

Infrastructure de Red Team

Objectifs du chapitre

Après avoir lu ce chapitre et réalisé les exercices, vous serez capable de :

- Comprendre les systèmes C2 courants
- Utiliser des canaux de communication C2
- Utiliser des redirecteurs C2
- Sécuriser votre infrastructure
- Comprendre comment mettre en place une infrastructure C2

Pour initier l'accès initial, il est nécessaire de créer une infrastructure Red Team. L'infrastructure utilisée par une équipe Red Team doit se fondre dans la masse et ressembler à du trafic légitime. Le souci du détail lors de cette étape est crucial, car l'infrastructure que vous déployez en tant qu'attaquant détermine votre réussite lors de l'exécution de l'attaque et du maintien de l'accès dans un environnement compromis.

Ces dernières années, on a observé une évolution significative vers l'utilisation de plateformes cloud, telles qu'Azure et Google Cloud, pour les infrastructures Red Team. Auparavant, les opérateurs déployaient généralement leurs infrastructures de commande et de contrôle (C2) sur des serveurs auto-hébergés ; toutefois, le passage aux services cloud permet de créer des infrastructures plus flexibles, évolutives et résilientes. La possibilité de déployer dynamiquement de nouvelles instances ou d'ajuster les services en fonction de l'ampleur de la mission a réduit la charge opérationnelle.

De plus, les services cloud s'intègrent souvent à des réseaux de diffusion de contenu (CDN), ce qui permet de mêler le trafic C2 à celui, légitime, du fournisseur cloud, ajoutant ainsi une couche supplémentaire d'obfuscation. Des CDN tels que Cloudflare ou le CDN classique d'Azure sont de plus en plus utilisés pour acheminer le trafic C2 via des infrastructures web légitimes et fiables. Les membres d'une équipe Red

Team peuvent tirer parti des CDN pour dissimuler leur trafic en passant par ces fournisseurs. Cette technique complique la tâche des équipes de réponse aux incidents, qui peinent alors à distinguer le trafic malveillant du trafic légitime.

Une autre évolution notable concerne l'adoption du protocole DNS over HTTPS (DoH) pour les canaux de communication C2 des équipes Red Team. Ces communications transmettent les requêtes DNS via des connexions HTTPS (DoH), contournant ainsi les systèmes traditionnels de surveillance réseau. Il est important de noter que, si le DoH offre une certaine obfuscation, il pose également des défis en matière de stabilité et de fiabilité des communications, car certains réseaux d'entreprise peuvent bloquer les requêtes DNS non standard. Par ailleurs, les anciens frameworks C2 utilisaient des intervalles de rappel statiques, susceptibles d'être détectés par les systèmes de surveillance réseau. L'introduction du « jitter » (variation aléatoire des délais) dans les profils C2 a permis aux équipes Red Team de randomiser le trafic associé à leurs agents et d'éviter ainsi les intervalles prévisibles.

Ce chapitre a pour but d'examiner l'évolution du développement des infrastructures ces dernières années, de passer en revue les canaux de communication C2 courants et d'expliquer comment configurer votre infrastructure pour garantir le succès de l'opération. Lors de la préparation de votre infrastructure Red Team, vous devrez choisir quels éléments mettre en place en fonction de l'objectif de la mission. Les équipes Red Team peuvent opérer en partant du principe qu'une intrusion a déjà eu lieu ou commencer par le périmètre externe, ce qui implique de cibler une application ou un utilisateur situé à l'extérieur. Nous approfondirons les aspects méthodologiques du Red Teaming au chapitre 4, « Méthodologie et outils modernes de Red Teaming ». Par ailleurs, si l'ingénierie sociale entre en jeu, il vous faudra préparer une infrastructure d'hameçonnage (phishing) ou de vishing (hameçonnage vocal), sujets que nous aborderons au chapitre 5, « Ingénierie sociale et évaluations physiques ».

Aperçu de l'infrastructure « Red Team »

Par le passé, l'infrastructure « Red Team » destinée aux communications de type « Command and Control » (C2) n'avait pas besoin d'être aussi sophistiquée qu'aujourd'hui ; toutefois, la situation a évolué ces dernières années. Par exemple, l'utilisation de réseaux de diffusion de contenu (CDN) permet de faire en sorte que le trafic généré par l'équipe d'attaque semble provenir d'une entreprise légitime, telle que Microsoft ou un autre fournisseur de services cloud. Le trafic issu des principaux fournisseurs cloud est généralement considéré comme fiable, car les entreprises légitimes utilisent elles-mêmes des CDN pour leur infrastructure. Cette technique permet de contourner les filtres de trafic qu'une entreprise pourrait avoir mis en place au niveau de son périmètre réseau.

L'infrastructure constitue le cœur de la mission de la « Red Team » et assure la cohérence globale de l'opération. Plusieurs éléments doivent être pris en compte lors de sa conception : les besoins nécessaires pour atteindre l'objectif, la sécurisation de l'infrastructure, la mise en place d'une infrastructure de secours et la garantie d'une journalisation adéquate tout au long du cycle de vie de la mission, en vue de l'élaboration des rapports.

Command and Control (C2)

Les serveurs de « Command and Control » (C2) sont utilisés par les opérateurs pour piloter leur mission et exécuter des actions à distance sur les ordinateurs des cibles. Le choix de ces serveurs dépend du type de C2 utilisé, car il existe une multitude d'options, tant commerciales qu'open source. La « C2 Matrix » est un document open source de référence pour les professionnels du secteur ; il permet de passer en revue les solutions C2 disponibles (commerciales et publiques) et fournit des informations sur les types d'agents, la prise en charge de SOCKS, etc.

Ces informations sont disponibles à l'adresse suivante :
<https://howto.thec2matrix.com>.

Sur le plan commercial, les solutions C2 les plus en vue sont Nighthawk (de MDSec) et Cobalt Strike (de Forta). Parmi les frameworks C2 publics les plus répandus et soutenus par la communauté, on compte Mythic, Sliver, Havoc et BadRats. L'un des avantages des frameworks C2

publics réside dans leur flexibilité de personnalisation, permettant une efficacité optimale, alors que les solutions commerciales peuvent s'avérer plus restrictives à cet égard. Bien que les solutions commerciales puissent être détectées par signature (via les EDR) lorsqu'elles sont utilisées avec leurs paramètres par défaut, elles nécessitent généralement moins de personnalisation initiale que les options C2 publiques pour être opérationnelles. Nighthawk (<https://www.mdsec.co.uk/nighthawk/>), lancé en 2021, est un framework C2 de pointe conçu pour des techniques opérationnelles avancées ; son utilisation est réservée aux opérateurs « Red Team » autorisés. L'équipe de MDSec a développé ce C2 en s'appuyant sur des travaux de recherche et développement propriétaires, ce qui lui confère des capacités avancées. Il prend en charge plusieurs opérateurs, des techniques d'évasion sophistiquées, des chargeurs (loaders) personnalisés, des profils malléables, le protocole SOCKS5, les Beacon Object Files (BOF), les communications furtives, des options sécurisées de post-exploitation ainsi qu'un générateur de fichiers PE personnalisé pour les opérateurs.

Cobalt Strike (<https://www.cobaltstrike.com/>) a été créé par Raphael Mudge en 2012 en tant que solution C2 commerciale destinée aux opérateurs lors de missions « Red Team ».

Il est accompagné de l'« Arsenal Kit », une suite d'outils personnalisés visant à renforcer les capacités des opérateurs pour la réussite de la mission ; cette suite inclut notamment un kit « sleep mask », des chargeurs personnalisés et diverses autres options.

Cobalt Strike est largement utilisé par les entreprises de cybersécurité pour les opérations « Red Team ». Ce C2 prend également en charge plusieurs opérateurs, les BOF, les profils malléables, les protocoles SOCKS4/SOCKS5 et les communications furtives.

Mythic (<https://docs.mythic-c2.net/>) repose sur une architecture « plug-and-play » permettant aux opérateurs de modifier les canaux de communication, les agents et d'autres paramètres en cours d'opération. C'est le seul framework C2 à prendre en charge des implants pour Linux, macOS et Windows, offrant ainsi une grande flexibilité aux opérateurs. De plus, Mythic gère le protocole SOCKS et les

communications sécurisées. Il se compose d'une interface web développée avec React et d'un backend conteneurisé via Docker. Une autre fonctionnalité majeure de Mythic est la possibilité de mapper les tâches exécutées durant une mission au framework MITRE (<https://attack.mitre.org>).

