

Security Orchestration Platform

Team: sdmay19-19

Advisor: Doug Jacobson

Client Info: [redacted]

What is a Red Team/Adversary Simulation?

- Simulate an advanced attack against an organization
- Objective-based: “steal credit card numbers from PCI network”
- Typically only 1-2 people know that it is happening
- Red team has zero insider knowledge
- Blue team does not know about the red team assessment and will respond to red team actions like they would a real attack



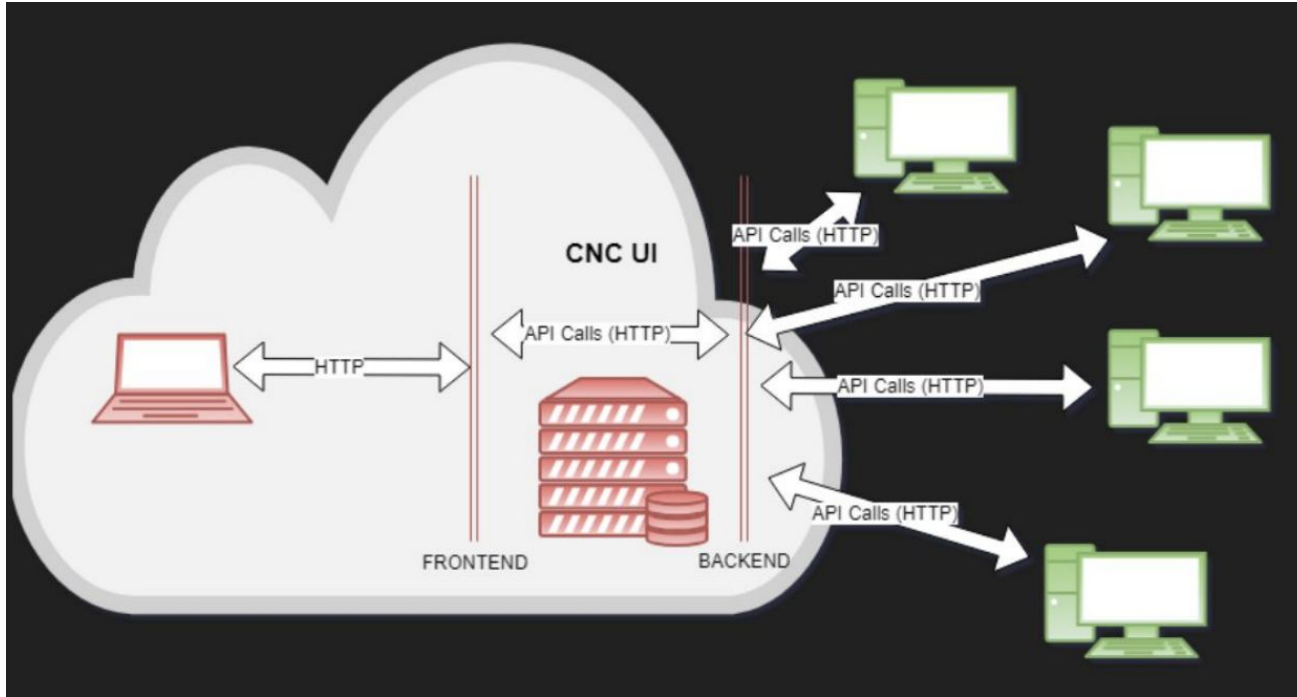
<https://render.fineartamerica.com/images/rendered/default/poster/8/10/reak/images/artworkimages/medium/1/spy-vs-spy-mr-minor.jpg>

Problem Statement

- Client faces issues with off the shelf tools being detected during red team engagements. Utilize a large number of manual processes which could be automated.
- Automation reduces the cost to deliver a red team assessment
- Custom tooling is extensible and is less likely to be detected by security solutions which are focused on detecting pre built tools



Conceptual Sketch




Functional Requirements

Bot

- Communicates with C2 via secure, encrypted API
- Tested and executable on recent versions of Windows
- Able to disconnect from a C2 and destroy itself
- Supports domain fronting
- Demonstrates persistence while remaining stealthy
- EDR solution bypass capabilities

C2

- UI for sending commands to different bots
 - Single page application (SPA) for managing bots
 - Django backend and ReactJS Frontend
 - Encrypted communication
 - Multi-user creation/deletion/authentication
 - Logs all activity by users
 - Realtime websockets for receiving data
 - Building & downloading of malware in-app
 - App managed as containers via docker-compose
 - Documentation/help for users
- 

