

GHARF: GitHub Actions RedTeam Framework



Nov 18, 2025

NTT DOCOMO BUSINESS, Inc.

Yuuki Matsumoto, Yusuke Kubo

Self-Introduction

Yuuki Matsumoto

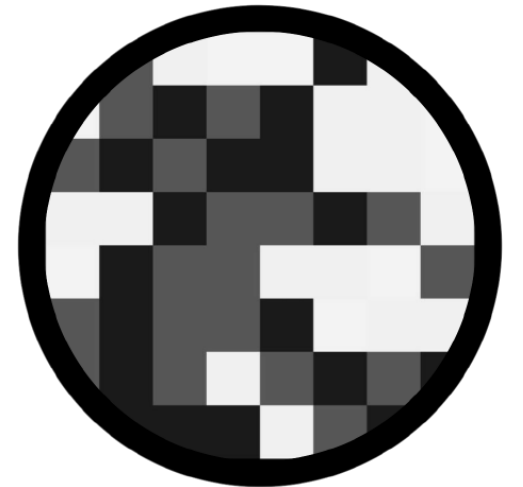
Affiliation: NTT DOCOMO Business, Innovation Center

Background:

- Joined NTT DOCOMO Business (formerly NTT Communications) in FY2024
- Specialized in the security field since student days

Work Responsibilities:

- Technical assistance for WideAngle RedTeam service
- Internal security enhancement
- R&D on Applied Offensive Security techniques
- etc.



SNS(X): @Raster0x2a_tech

Agenda

- **Introduction:** Overview of today's talk
- **Background:** Challenges in RedTeam Exercises
- **Concept:** "CAI/CAD"
- **Framework:** "GHARF"
 - Use Cases
 - Underlying Technologies
 - Features of GHARF
 - How it works (Mechanism)
 - Implementation
- **Demo**
- **Ethical Considerations**
- **Summary**

Theme of this Presentation:

- Proposing a **RedTeam x CI/CD approach**
- Introducing a **framework implementing it with GitHub Actions**

Basic Knowledge

What is a RedTeam Exercise?

- A security assessment that tests an organization's resilience through simulated real-world cyberattacks.
- Often conducted as training for the defensive (Blue) Team.

What is CI/CD?

- Short for Continuous Integration / Continuous Delivery — a methodology for automating the software development pipeline.
- It involves phases such as code, build, test, delivery, and deployment, aiming to reduce operational costs and accelerate development cycles.

Background: Challenges in RedTeam Exercises

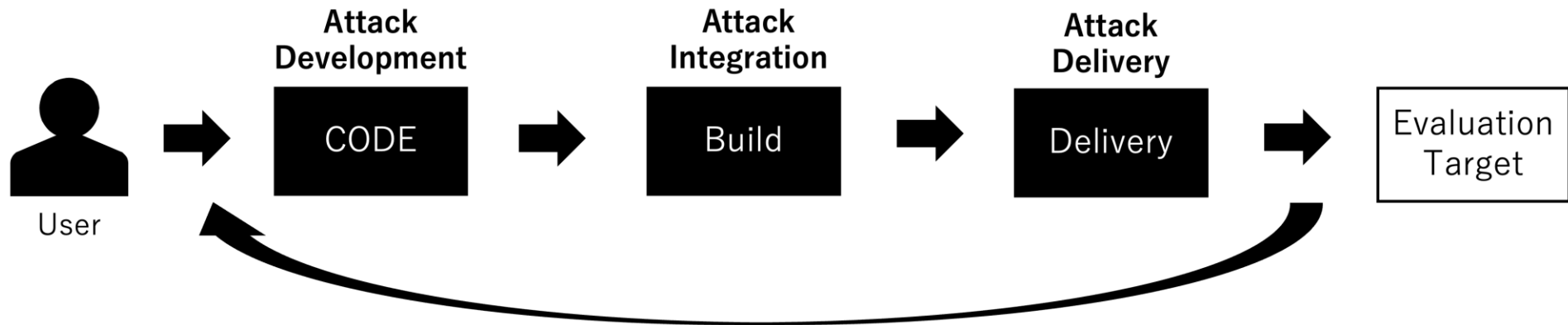


 **Executing these cycles requires significant effort**

Concept: "CAI/CAD"

CAI/CAD : Continuous Attack Integration / Continuous Attack Delivery

- An approach that applies CI/CD mechanisms like "build" and "delivery" to Red Team operations
- Automates the **development, preparation, and execution** phases of simulated attacks, enabling faster operation cycles



Framework: GHARF

GHARF: **Git**Hub **A**ctions **R**edTeam **F**ramework

An OSS framework that realizes the CAI/CAD concept using GitHub Actions



Available on GitHub

<https://github.com/nttcom/gharf>

<https://github.com/nttcom/gharf-workflows>

Use Cases

Primary Users

- RedTeam aiming to conduct highly customized exercises
- RedTeam seeking advanced scenarios while minimizing operational costs