Le framework MITRE ATT&CK (pour *Adversarial Tactics, Techniques, and Common Knowledge*) est une base de connaissances accessible au public qui répertorie les comportements des acteurs de la menace. Ces comportements sont associés à des tactiques, techniques et procédures (TTP) spécifiques, permettant à toute organisation d'établir un langage commun entre ses équipes de sécurité offensive et défensive. La matrice ATT&CK regroupe plusieurs catégories de tactiques : reconnaissance, développement de ressources, accès initial, exécution, persistance, élévation de privilèges, contournement des défenses, accès aux identifiants, découverte, déplacement latéral, collecte, C2 (commande et contrôle), exfiltration et impact. Ces tactiques se déclinent en 191 techniques et 385 sous-techniques. Les opérateurs peuvent s'appuyer sur le framework et les correspondances ATT&CK pour affiner leurs stratégies de communication C2 ; en comprenant quelles TTP sont susceptibles d'être détectées lors d'une opération (sur la base des actions d'adversaires précédents), ils peuvent combler les lacunes pour opérer en toute discrétion.

Mythic propose divers profils communautaires (<https://github.com/MythicC2Profiles>) exploitables avec ce framework, notamment : • http : utilise des requêtes GET/POST vers le serveur Mythic ; prend en charge les dates d'arrêt (kill dates), les intervalles de veille, les en-têtes personnalisés, les paramètres de proxy et le protocole SSL. • dns : s'appuie sur un serveur DNS à l'écoute des requêtes TXT de l'agent pour les communications ; prend en charge les dates d'arrêt, les intervalles de veille et l'utilisation d'un serveur DNS personnalisé. • httpx : profil HTTP dynamique et personnalisable. Il offre un éventail d'options exploitables par l'opérateur, telles que les domaines de rappel (callback), les domaines de repli, la rotation des domaines, l'échange de clés chiffrées, le *jitter* (variation aléatoire des

délais), les intervalles de veille et les messages de l'agent. • discord : utilise l'API REST de Discord pour les communications. • websocket : permet aux agents de se connecter via une connexion WebSocket statique ; prend en charge SSL, les dates d'arrêt et les intervalles de veille. • dynamichttp : utilise des messages HTTP extensibles avec des modifications dynamiques côté client et serveur ; prend en charge les dates d'arrêt, les intervalles de veille, les en-têtes personnalisés, les paramètres de proxy et le protocole SSL.

Canaux de communication C2

Des canaux de communication de type « commande et contrôle » (C2) sont nécessaires pour permettre à un opérateur de communiquer avec le serveur C2. Ces canaux servent à transmettre et recevoir des données, ainsi qu'à maintenir un accès permanent à un agent installé sur la machine de la victime. Les méthodes de communication sont tout aussi cruciales que les tactiques d'évasion lors du déploiement de la charge utile. Une sécurité opérationnelle (OPSEC) défaillante concernant vos canaux de commande et de contrôle (C2) vous expose à une détection quasi certaine. Face à l'évolution des équipes de défense (Blue Teams), les opérateurs ont dû adapter leur trafic pour qu'il se fonde dans le trafic réseau existant. Aujourd'hui, les méthodes de communication C2 les plus répandues sont le DNS over HTTPS (DoH) et les protocoles HTTP/HTTPS, bien que le HTTPS s'impose globalement comme la méthode privilégiée. • Le DNS over HTTPS (DoH) assure un chiffrement de bout en bout, à l'instar du HTTPS. Les requêtes et réponses DNS transitent via HTTPS. Les opérateurs peuvent exploiter cette méthode pour contacter des fournisseurs DNS spécifiques (tels que cloudflare-dns.com) et résoudre les requêtes DNS au sein d'un canal chiffré. • Les protocoles HTTP et HTTPS sont couramment utilisés pour le trafic web sur Internet. La quasi-totalité des réseaux autorise les communications via ces protocoles, ce qui permet aux opérateurs de dissimuler leur trafic en les utilisant. Du point de vue de l'OPSEC, il est préférable d'éviter le HTTP, car le trafic n'est alors pas chiffré. Pour que votre trafic se fonde le mieux possible dans l'environnement, la solution la plus sûre consiste à garantir qu'il soit chiffré.

Redirecteurs

Les redirecteurs relaient le trafic entrant d'un point de vue externe et transmettent le trafic vers l'emplacement spécifié, qui peut être vers un hôte accessible uniquement via le redirecteur initial. Les redirecteurs sont utilisés pour le trafic C2 afin de masquer l'emplacement du serveur C2, de tromper le trafic pour qu'il paraisse légitime et également de protéger votre serveur C2 contre les brûlures. La beauté des redirecteurs est que même si l'un de vos redirecteurs est brûlé, votre serveur C2 est en sécurité et vous pouvez remplacer le redirecteur externe par un nouveau sans avoir besoin de réinitialiser aucune de votre infrastructure C2.

Les redirecteurs qui demanderont moins d'efforts, tels que socat ou iptables, transmettront aveuglément le trafic vers votre serveur C2, ce qui peut constituer un problème du point de vue de la sécurité opérationnelle, car les scanners Internet peuvent être capables d'identifier le trafic malveillant. Cependant, l'utilisation de redirecteurs plus sophistiqués qui exploitent des proxys inverses avec des règles permet de filtrer uniquement le trafic que vous souhaitez transmettre en tirant parti d'agents utilisateurs spécifiques, autorisant uniquement certaines adresses IP sources ou d'autres conditions spécifiques à protéger l'infrastructure.

Types de redirecteurs

Voici les types de redirecteurs, allant des méthodes de base aux méthodes plus sophistiquées :

- socat est un client utilisé pour transférer aveuglément toutes les connexions entrantes, connu sous le nom de trafic « dumb pipe ». L'outil établit des flux bidirectionnels entre l'adresse IP et le port spécifiques.
- iptables est généralement utilisé comme outil de pare-feu pour contrôler le trafic réseau et est un composant natif pour les réseaux Linux. Avec iptables, vous pouvez avoir plusieurs règles et configurations pour votre redirection de port.
- Les proxys inverses sont utilisés pour protéger votre backend C2 en tirant parti des configurations de serveur Web Apache ou Nginx pour transférer votre trafic vers le prochain serveur Web spécifié. Cela permet

de protéger l'identité du serveur backend et d'exploiter les certificats SSL d'un point de vue externe.

- Les réseaux de diffusion de contenu (CDN) sont un groupe de serveurs utilisés pour distribuer géographiquement le contenu de votre site afin d'offrir des performances plus rapides à un utilisateur final. Vous pouvez abuser des CDN en exploitant leurs adresses de distribution qui se trouveront devant votre domaine. Les CDN sont généralement utilisés parallèlement à une configuration de proxy inverse et filtrent les serveurs Web pour permettre uniquement aux adresses IP sources des CDN d'accéder au redirecteur externe.
- Les CDN avec reverse proxy utilisant des réécritures combinent un serveur web reverse proxy standard avec un CDN en face du serveur web et prennent en compte les réécritures spécifiques à la configuration Apache/Nginx. Ces réécritures peuvent contrôler la directive et manipuler l'URL à la volée en fonction des configurations en place.

Configuration de commande et de contrôle

Dans cette section, vous apprendrez comment configurer et exécuter des opérations avec le Mythic C2. La figure 3-1 montre un exemple de création d'infrastructure pour une équipe rouge typique.

