Mythic - Mythic

Archived: 2026-04-05 15:47:28 UTC

What is Mythic?

Mythic is a multiplayer, command and control platform for red teaming operations. It is designed to facilitate a plug-n-play architecture where new agents, communication channels, and modifications can happen on the fly. Some of the Mythic project's main goals are to provide quality of life improvements to operators, improve maintainability of agents, enable customizations, and provide more robust data analytic capabilities to operations. Fundamentally, Mythic uses a web-based front end (React) and Docker containers for the back-end. A GoLang server handles the bulk of the web requests via GraphQL APIs and WebSockets. This server then handles connections to the PostgreSQL database and communicates to the other Docker containers via RabbitMQ. This enables the individual components to be on separate physical computers or in different virtual machines if desired. A helpful view of the current state of the C2 Profiles and Agents for Mythic can be found here:

A reverse Nginx proxy provides a single port to connect through to reach back-end services. Through this reverse proxy, operators can connect to:

- React UI
- Hugo documentation container (documentation on a per-agent and per-c2 profile basis)
- Hasura GraphQL console (test GraphQL queries, explore/modify the database)
- Jupyter Notebook (with Mythic Scripting pre-installed and pre-created examples)

Why use Mythic?

Data modeling, tracking, and analysis are core aspects of Mythic's ability to provide quality of life improvements to operators. From the very beginning of creating a payload, Mythic tracks the specific command and control profile parameters used, the commands loaded into the payload and their versions, who created it, when, and why. All of this is used to provide a more coherent operational view when a new callback checks in. Simple questions such as "which payload triggered this callback", "who issued this task", and even "why is there a new callback" now all have contextual data to give answers. From here, operators can start automatically tracking their footprints in the network for operational security (OpSec) concerns and to help with deconflictions. All agents and commands can track process creates, file writes, API calls, network connections, and more. These artifacts can be automatically recorded when the task is issued or even reported back by agents as they happen. Mythic also incorporates MITRE ATT&CK mappings into the standard workflow. All commands can be tagged with ATT&CK techniques, which will propagate to the corresponding tasks issued as well. In a MITRE ATT&CK Matrix view (and ATT&CK Navigator view), operators can view coverage of possible commands as well as coverage of commands issued in an operation. Operators can also provide comments on each task as they operate to help make notes about why a task was important. Additionally, to make deconfliction and reporting easier, all commands, output, and comments can be globally searched through the main web interface.

Resources and Contributing

- Check out the code on GitHub: https://github.com/its-a-feature/Mythic
- Join the #Mythic channel in the BloodHound public slack
- Reach out on Twitter: @its_a_feature_