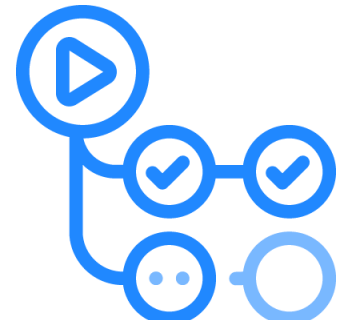
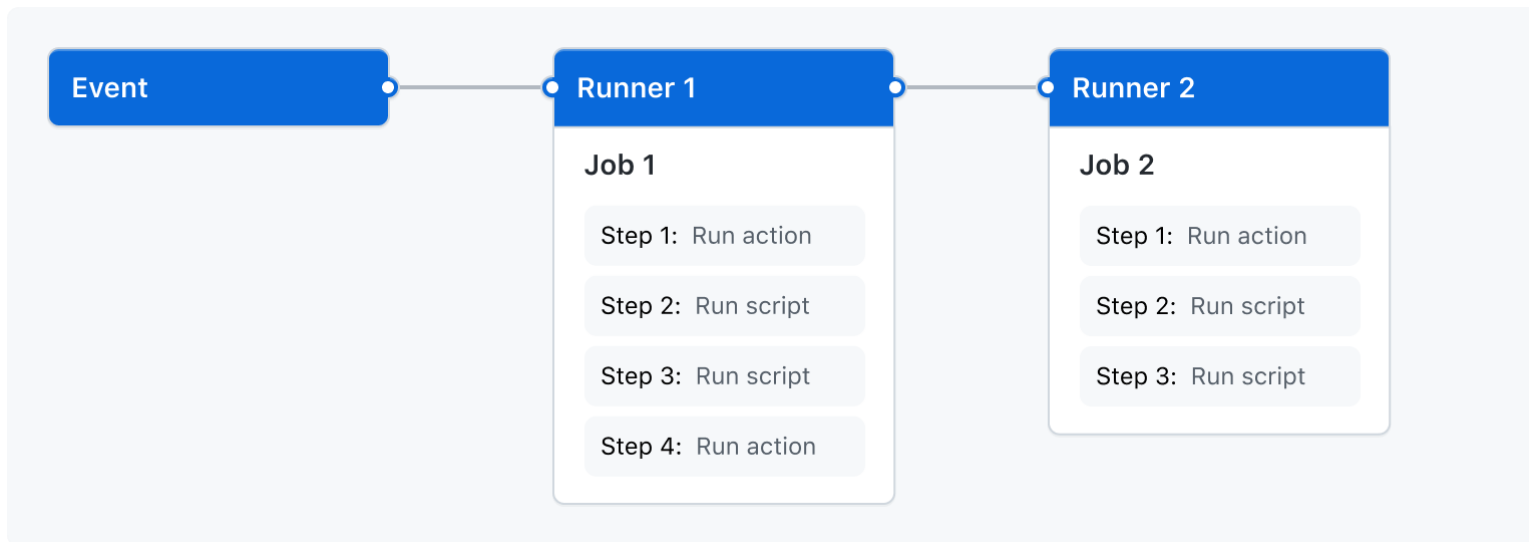
Other Users

- Researchers evaluating new C2 techniques using GitHub Actions Self-hosted Runners
- BlueTeam wishing to repeatedly execute attack scenarios for continuous training

Underlying Technology for GHARF

What is GitHub Actions?