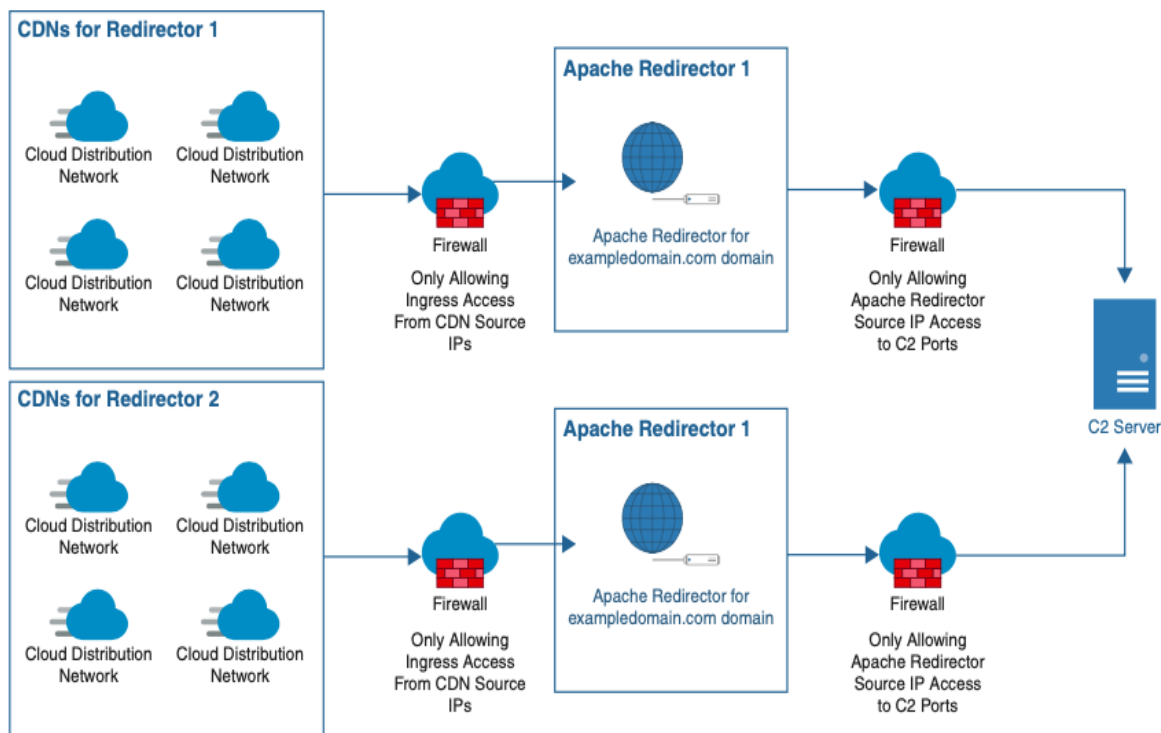


Figure 3-1
C2 Infrastructure Setup

Pour commencer, vous aurez besoin de la configuration suivante :

- Serveur Azure Ubuntu (B1s) servant de redirecteur
- Serveur Azure Ubuntu (B2ms) servant de point de terminaison C2
- CDN Azure pour le domaine du redirecteur
- Poste de l'opérateur (machine virtuelle ou instance cloud)
- Poste de test Windows (avec Windows Defender désactivé)

Redirecteur Apache 1

Redirecteur Apache pour le domaine exampledomain.com

Redirecteur Apache 1

Redirecteur Apache pour le domaine exampledomain.com

Une fois le déploiement effectué, nous configurerons le serveur Apache pour le domaine utilisé par le redirecteur. Un article de blog de l'équipe « Attack & Penetration » d'Optiv (archivé à l'adresse https://lazzslayer.github.io/References/Redirectors/Redirectors_Optiv.pdf)

f) propose des configurations Apache faciles à utiliser pour votre serveur. Vous devrez également configurer les enregistrements A pour les domaines utilisés par l'infrastructure. Pour la génération des certificats SSL, il faudra ouvrir temporairement les ports 80 et 443 vers l'extérieur afin de récupérer les certificats du domaine, puis restreindre à nouveau l'accès pour n'autoriser que les adresses IP sources du CDN Azure.

Redirecteur Apache

Les exemples 3-1 à 3-4 détaillent la procédure de configuration de votre serveur Apache pour le redirecteur, de l'installation jusqu'au paramétrage du serveur.

EXAMPLE 3-1 Installing Apache

```
sudo apt update
sudo apt full-upgrade
apt-get install apache2
systemctl stop apache2.service
a2enmod ssl proxy remoteip rewrite proxy_http
```

EXAMPLE 3-2 Installing Certificates for Your Domain

```
apt-get install certbot

certbot certonly --standalone --preferred-challenges http -d exampledomain.com
--register-unsafely-without-email --agree-tos
```

EXAMPLE 3-3 Apache Config for Redirector

```
<VirtualHost *:80>
ServerName exampledomain.com
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
<VirtualHost *:443>
ServerName exampledomain.com
RemoteIPHeader X-Forwarded-For

ProxyPass / https://exampledomain.com/
ProxyPassReverse / https://exampledomain.com/

SSLProxyEngine On
SSLProxyVerify none
SSLProxyProtocol TLSv1.2

SSLCertificateFile /etc/letsencrypt/live/exampledomain.com/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/exampledomain.com/privkey.pem

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

EXAMPLE 3-4 Adding in the IP of Your C2 Server in /etc/hosts

```
echo <C2ServerIP> <C2Domain> >> /etc/hosts
```

Maintenant que votre redirecteur est configuré, vous pouvez le paramétrer pour n'autoriser que le CDN d'Azure (via HTTP et HTTPS) en utilisant la balise de service Azure « AzureFrontDoor.Backend ». La figure 3-2 montre comment effectuer cette mise à jour pour le protocole HTTP, mais veillez à mettre à jour les règles pour les protocoles HTTP et HTTPS.

 **AllowAnyHTTPInbound** Server-nsg ×

Source ⓘ

Service Tag ▼

Source service tag * ⓘ

AzureFrontDoor.Backend ▼

Source port ranges * ⓘ

*

Destination ⓘ

Any ▼

Service ⓘ

HTTP ▼

Destination port ranges ⓘ

80

Protocol

☐ Any

☒ TCP

☐ UDP

☐ ICMP

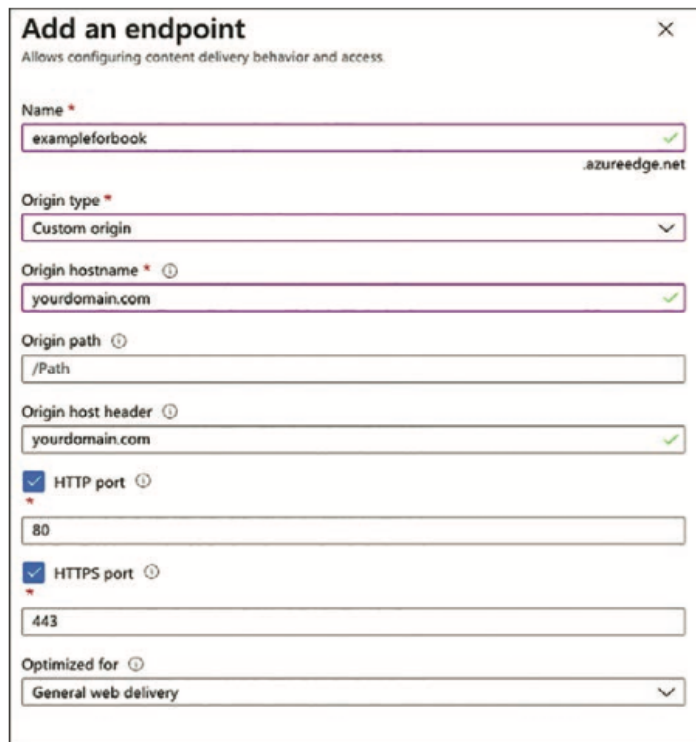
Action

☒ Allow

☐ Deny

Figure 3-2
Filtering HTTP

Next, you will create a CDN for your redirector domain with Azure Classic CDN, as demonstrated in Figure 3-3.



Add an endpoint ✕

Allows configuring content delivery behavior and access.

Name *
exampleforbook ✓
.azureedge.net

Origin type *
Custom origin ▼

Origin hostname * ⓘ
yourdomain.com ✓

Origin path ⓘ
/Path

Origin host header ⓘ
yourdomain.com ✓

☒ HTTP port ⓘ
80

☒ HTTPS port ⓘ
443

Optimized for ⓘ
General web delivery ▼

Figure 3-3
Creating CDN Endpoint

Ensuite, vous configurerez les paramètres de compression et de mise en cache de votre point de terminaison CDN, comme illustré à la figure 3-4. Il est recommandé de désactiver ces deux paramètres afin d'éviter tout problème de communication avec votre serveur final.

Serveur C2

Avant de configurer Mythic, assurez-vous que l'enregistrement A du domaine utilisé pointe vers l'adresse IP du serveur C2. Pour la génération du certificat SSL, vous devrez également ouvrir temporairement les ports 80 et 443 au trafic externe afin de récupérer les certificats du domaine. Une fois cette étape terminée, vous modifierez les règles du pare-feu pour n'autoriser que le trafic provenant de l'adresse IP de votre redirecteur.

L'exemple 3-5 montre comment installer des certificats Let's Encrypt pour votre domaine sur votre serveur C2 et les déplacer dans votre répertoire Mythic. La variable ``$MYTHICDIRECTORY`` doit correspondre à l'emplacement du répertoire Mythic sur votre serveur C2.

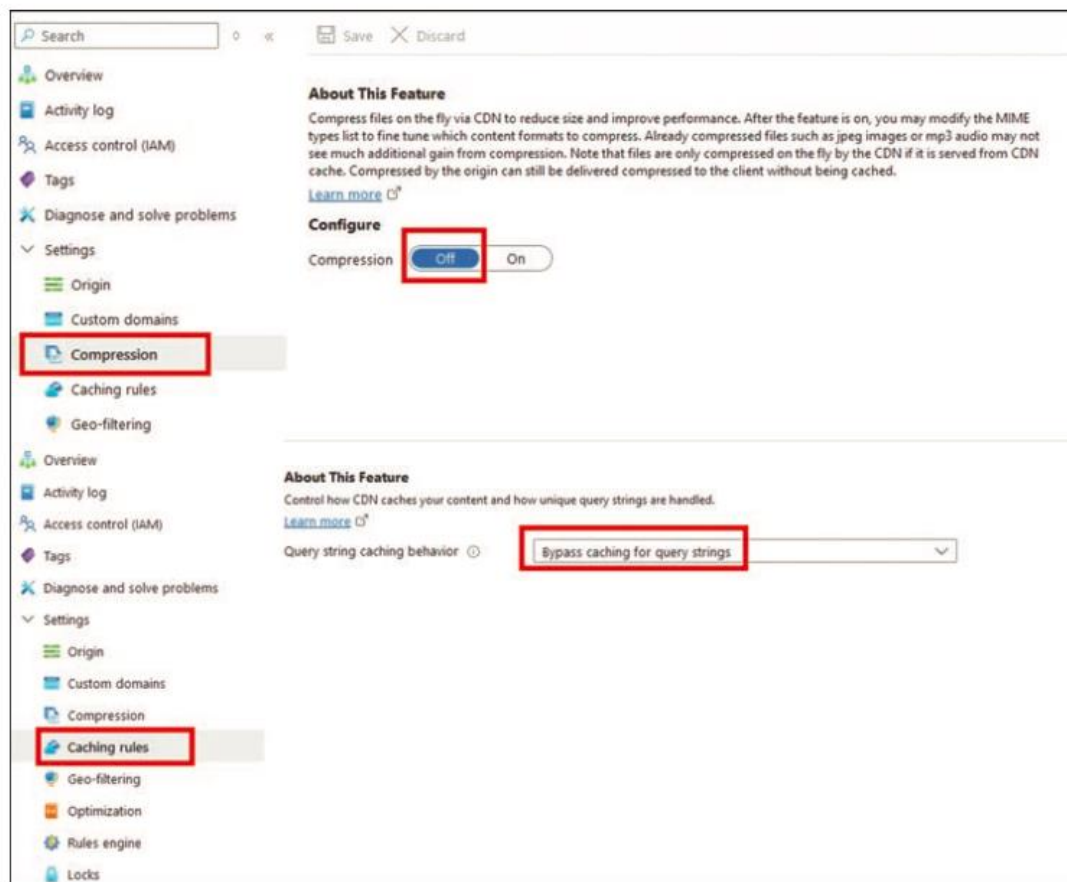


Figure 3-4
Compression and Caching Configurations

EXAMPLE 3-5 Installing Certificates for Your Domain