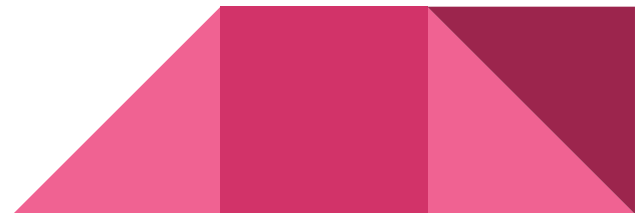
Non-Functional Requirements

Bot

- Shall be able to destroy itself upon demand or if it cannot locate the C2
- Shall not be noticeable by the average user whose system is compromised
- Shall not have predictable network traffic (ie., beacon jitter)
- Shall be configurable and support multiple deployment options
- Shall be secure against reverse engineering/losing source code/identifying the owner

C2

- Lengthy tasks shall be performed asynchronously
- User shall be able to navigate application freely without interrupting any ongoing processes
- Multiple users shall be able to access the application simultaneously
- Application shall not be accessible to general public
- Shall be quickly deployable in a temporary state



Technical/Other Constraints/Considerations

- EDR
 - Limited by accessible solutions
- Implant
 - C# and DotNetToJScript limits functionality
- Cost
 - Limit AWS instances
 - Avoid license software



Market survey

- Increased demand for red teams
 - Expensive
 - Time consuming
- Developing automation platform
 - Drastically decreases cost of development/deployment
- Custom solution
 - Not signed, no license fee

<https://workingnation.com/cybersecurity-worker-shortage-national-security/>



Potential Risks & Mitigation

- Security of the toolkit
 - Avoid EDR signature
 - Don't leak source code
- Reliability
 - Can't crash client machines
 - Want a consistent connection

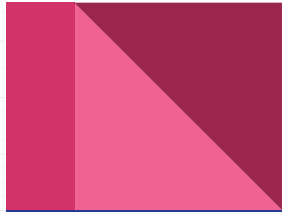
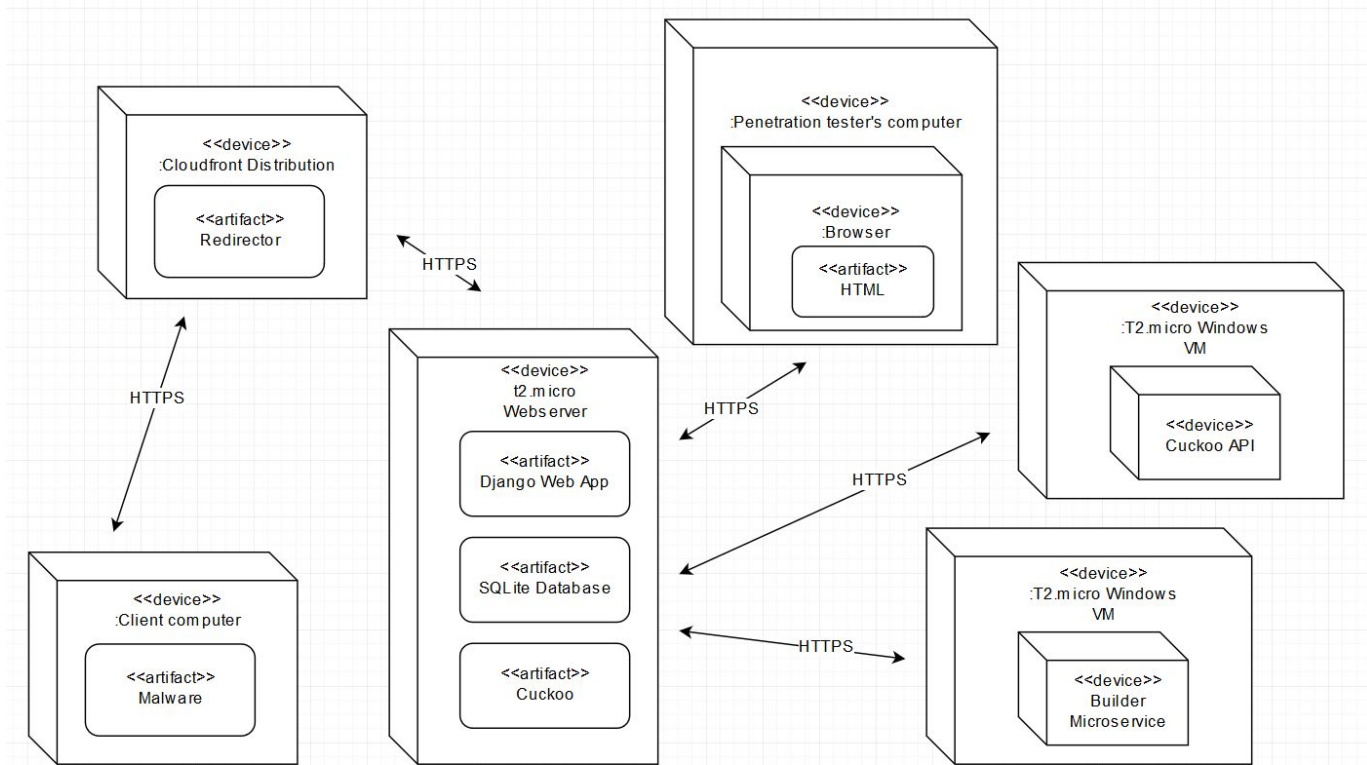