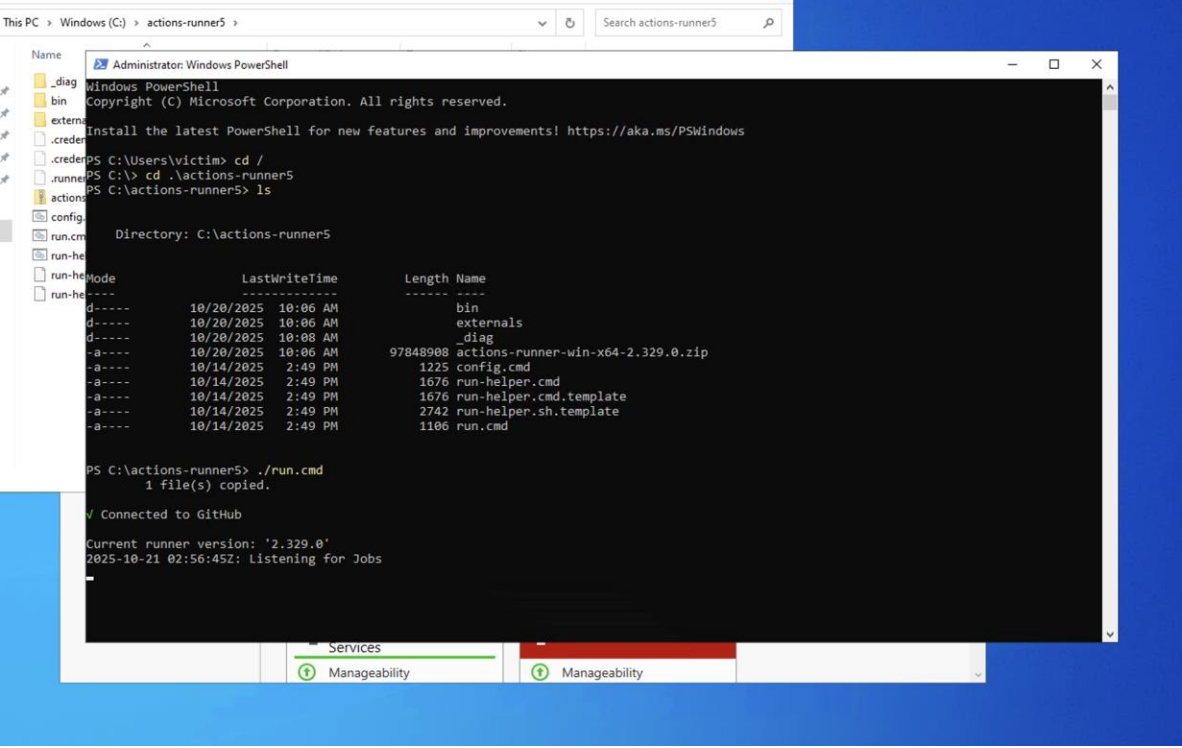
- A CI/CD platform that automates software development workflows on GitHub
- Automatically executes tasks such as build, test, and deployment, triggered by repository events
- Workflows are defined in YAML files and executed on virtual machines



Underlying Technology for GHARF

What is Self-hosted runner?

- In addition to GitHub-hosted virtual machines, users can run workflows on their own machines.
- Commonly used to customize machine configurations or to build/test in specialized environments.



```
This PC > Windows (C:) > actions-runner5 >
Name
-----
bin
externals
_diag
config
run.cmd
run-helper.cmd
run-helper.sh
run-helper.sh.template
run.cmd

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\victim> cd /
PS C:\> cd .\actions-runner5
PS C:\actions-runner5> ls

Directory: C:\actions-runner5

Mode                LastWriteTime         Length Name
----                -
d-----          10/20/2025  10:06 AM             bin
d-----          10/20/2025  10:06 AM          externals
d-----          10/20/2025  10:08 AM             _diag
-a----          10/20/2025  10:06 AM    978489808 actions-runner-win-x64-2.329.0.zip
-a----          10/14/2025   2:49 PM        1225 config.cmd
-a----          10/14/2025   2:49 PM        1676 run-helper.cmd
-a----          10/14/2025   2:49 PM        1676 run-helper.cmd.template
-a----          10/14/2025   2:49 PM        2742 run-helper.sh.template
-a----          10/14/2025   2:49 PM        1106 run.cmd

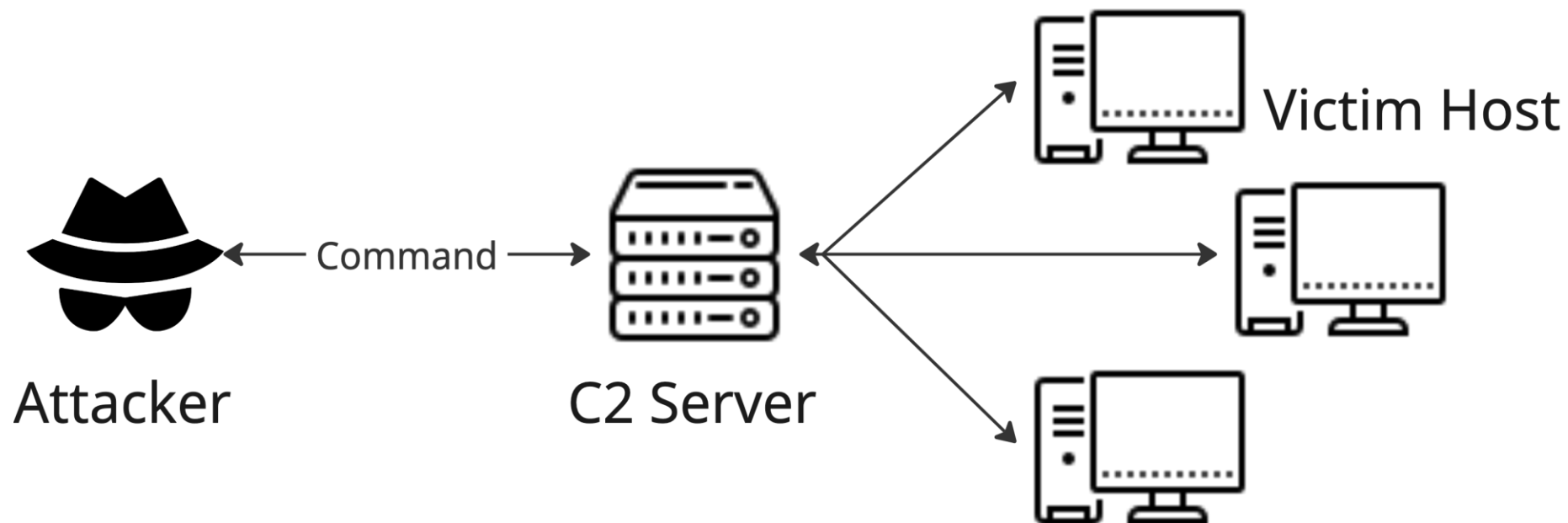
PS C:\actions-runner5> ./run.cmd
1 file(s) copied.

[X] Connected to GitHub
Current runner version: '2.329.0'
2025-10-21 02:56:45Z: Listening for Jobs
```