```
MYTHICDIRECTORY=(/opt/Mythic)
sudo apt-get install certbot
certbot certonly --standalone --preferred-challenges http -d
  exampledomain.com --register-unsafely-without-email --agree-tos
cp /etc/letsencrypt/live/exampledomain.com/privkey.pem $MYTHICDIRECTORY
cp /etc/letsencrypt/live/exampledomain.com/fullchain.pem $MYTHICDIRECTORY
```

Une fois les certificats générés, restreignez l'accès à votre instance pour n'autoriser que l'adresse IP de votre redirecteur via HTTPS (port 443). Cette mesure empêchera les scanners d'atteindre directement votre serveur C2. Si l'instance cloud hébergeant votre C2 est identifiée comme malveillante et reste accessible depuis Internet, vous risquez une suspension de l'instance par le fournisseur cloud, tout en déclenchant des alertes lors du scan de votre hôte.

Par ailleurs, vous pouvez désormais supprimer l'enregistrement A associé à votre domaine C2, car la configuration illustrée dans l'exemple 3-5 indique au serveur redirecteur la destination du domaine C2. Ensuite, vous pouvez installer Mythic en suivant les étapes décrites dans l'exemple 3-6.

EXAMPLE 3-6 Installing Mythic

```
git clone https://github.com/its-a-feature/Mythic
cd /Mythic
./install_docker_ubuntu.sh
```

Par défaut, Mythic n'inclut aucun profil C2 ni agent. Pour installer des agents et des profils spécifiques, vous pouvez utiliser les différentes options disponibles aux emplacements suivants :

- Agents Mythic : <https://github.com/MythicAgents>
- Profils C2 Mythic : <https://github.com/MythicC2Profiles>

Pour ce guide pratique, vous pouvez installer le profil C2 « http » et l'agent Apollo, comme illustré dans l'Exemple 3-7. Nous utiliserons le profil HTTP pour cette démonstration car il repose sur le même protocole que le serveur Mythic, contrairement aux autres profils.

Le profil HTTP permet notamment de définir des dates d'expiration (kill dates) et des intervalles de veille (sleep intervals), ainsi que de personnaliser les en-têtes, les paramètres de proxy et la prise en charge SSL. L'agent Apollo est basé sur .NET et offre des fonctionnalités telles que le chargement dynamique de commandes, la prise en charge de SOCKS5, la communication pair-à-pair, l'exécution de code non managé, ainsi que de nombreuses autres fonctions utiles pour un opérateur. L'agent Apollo est également utilisé lors des formations annuelles de SpecterOps (<https://specterops.io/>). Les Exemples 3-7 et 3-8 montrent comment installer votre profil et votre agent, puis démarrer le serveur Mythic.

EXAMPLE 3-7 Installing the C2 Profile and Agent

```
sudo ./mythic-cli install github https://github.com/MythicAgents/Apollo.git  
sudo ./mythic-cli install github https://github.com/MythicC2Profiles/http
```

EXAMPLE 3-8 Starting Mythic

```
# Start Mythic Client  
sudo -E ./mythic-cli start
```

Pour accéder à l'interface graphique de Mythic, vous pouvez mettre en place un transfert de port SSH local depuis votre poste d'opérateur à l'aide de la commande présentée dans l'Exemple 3-9.

Conseil

Il est recommandé d'utiliser un poste d'opérateur dédié pour chaque mission lors de la conduite des opérations. Une machine virtuelle Ubuntu ou une instance cloud suffit pour cet usage.

EXAMPLE 3-9 Accessing the GUI