Resource/Cost Estimate

- AWS credits - \$30/month
 - CloudFront redirector for Domain Fronting - \$5
 - Linux t2.micro for Web Application prototype - \$7.59
 - Windows t2.nano for Malware Builder - \$5.91
 - Windows t2.micro for Cuckoo - \$11.82
- Time Estimate - 500 hours



Functional Decomposition



Detailed Design (functional modules design, interface definition, user interfaces, etc.)


The screenshot displays a web interface for managing a bot. At the top, there are navigation tabs for 'Overview', 'Your Files', and 'Docs', and a 'BotNet' logo in the top right corner. The main title is 'Bot rzak5pcfm5ljl8kent6', with a 'Back to Overview' button below it.

The interface is divided into four main sections:

- Messages:** Contains a 'Refresh' button and a terminal window showing system information:

```
ISADMIN=False
OS=Microsoft Windows NT 6.2.9200.0
ARCH=AMD64
USER=WINDEV1886EVAL\User
IP ADDRESS=192.168.64.129
HOSTNAME=winDev1886Eval
```
- Command:** Shows the current command as '"info" (currently)'. It includes an 'Update Command' section with a 'New Command' dropdown menu (set to 'shell'), a text input field for 'Command to run on bot' (with a placeholder 'Type command you wish to run (e.g., !pcntfg)'), and an 'Update' button.
- Details:** Displays the bot's 'IP ADDRESS' as 192.168.64.129, with a 'Last Checked In: 7 seconds ago' status. It also shows 'Created: 50 seconds ago' and a 'TAG' field with a 'Default tag' and a 'DELETE BOT' button.
- Files:** Features a 'Refresh' button and a message stating 'No files yet.'

HW/SW/Technology Platform(s) used

- ReactJS UI framework
 - Semantic UI CSS framework
 - Django Python Web framework
 - Django REST Framework (for APIs)
 - SQLite which we may switch to MariaDB or Postgress
 - Docker
 - C#
 - Cuckoo
- 

Test Plan

- Tests focused on completing deliverables
- Focusing on manual testing over Automated testing
- Time sink is too large for Automated testing with minimal reward
- Can use that time for developing additional features
- EX: Ensure that our malware is not detected by common EDR solutions
 - Run malware while C2 Server+Bots communicating. Check for detection.



Prototype Implementations

```
15:16:41] "GET /api/v1/message/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 193
15:16:41] "GET /api/v1/viewbotupload/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 2
15:16:41] "GET /api/v1/filetobot/ HTTP/1.1" 200 1960
15:16:41] "GET /api/v1/bot/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 290
15:16:41] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 4
15:16:41] "GET /favicon.ico HTTP/1.1" 200 557
15:16:42] "GET /api/v1/bots/ HTTP/1.1" 200 292
15:16:42] "GET /favicon.ico HTTP/1.1" 200 557
15:16:45] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 4
15:16:45] "GET /api/v1/message/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 193
15:16:45] "GET /api/v1/bot/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 290
15:16:45] "GET /api/v1/viewbotupload/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 2
15:16:45] "GET /favicon.ico HTTP/1.1" 200 557
15:16:45] "GET /api/v1/filetobot/ HTTP/1.1" 200 1960
15:16:50] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 4
15:16:51] "POST /api/v1/heartbeat/ HTTP/1.1" 200 9
15:17:00] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 4
15:17:00] "POST /api/v1/heartbeat/ HTTP/1.1" 200 9
15:17:07] "PATCH /api/v1/bot/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 300
15:17:11] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 14
15:17:11] "POST /api/v1/message/ HTTP/1.1" 201 1371
15:17:20] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 14
15:17:20] "POST /api/v1/heartbeat/ HTTP/1.1" 200 9
15:17:31] "GET /api/v1/command/rzak5pcfm5ljlt8kent6 HTTP/1.1" 200 14
15:17:31] "POST /api/v1/heartbeat/ HTTP/1.1" 200 9
```


Conclusion (Project Status)

- Project deliverables determined feasible
- Project responsibilities split between two teams
 - Implant: Paul / Adam / Logan
 - Command & Control: Daniel / Vijay / Justin
- Next semester will be heavily focused on implementation