Underlying Technology for GHARF

What is C2 (Command & Control)?

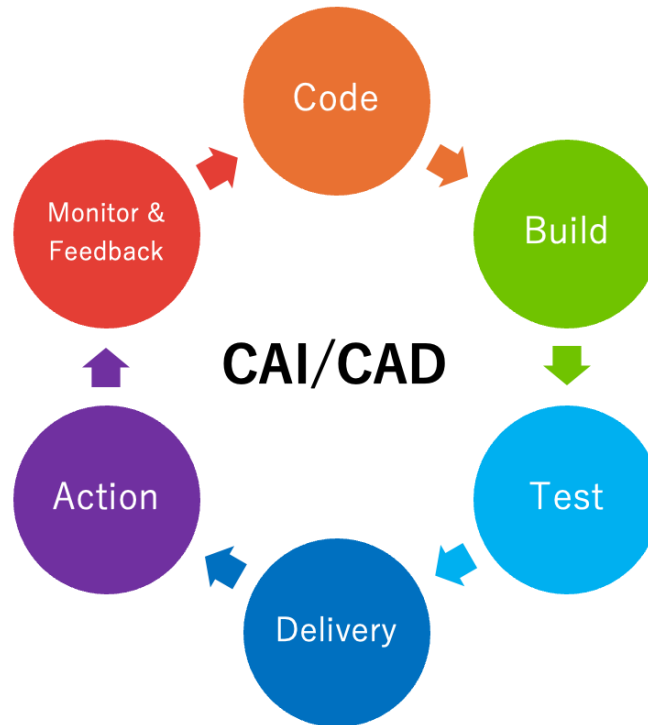
- A cyberattack technique in which an attacker establishes communication with a compromised environment, enabling remote command execution and data exfiltration.
- This framework utilizes GitHub Actions as C2 infrastructure.



Features of GHARF

1. Fully Automated RedTeam Operations

- Automates the entire flow from attack development to preparation and execution
- Minimizes the operational overhead, allowing RedTeam to focus on scenario development
- Integrates each process into a continuous pipeline, enabling seamless data transfer between phases
 - For example, the framework can automate the full cycle: building an attack tool -> delivering it to the target environment -> executing it -> analyzing the obtained information with other tools



Features of GHARF

2. RedTeam Operations as Code

- Operations are structurally defined using YAML as GitHub Actions workflow files
- Enables repeatable execution of operations
- Supports version control for operations
- Allows the same operation to be executed across different environments

Describes workflow processes in YAML format

```
name: GitHub Actions Demo
run-name: ${ github.actor }} is testing out GitHub Actions 🚀
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${ github.event_name }} event."
      - run: echo "🌍 This job is now running on a ${ runner.os }} server hosted by GitHub!"
      - run: echo "📍 The name of your branch is ${ github.ref }} and your repository is ${ github.repository }}."
      - name: Check out repository code
        uses: actions/checkout@v5
      - run: echo "💡 The ${ github.repository }} repository has been cloned to the runner."
      - run: echo "🖨️ The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${ github.workspace }}
      - run: echo "🍏 This job's status is ${ job.status }}."
```