```
# SSH Port Forwarding  
ssh -L 7443:127.0.0.1:7443 user@mythicserver
```

Once the forwarding is in effect, navigate to <https://127.0.0.1:7443> to log in (see Figure 3-5). You can find the mythic_admin password in the `.env` file of the Mythic directory.

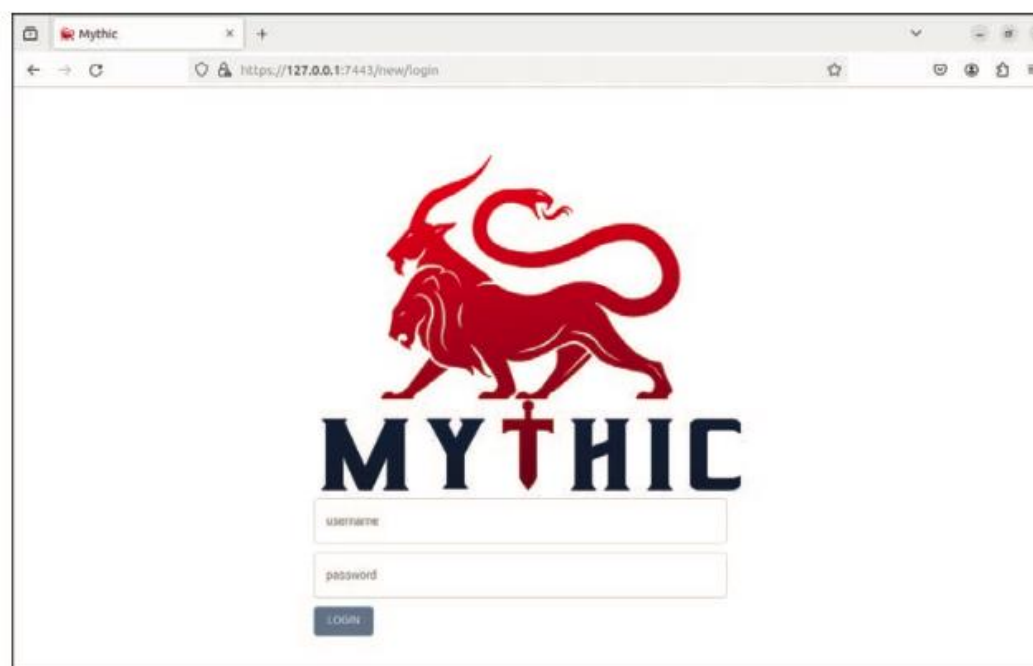


Figure 3-5
Mythic Login

Une fois connecté, accédez à la section « Agents & C2 » dans le panneau de gauche, comme illustré à la figure 3-6. Vous devrez modifier le profil C2 pour y intégrer les certificats générés précédemment.

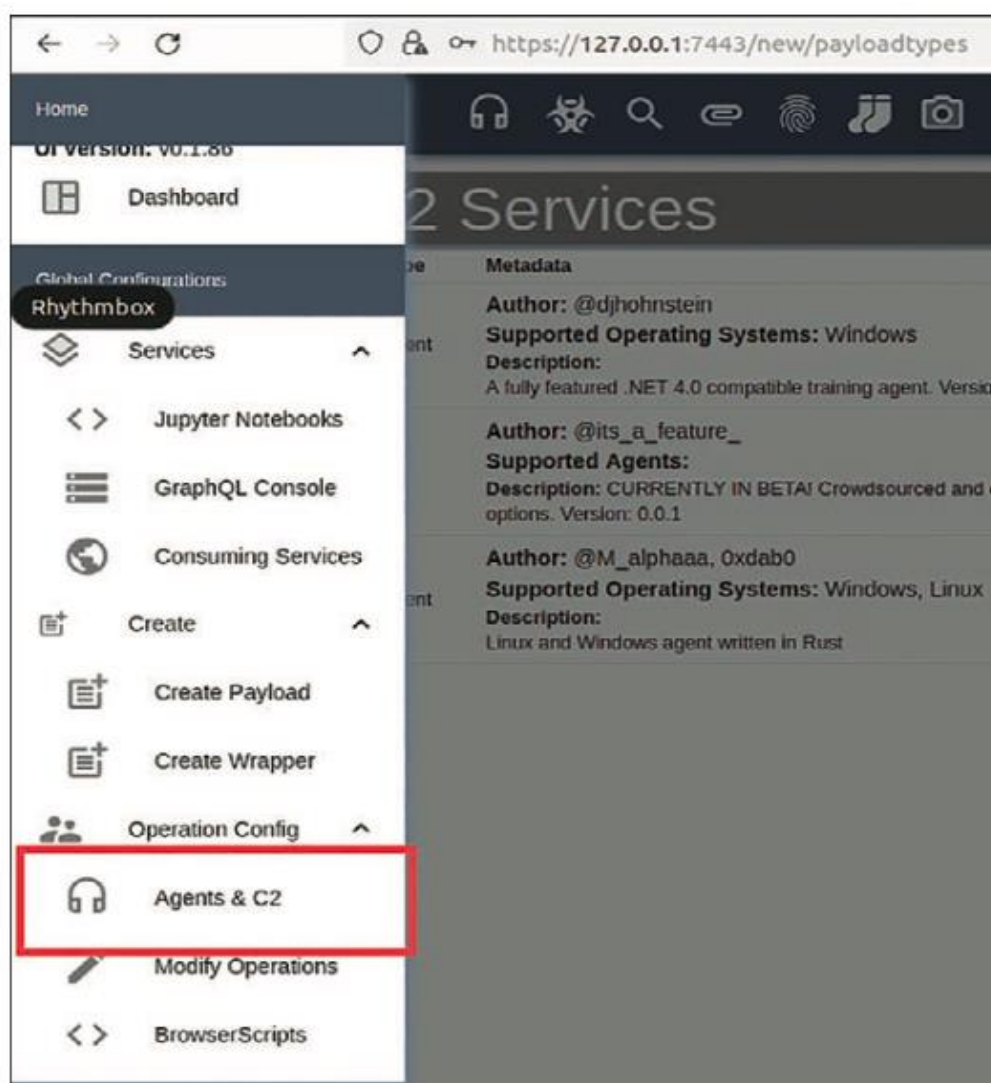


Figure 3-6
Updating the C2 Profile Config

Avant de modifier la configuration du profil, vous devrez en mettre à jour les valeurs : cliquez sur l'icône du presse-papiers, puis modifiez les fichiers `fullchain.pem` et `privkey.pem` en y ajoutant les certificats de domaine générés à l'étape de l'exemple 3-5.

Il vous suffit de copier le contenu du fichier `/etc/letsencrypt/live/yourdomain.com/fullchain.pem` et de l'insérer dans le fichier `fullchain.pem`. Procédez de la même manière pour le fichier `privkey.pem` en utilisant le contenu de `/etc/letsencrypt/live/yourdomain.com/privkey.pem`. La figure 3-7 indique où trouver ces fichiers dans la configuration C2.

Ensuite, modifiez la configuration pour éviter d'utiliser les options par défaut. Vous pouvez personnaliser cette partie à votre convenance, comme le montre la figure 3-8. Pour cet exemple, vous pouvez utiliser des en-têtes similaires à ceux de « Microsoft-IIS/10.0 », comme illustré dans l'exemple 3-10.

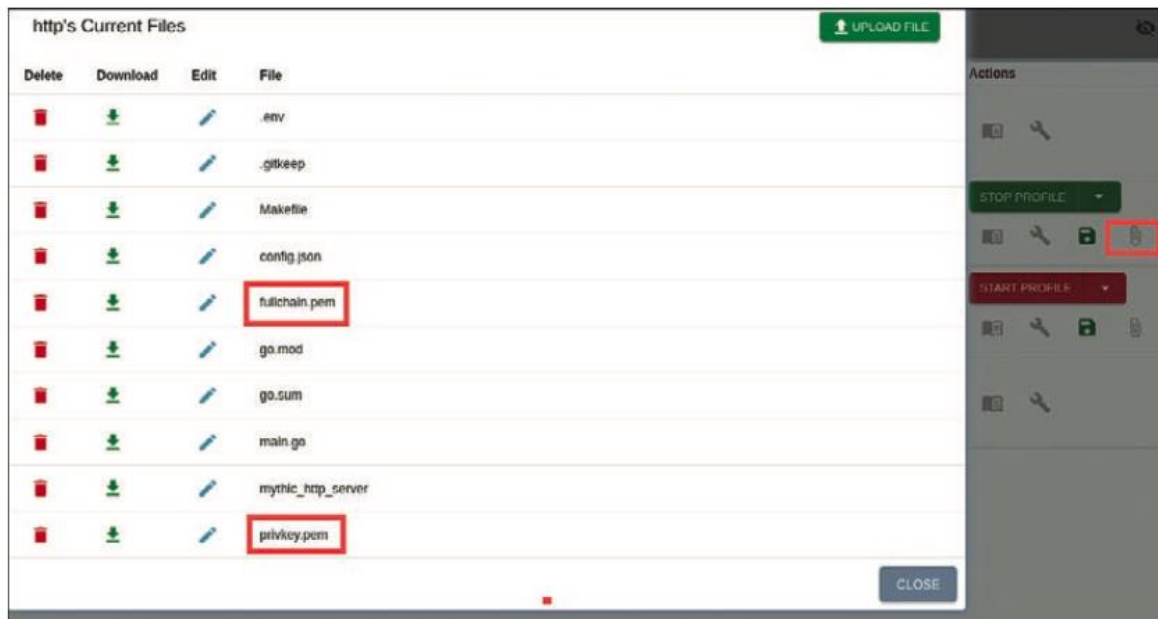


Figure 3-7
Updating Certificate Files for the C2 Config

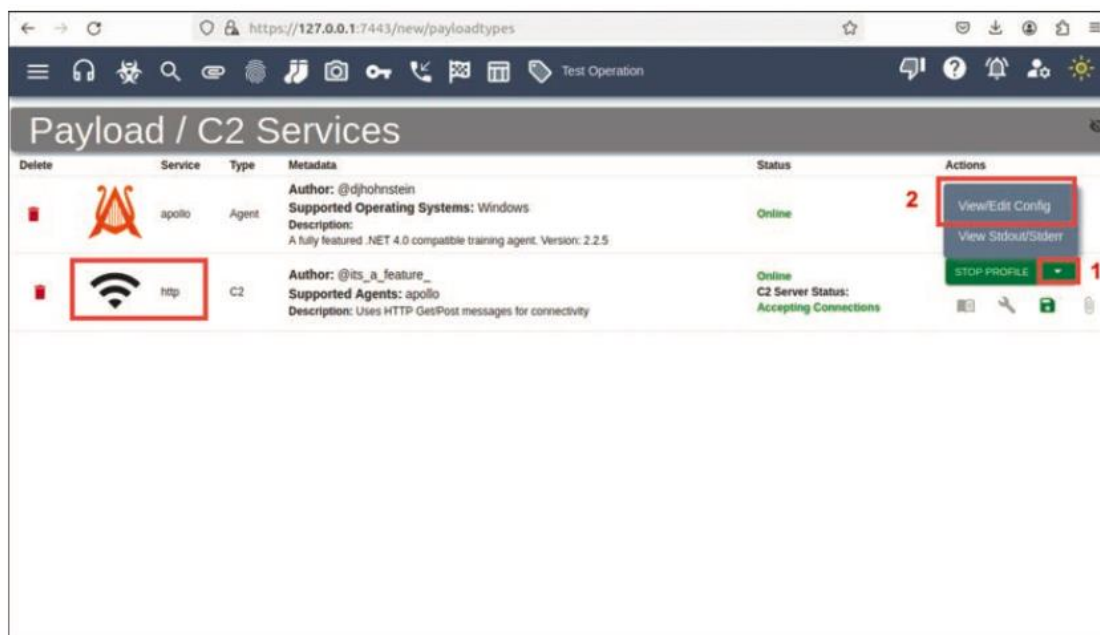


Figure 3-8
Editing the Config

EXAMPLE 3-10 http Config

