamically construct plans and choose actions
- Meets our goals:
  - Can construct plans to make intelligent choices
  - Can easily incorporate new techniques by defining pre/postconditions
  - Can modify rewards/included actions to vary operations
  - Chains weaknesses to achieve goals
  - Functions in new environments*

- Wait – functions in new environments?
  - Actually, maybe not: when constructing plans, there is a significant amount of uncertainty!
  - Consider dumping credentials:
    - Sometimes they’re great creds
    - Sometimes there’s none
    - In reality, handling uncertainty is a very hard problem!

A Quick Fix with Hints

- If we can’t predict the outcome of an action, use hints
- Hints are crafted to be the “best” outcome of the action
  - E.g. performing credential dumping gives me a “useful” credential
Architecture
System Architecture
System Architecture

- Server and Agent written in Python 3
- Rat written in C#
- MongoDB
- Web interface is a JavaScript based web app
- pyDatalog logic backend
Demo
Walking Through an Operation
Walking Through an Operation

Admin

Start an operation

Agent

Agent

Clients

Clients

HTTP Server

Database

Execution Engine

Server

Attacker Model

World State

Planner
Walking Through an Operation

Diagram showing the components of an operation:

- Admin
- Agent
- Agent
- Clients
- Clients
- HTTP Server
- Database
- Execution Engine
- Server
- Attacker Model
- World State
- Planner

Bootstrap RAT
Walking Through an Operation
Walking Through an Operation
Walking Through an Operation
Walking Through an Operation
Walking Through an Operation

Admin

Agent

Agent

RAT

Clients

Clients

Get Command

HTTP Server

Database

Execution Engine

Server

Attacker Model

World State

Planner
Walking Through an Operation
Walking Through an Operation

![Diagram](image-url)
Walking Through an Operation

```
sc create \hostb ...
```
Walking Through an Operation
Walking Through an Operation

Admin

Agent

RAT

Checkin

Clients

Agent

RAT

Clients

HTTP Server

Attacker Model

World State

Planner

Database

Execution Engine

Server
Walking Through an Operation
CALDERA Applications

- Testing analytics and sensors – Does my stuff work?
- Data generation – What does bad look like?
- Red/blue team training – I need practice!
Community Participation

- **Want to get involved with CALDERA? We accept Pull Requests**
  - Bugfixes
  - Implement a new adversary (ATT&CK) technique
  - Usability features
  - Integration with other tools and frameworks
  - Enhancements to our data model
Other (Free) Tools

BloodHound – Attack Path Generation
  - https://github.com/BloodHoundAD/BloodHound

GoFetch – Automatic Execution of BloodHound paths with PowerShell tools
  - https://github.com/GoFetchAD/GoFetch

ANGRYPUPPY – Automatic execution of BloodHound paths with Cobalt Strike
  - https://github.com/vysec/ANGRYPUPPY

Death Star – Automatic Execution of attack paths with PowerShell Empire
  - https://github.com/byt3bl33d3r/DeathStar

Atomic Red Team
  - https://github.com/redcanaryco/atomic-red-team

Metta
  - <no url yet>

(Probably more, sorry if we missed you)
Related (MITRE) Efforts

- **BRAWL: Automated Bot-vs-Bot Games**
  - Free data!

- **BRAWL Shared Format (BSF)**
  - Standardized format to correlate red bot vs blue bot cyber games

- **CASCADE: Automated Host-based Investigations**
  - [https://github.com/mitre/cascade-server](https://github.com/mitre/cascade-server)
Why this Matters

- The False Negative problem is real
- Offensive testing with Adversary Emulation can help
- Automation (like CALDERA) and human adversary emulation are complementary
- Pre and postconditions + planning are powerful
- Help Us!
Show me the code!

[github.com/mitre/caldera]
MITRE is a not-for-profit organization whose sole focus is to operate federally funded research and development centers, or FFRDCs. Independent and objective, we take on some of our nation’s—and the world’s—most critical challenges and provide innovative, practical solutions.

Learn and share more about MITRE, FFRDCs, and our unique value at [www.mitre.org](http://www.mitre.org)