<https://docs.github.com/ja/actions/get-started/quickstart>

Features of GHARF

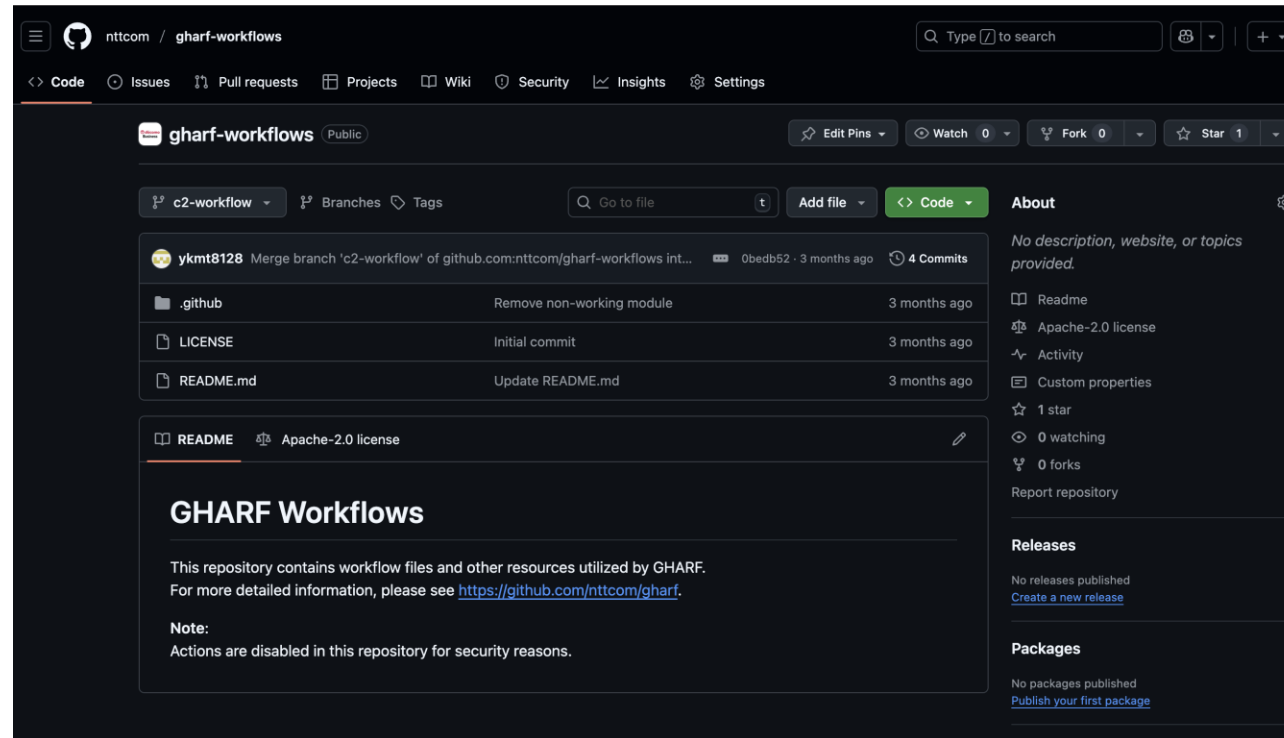
3. Resource-Less Architecture

- No custom C2 agent development required, as it uses the GitHub Actions self-hosted runner as the agent.
- GitHub-hosted runners can be used as machine resources.
 - For building attack tools, obfuscation processing, testing, etc.
 - For analyzing and processing of results (e.g., password cracking)