```
{
  "instances": [
    {
      "ServerHeaders": {
        "Server": "Microsoft-IIS/10.0",
        "Cache-Control": "max-age=0, no-cache",
        "Pragma": "no-cache",
        "Connection": "keep-alive",
        "Content-Type": "text/html; charset=utf-8"
      },
      "port": 443,
      "key_path": "privkey.pem",
      "cert_path": "fullchain.pem",
      "debug": false,
      "use_ssl": true
    }
  ]
}
```

Next, you need to stop and start your profile to ensure the changes went into effect (see Figure 3-9).



Figure 3-9
Starting and Stopping the Profile

Vous pouvez ensuite tester le flux de trafic en sollicitant le point de terminaison de votre CDN. Dans ce cas, vous devriez recevoir une réponse 405, ce qui indique que le trafic transite correctement (puisque vous ne spécifiez pas de point de terminaison précis sur le serveur) tout en confirmant que le trafic circule bien de bout en bout, comme illustré dans l'exemple 3-11.

EXAMPLE 3-11 Checking the Connection from End to End

```
curl -I https://examplenameplaceholder.azureedge.net
HTTP/2 405
date: Sun, 23 Jun 2024 20:20:25 GMT
content-type: text/plain

content-length: 22
x-azure-ref: <snipped>
x-fd-int-roxy-purgeid: 0
x-cache: TCP_MISS
```

Now, you can generate payloads for Mythic by using the Apollo agent. Select the Windows option for the Target Operating System, as illustrated in Figure 3-10.

The screenshot shows a web interface with a dark blue header containing various icons. Below the header is a progress bar with two steps: '1 Select Target OS' and '2 Payload Type'. The main heading is 'Select Target Operating System'. A dropdown menu is set to 'Windows'. Below it, the text 'Target Operating System' is displayed. At the bottom, there are 'BACK' and 'NEXT' buttons.

Figure 3-10
Selecting the Target OS

The Target Payload Type will be *apollo* and the output will be *WinExe*, as illustrated in Figure 3-11.

The screenshot shows a web interface with a dark blue header. Below the header is a progress bar with three steps: '1 Select Target OS', '2 Payload Type', and '3 Select Command'. The main heading is 'Select Target Payload Type'. A dropdown menu is set to 'apollo'. Below this is a section titled 'Build Parameters' with a table. The table has two columns: 'Build Parameter' and 'Value'. The first row shows 'Output as shellcode, executable, or dynamically loaded library.' and 'WinExe'. At the bottom, there are 'BACK' and 'NEXT' buttons.

Build Parameter	Value
Output as shellcode, executable, or dynamically loaded library.	WinExe

Figure 3-11
Output Option–Executable

Ultimately, an operator determines which commands are built into the agent and can add or remove which ones are included in the payload (see Figure 3-12).

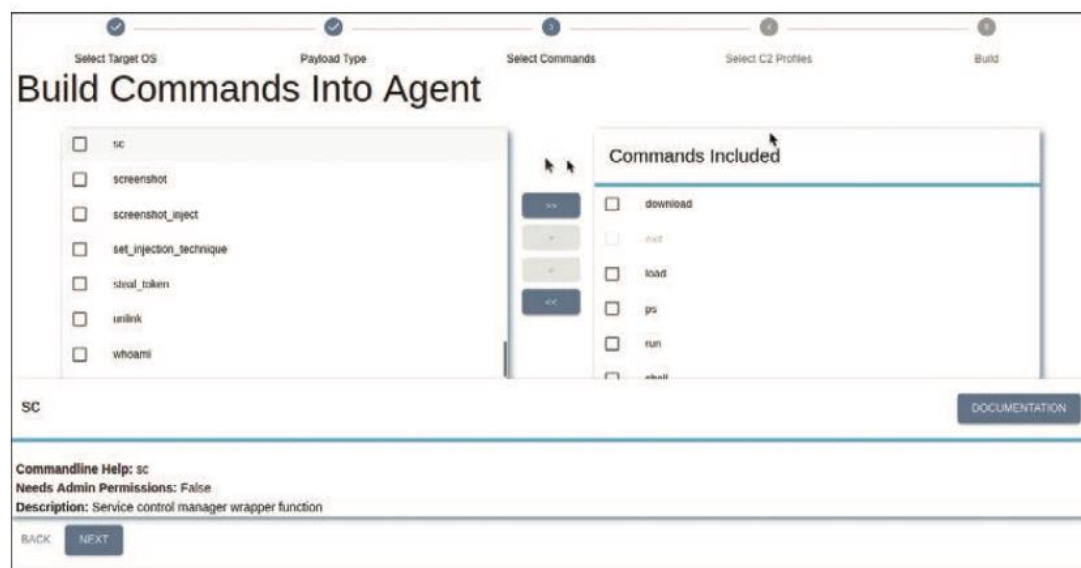


Figure 3-12
Commands Built Into Payload

La section « C2 Profile » permet de personnaliser les requêtes GET et POST. Les URI peuvent être adaptés pour imiter des requêtes courantes (similaires à celles des serveurs IIS) ou pour simuler un autre type de trafic afin de se fondre dans le réseau. Il est déconseillé d'utiliser les options par défaut de cette section. Plusieurs modifications doivent être effectuées dans cette section, comme l'illustre la figure 3-13 :

- **Callback Host** : L'hôte vers lequel le trafic doit se connecter (rappel) depuis l'extérieur. Il s'agit du premier point de contact vu de l'extérieur.
- **Callback Interval** : La fréquence (en secondes) à laquelle l'agent établit une connexion de rappel ; cet intervalle est fixe.
- **Callback Jitter** : Le pourcentage de variation aléatoire (*jitter*) appliqué aux rappels de l'agent ; cela introduit de l'imprévisibilité dans les connexions, faisant varier le moment de chaque signalement de l'agent.
- **Callback Port** : Le port de l'hôte utilisé pour la connexion de rappel.
- **Encryption Type** : La méthode de chiffrement utilisée pour le trafic.
- **Headers** : Personnalisation des en-têtes des messages GET/POST.
- **Kill Date** : Date limite au-delà de laquelle l'agent cesse d'effectuer des rappels.

La section « Payload Review » permet à l'opérateur de personnaliser le nom de la charge utile (*payload*) finale et d'y ajouter une description, comme le montre la figure 3-14. L'option de description est utile pour

créer des charges utiles spécifiques destinées à des cibles précises au cours d'une opération.

Select C2 Profiles

Include?	C2 Name	Pre-created Instances	Description
☒	http		Uses HTTP Get/Post messages for connectivity
Parameter		Value	
Callback Host Modified		https://[REDACTED].zureedge.net	
Callback Interval in seconds Modified		30	
Callback Jitter in percent Modified		20	
Callback Port Modified		443	

Figure 3-13
Modifying C2 Profile Options

☒ Select Target OS

☒ Payload Type

☒ Select Co

Payload Review

testing.exe

description

BACK

CREATE PAYLOAD

START OVER

CREATE WRAPPER

Figure 3-14
Customizing the Output Filename

All generated payloads for the operation you are currently using on Mythic are available at <https://127.0.0.1:7443/new/payloads>, as illustrated in Figure 3-15.

← → ↻ https://127.0.0.1:7443/new/payloads

⋮ 🔊 🔍 📎 🖨️ 📷 🔑 📞 📧 📁 Test Operation

Payloads

Delete	Modify	Build Progress	Download	File	Description
	ACTIONS	🟢🟢🟢		testing.exe	Created by mythic_admin at 2024-06-16 22:33:03 Z

Figure 3-15
Payloads Page

You can download the payload locally to your machine and transfer it over to your testing Windows box, as illustrated in Figure 3-16 and Figure 3-17.

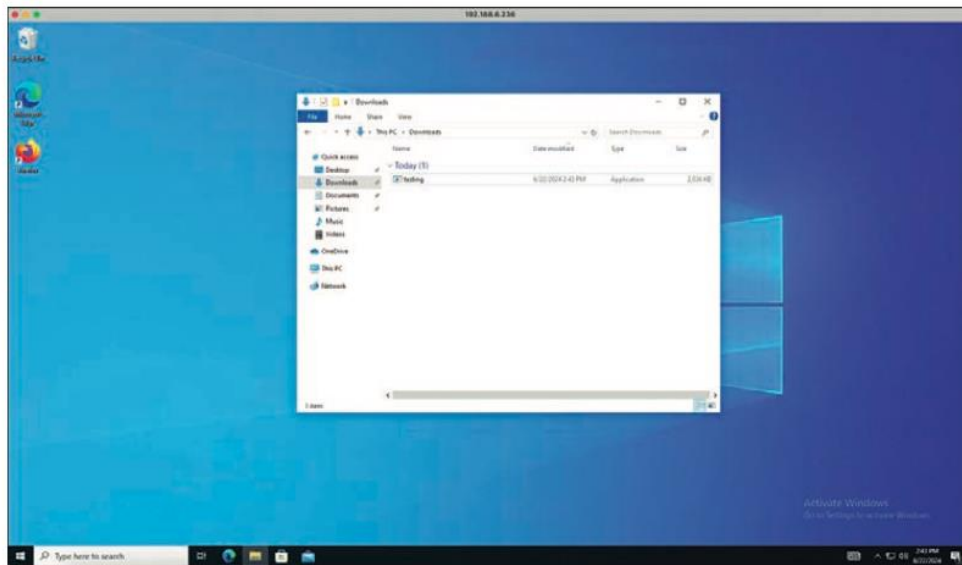


Figure 3-16
Transferred testing.exe Payload on the Windows Box

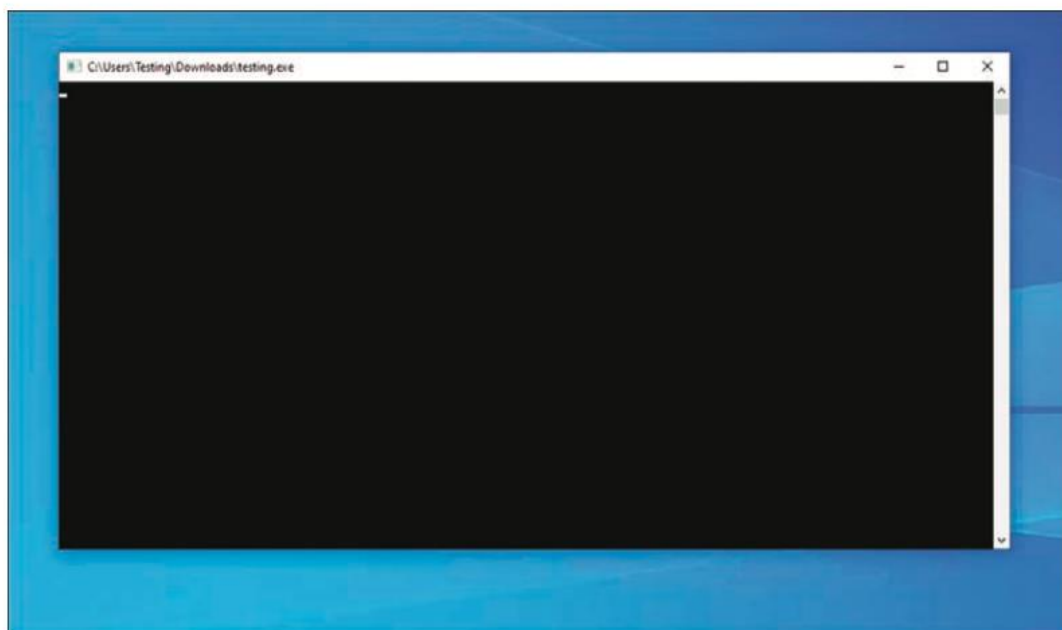


Figure 3-17
Executing testing.exe on the Windows Box

Lors de son exécution, un agent a établi une connexion de rappel vers le serveur C2, comme l'illustre la figure 3-18.

Remarque

Dans un scénario réel, vous devriez modifier l'agent C2, car sa signature est très reconnaissable, et apporter des modifications supplémentaires

pour obfusquer votre shellcode en vue de sa transmission, afin de contourner les protections en place sur le poste cible. Nous aborderons les différentes options d'obfuscation du shellcode au chapitre 4.

INTERACT	IP	HOST	1 USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION
 5	192.168.6.236		Testing		2252	34 seconds	Created by mythic_admin at

Figure 3-18
Mythic Callback

Dans l'agent Mythic, cliquez sur l'option clavier pour interagir avec l'agent. Tapez ensuite help dans la console pour voir quelles options de commande l'agent a la possibilité d'utiliser sur l'hôte victime, comme le montre la figure 3-19. N'hésitez pas à essayer différentes options de commande à ce stade et à prendre en compte celles qui peuvent être les plus sûres du point de vue OPSEC et les commandes qu'il est préférable d'éviter pour une opération.

```
CALLBACK: 5 X
1 Loaded Commands In Agent:
2
3 cat
4   Usage: cat [file]
5   Description: Print the contents of a file specified by [file]
6 cd
7   Usage: cd [path]
8   Description: Change directory to [path]. Path relative identifiers such as ../ are accepted.
9 cp
10  Usage: cp [source] [dest]
11  Description: Copy a file from one location to another.
12 download
13  Usage: download -Path [path/to/file] [-Host [hostname]]
14  Description: Download a file off the target system.
15 exit
16  Usage: exit
17  Description: Task the implant to exit.
18 ifconfig
19  Usage: ifconfig
20  Description: Get interface information associated with the target.
```

Task an agent...

Figure 3-19
Mythic help Output

L'option SOCKS sera la plus utile pour un opérateur accédant à une machine cible, car elle lui permet d'exploiter la connexion interne de cette dernière et d'acheminer le trafic via la machine cible à l'aide d'un proxy SOCKS5. L'exemple 3-12 montre comment installer proxychains4

sur le serveur, démarrer le serveur SOCKS, puis ajuster les paramètres de délai (sleep) et de gigue (jitter) de l'agent.

EXAMPLE 3-12 Setup Proxy

```
# On the C2 Server, install proxychains4
apt-get install proxychains4

# On the C2 Server, edit /etc/proxychains4.conf
socks5 127.0.0.1 7006

# In the Mythic console, start the SOCKS5 proxy on Mythic agent
socks 7006

# In the Mythic console, set the agent to interactively callback with sleep
sleep 0 0
```

You can find the SOCKS proxies by clicking the Socks icon (see Figure 3-20); they also are available at <https://127.0.0.1:7443/new/search?tab=socks>.

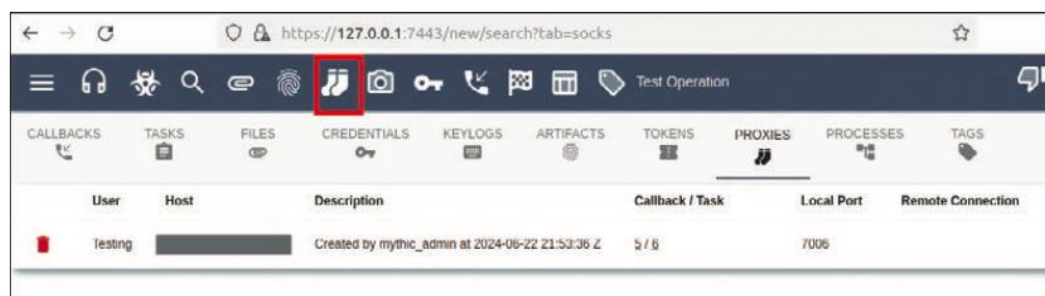


Figure 3-20
Mythic SOCKS Proxies

Pour tester le proxy, accédez au serveur C2 et récupérez votre adresse IP via ipconfig.io. Dans l'exemple de la figure 3-21, la comparaison entre la commande `curl` sans `proxychains4` et avec `proxychains4` démontre que le trafic est correctement acheminé via la machine de la victime lorsque `proxychains4` est utilisé.