Features of GHARF

4. Easy and Quick Setup

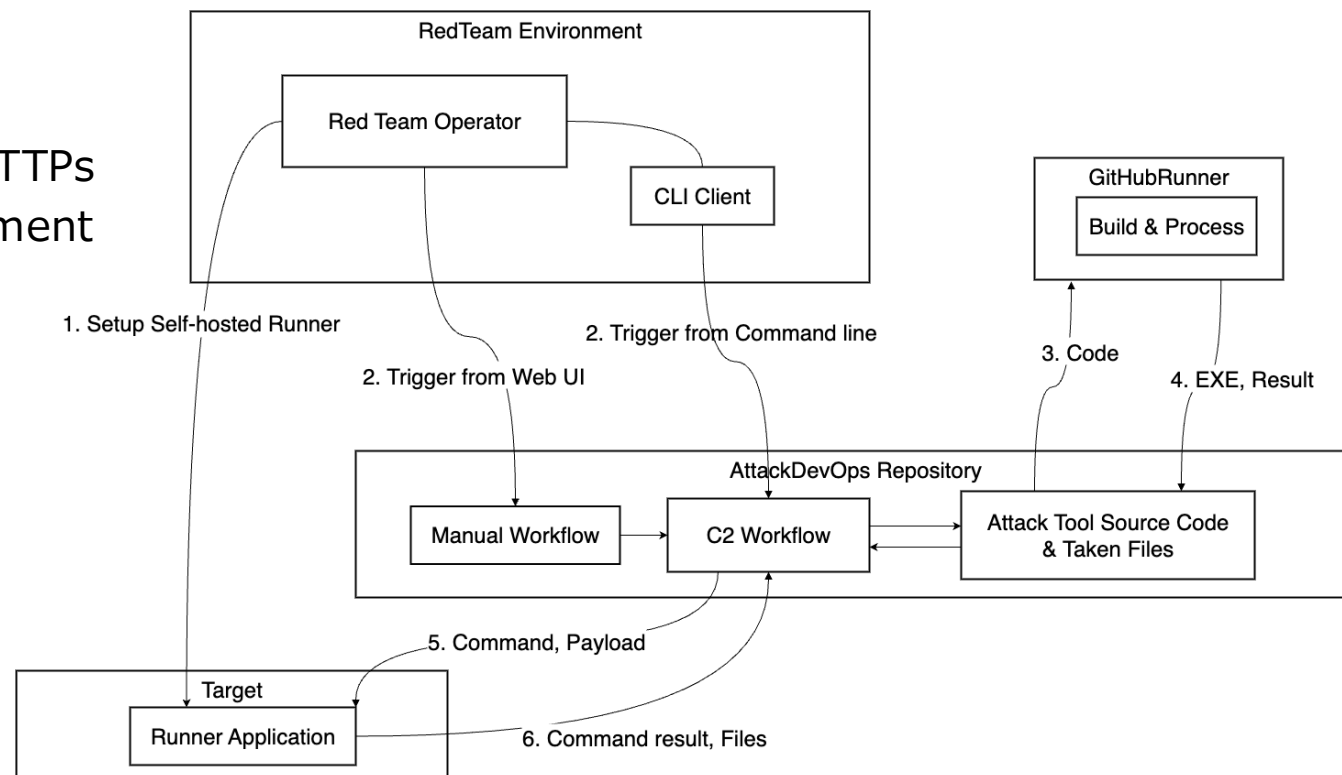
- Can be launched with minimal preparation:
 1. Create a GitHub account
 2. Set up a GitHub repository for Attack/Development
 3. Download and execute the runner application in the target environment



<https://github.com/nttcom/gharf-workflows>

How GHARF Works – Architecture

- RedTeam Operator Environment
 - CLI tool to execute C2 commands via GitHub (Optional)
- AttackDevOps GitHub Repository (Managed by Red Team)
 - GitHub Actions workflow that functions as a C2 server
 - Workflow for manually executing attack tasks
 - Attack Resources
 - Source code of attack tools
 - Workflows and actions implementing TTPs
 - Files to exchange with target environment
- Target Environment
 - Runner Agent (Self-hosted runner)



How GHARF Works - Usage Flow

Example usage flow

1. Create a repository on GitHub (Clone the GHARF starter repository)
2. Download the self-hosted runner agent from GitHub
3. Set up the self-hosted runner in the target environment
4. Push the attack workflow file to the repository
5. Trigger the workflow from the CLI client or GitHub Web UI
6. The operations defined in the workflow are executed automatically
7. Repeat steps 4-6 according to the operation's objective

GHARF Implementation - C2 Workflow

A workflow to manage and execute C2 operations

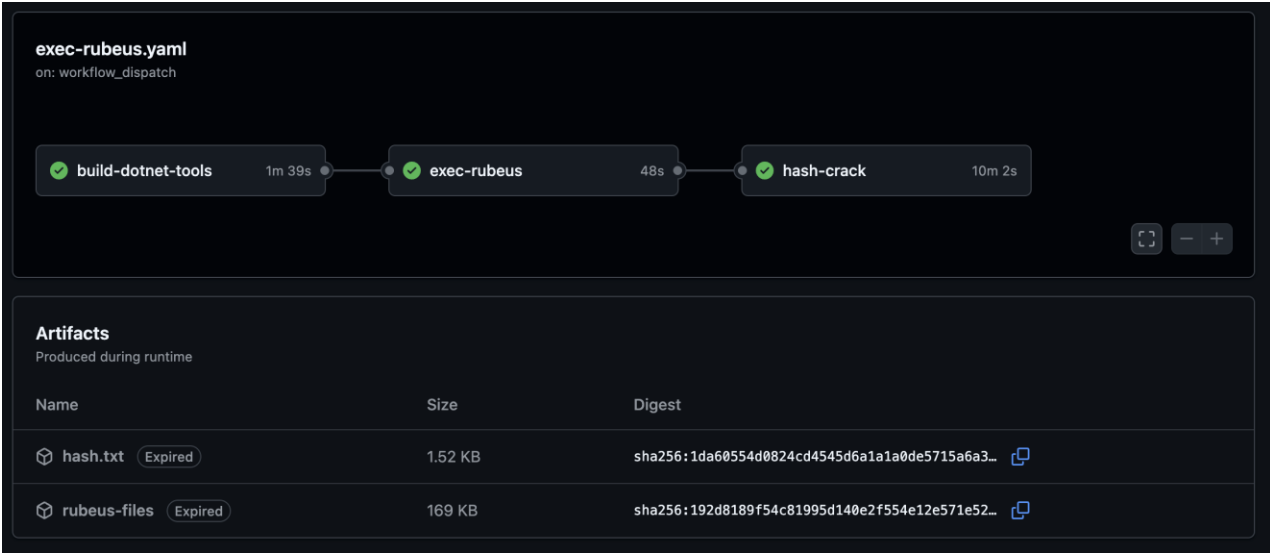
- Implemented in YAML as a GitHub Actions workflow
- Supported Commands
 - Execute shell commands / JavaScript code
 - File upload / download
 - Execute custom workflows
- Runner Specification
 - Use GitHub Actions' runner label feature to specify the C2 host on which commands are executed.

```
1  name: c2
2  on:
3    workflow_dispatch:
4    inputs:
5      hostlabels:
6        description: 'Host label to execute'
7        type: string
8        required: true
9    command_type:
10     description: 'Command type'
11     type: choice
12     required: true
13     options:
14       - 'shell'
15       - 'javascript'
16       - 'download'
17       - 'upload'
18       - 'custom-module'
```

GHARF Implementation - Custom Workflow

つながり。驚きを。幸せを。

- GHARF supports a mechanism that allows users to create and integrate custom workflows to add their own TTPs into the operation flow.
- Designed so that users can publish and share their workflows for reuse by others.
- The following sample workflows are included in the repository:
 - A workflow that executes Kerberoasting with Rubeus and cracks the obtained NTLM hash with Hashcat
 - A workflow that executes SharpHound and retrieves the resulting ZIP file.



The screenshot displays a workflow execution interface for a file named 'exec-rubeus.yaml'. The workflow is triggered by the event 'on: workflow_dispatch'. It consists of three sequential steps, each marked with a green checkmark to indicate successful completion:

- build-dotnet-tools**: Duration 1m 39s
- exec-rubeus**: Duration 48s
- hash-crack**: Duration 10m 2s

Below the workflow steps, there is an 'Artifacts' section titled 'Produced during runtime'. It contains a table with the following data:

Name	Size	Digest
hash.txt Expired	1.52 KB	sha256:1da60554d0824cd4545d6a1a1a0de5715a6a3...
rubeus-files Expired	169 KB	sha256:192d8189f54c81995d140e2f554e12e571e52...

GHARF Implementation - Manual Workflow

つながる。驚きを。幸せを。



Workflow for manual execution from the Web UI

- The workflow can also be executed manually from the GitHub Web UI.
 - Uses GitHub's interface to manually trigger a Workflow Dispatch event.
- The workflow file is available in the starter repository
 - You can start using it immediately by cloning the repository and setting it up as a private repository

The screenshot shows a dark-themed web interface for running a workflow. At the top right, there is a 'Run workflow' button. Below it, a form titled 'Use workflow from' contains several fields: a dropdown menu for 'Branch: c2-workflow', a text input for 'Select the label of the host to execute *' with the value 'c2-win-20250609', a dropdown for 'Type of command, e.g.' with 'shell' selected, and a text input for 'The shell script(PowerShell/Bash) or JavaScript source code to execute (Required if command_type is shell/javascript)' with the value 'whoami'. Below this is a text input for 'The filepath to download (Required if command type is download, otherwise blank)' which is empty. At the bottom, there is a text input for 'Custom module workflow name (Required if command_type is custom-module)' which is empty, and a green 'Run workflow' button.

GHARF Implementation - CLI Client

CLI Client Implementation

- GHARF also supports a CLI client that enables operation directly from the command line.
- Features
 - Calling workflows set in the repository
 - Executing shell commands / JavaScript
 - File upload / download
 - Executing custom modules
 - Log retrieval and file exchange with the repository
- Command-line version
 - Specifying information via options

```
(.venv)-(basdev@kali)-[~/GHARF_dev/gharf]
└─$ python3 client.py shell --sourcecode 'whoami' --hostlabel c2-win-20251020-1
runners: c2-win-20251020-1

(.venv)-(basdev@kali)-[~/GHARF_dev/gharf]
└─$ python3 client.py logs --n 1
results found: 1

=====
id           : 18773658550
name         : c2-shell-whoami
display_title : c2-shell-whoami
run_number   : 109
status       : queued
conclusion    : None
workflow_id  : 137345713
created_at   : 2025-10-24T08:06:07Z
updated_at   : 2025-10-24T08:06:07Z
log          : None
```

GHARF Implementation - Interactive CLI

CLI Client Interactive Version

- A wrapper tool for the command-line version that allows for intuitive operation via an interactive UI
 - Easy to use with mode-switching functionality
 - Currently under development (most features are already implemented)

```
└─$ python3 client_interactive.py
Welcome to GHARF!



Type 'help' for commands or 'exit' to quit
GHARF> help
Commands:
  shell           Run any shell command
  javascript      Run any JavaScript
  download-run    Run file download job
  download-list   View a list of recently downloaded files
  download-file   Download a file from job-id
  upload-file     Upload a file(Local→upload branch)
  upload-run      Run file upload job (upload branch→Target host)
  custom-module  Invoke a custom workflow
  logs           View recent command execution logs
GHARF> shell
Select the target hostlabel by number from the following
1. c2-win-kerb-20250609
Enter the number: 1
Enter shell command(inline, stateless)
Type '!exit' to exit shell mode
GHARF[c2-win-kerb-20250609][shell]$ ls -al
Executing shell command: ls -al
runners: c2-win-kerb-20250609
GHARF[c2-win-kerb-20250609][shell]$
```

Demo - Runner Setup

Demo - CLI Client

CLI Client Demonstration

Demo – GitHub Web Interface

Execute commands via GitHub Web UI

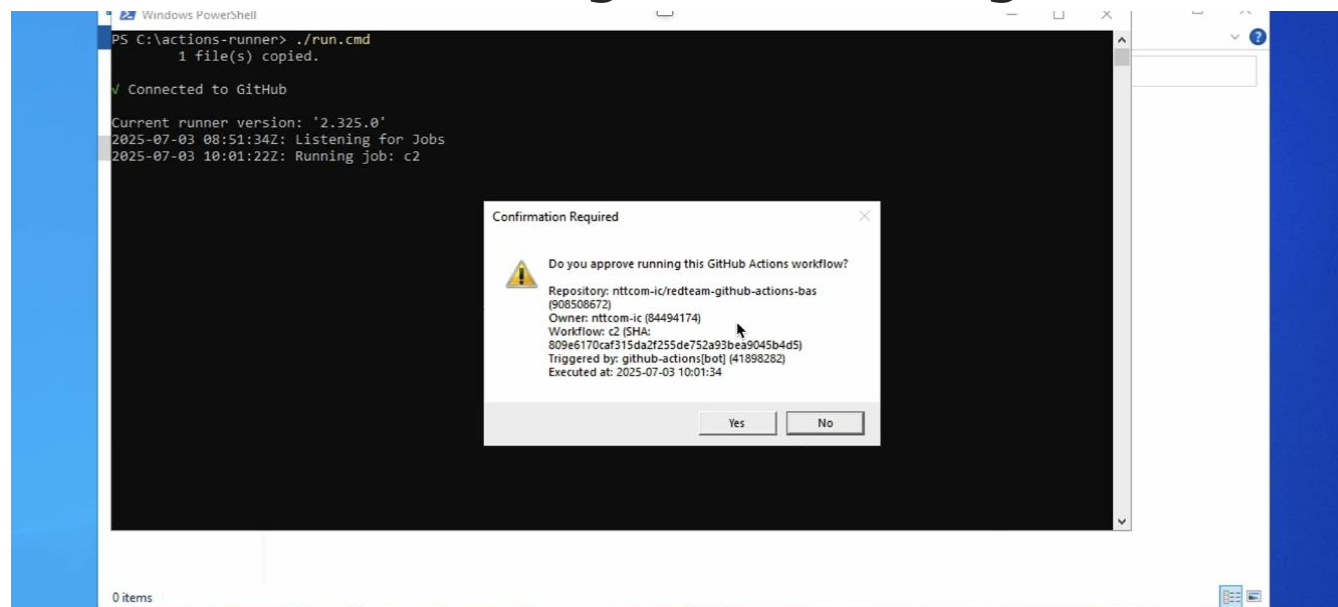
Ethical Considerations

Responsible Disclosure

- The technique of using a self-hosted runner as a C2 agent, which is core to GHARF, has been disclosed to and approved by GitHub prior to public release

User Consent Process

- An approval process is implemented to prevent unauthorized actions or unexpected operations from being executed against the user's intent



Ethical Considerations

Artifact Traceability

- Implemented a mechanism to embed Indicators of Compromise (IoC) into artifacts (e.g., binaries) generated by the workflow.
- YARA rules for detecting these artifacts are also provided in the repository.

Security Policy

- A contact point is available to promptly receive reports when tool abuse is detected.
- The security policy is published in the GitHub repository:
<https://github.com/nttcom/gharf?tab=security-ov-file#readme>

Summary

1. Proposed the "CAI/CAD" approach to address the operational cost problem of Red Team exercises by applying CI/CD concepts
2. Introduced "GHARF," a framework that implements this concept
3. Demonstrated that more advanced RedTeam exercises can be achieved through operational automation and reduced resource preparation costs