```

azureuser@EndServer:/opt$ sudo curl ipconfig.io
3.14 Azure Instance IP
azureuser@EndServer:/opt$ sudo proxychains4 curl ipconfig.io
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
[proxychains] Strict chain ... 127.0.0.1:7006 ... ipconfig.io:80 ... OK
. 255 Proxying Traffic Through Victim's IP

```

Figure 3-21
Testing Proxy

Conseil

Pour une opération réelle, utilisez une machine d'opérateur distincte du serveur C2 et configurez `proxychains4` sur cet hôte. Dans le fichier `/etc/proxychains4.conf`, ajoutez la ligne `socks5` comme indiqué dans l'exemple 3-12, mais remplacez `127.0.0.1` par l'adresse IP publique ou privée du serveur C2, selon l'emplacement de votre machine d'opérateur. Vous devrez ensuite ouvrir ce port spécifique pour votre machine d'opérateur depuis votre serveur C2 via des règles de pare-feu. Ainsi, vous éviterez toute attribution à votre serveur C2 en cas de fuite de l'adresse IP.

L'utilisation d'un proxy SOCKS est extrêmement utile pour les opérations, car elle permet d'éviter d'exécuter quoi que ce soit directement sur l'agent, ce qui est préférable du point de vue de l'OPSEC. De nombreux outils peuvent être utilisés via un proxy SOCKS, notamment :

- Certipy (<https://github.com/ly4k/Certipy>)
- SCCMHunter (<https://github.com/garrettfoster13/sccmhunter>)
- Impacket (<https://github.com/fortra/impacket>)
- Idapper (<https://github.com/Synzack/Idapper>)
- Vibe (<https://github.com/Tylous/Vibe>)

Nous examinerons en détail les outils à utiliser avec le proxy une fois l'accès initial obtenu, au chapitre 4.

Un autre aspect notable de Mythic, en matière de documentation, est la possibilité de faire correspondre les tâches exécutées lors d'une opération à la matrice MITRE ATT&CK grâce à la fonction de rapport de

Mythic. Vous pouvez accéder à cette option via l'icône « Colonne » ou à l'adresse <https://127.0.0.1:7443/new/mitre>. Une fois sur la page, vous pouvez récupérer toutes les commandes possibles de vos agents et les tâches associées à MITRE via l'onglet « Actions », comme illustré à la figure 3-22. Les commandes possibles sont celles intégrées à votre agent, mais pas nécessairement les tâches effectivement exécutées.

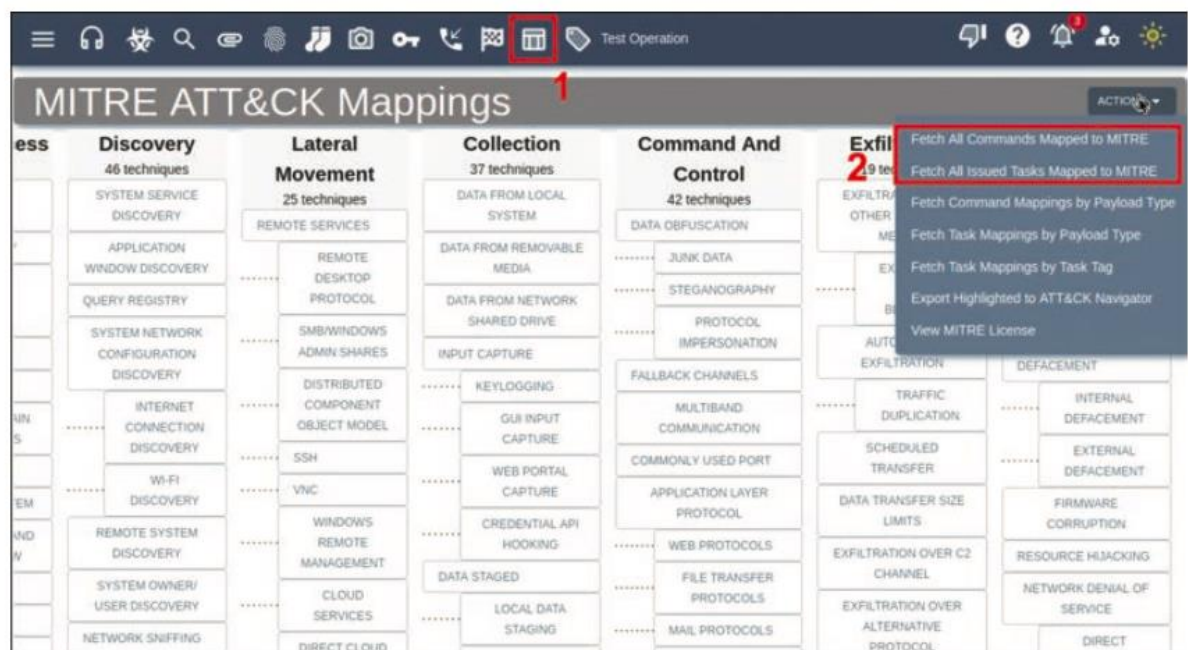


Figure 3-22
MITRE ATT&CK Mappings

Pour l'action « Fetch All Issued Tasks Mapped to MITRE » (Récupérer toutes les tâches émises associées à MITRE), les associations s'affichent en rouge, indiquant leur correspondance. Dans l'exemple illustré par la figure 3-23, seule l'association « Scheduled Transfer » (Transfert planifié) apparaît pour les options de mise en veille (« sleep options ») utilisées précédemment.

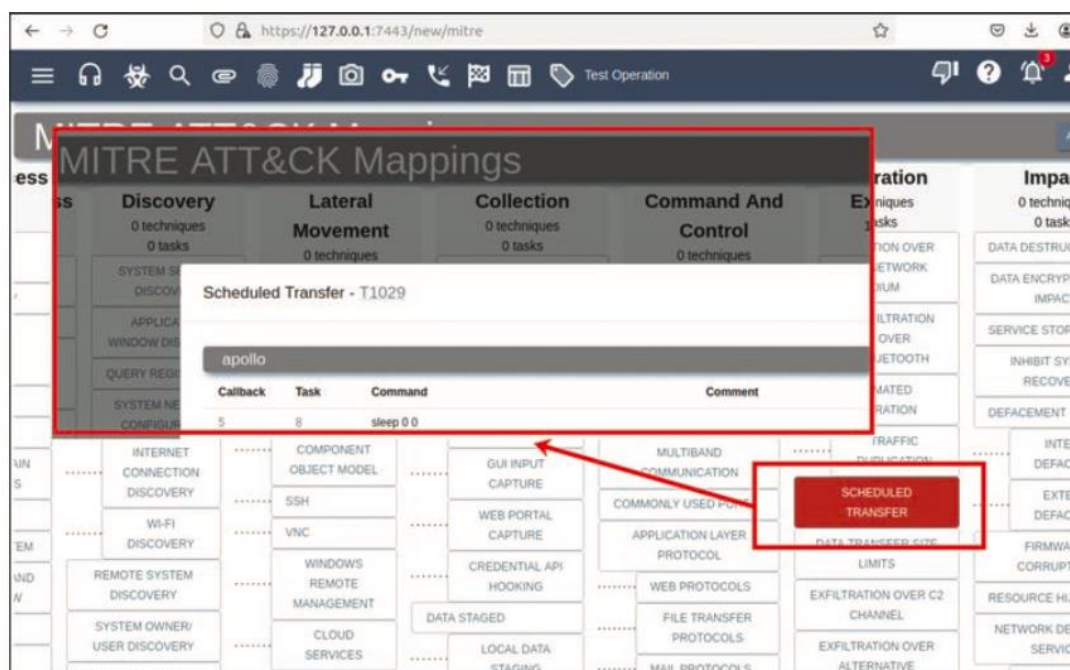


Figure 3-23

Fetch All Issued Tasks Mapped to MITRE Option

Pour avoir un aperçu des associations liées aux agents, vous pouvez sélectionner l'action « Fetch All Commands Mapped to MITRE » (Récupérer toutes les commandes associées à MITRE). Cette action a pour but d'afficher uniquement les associations potentielles basées sur les commandes intégrées à l'agent que vous utilisez, comme l'illustre la figure 3-24.

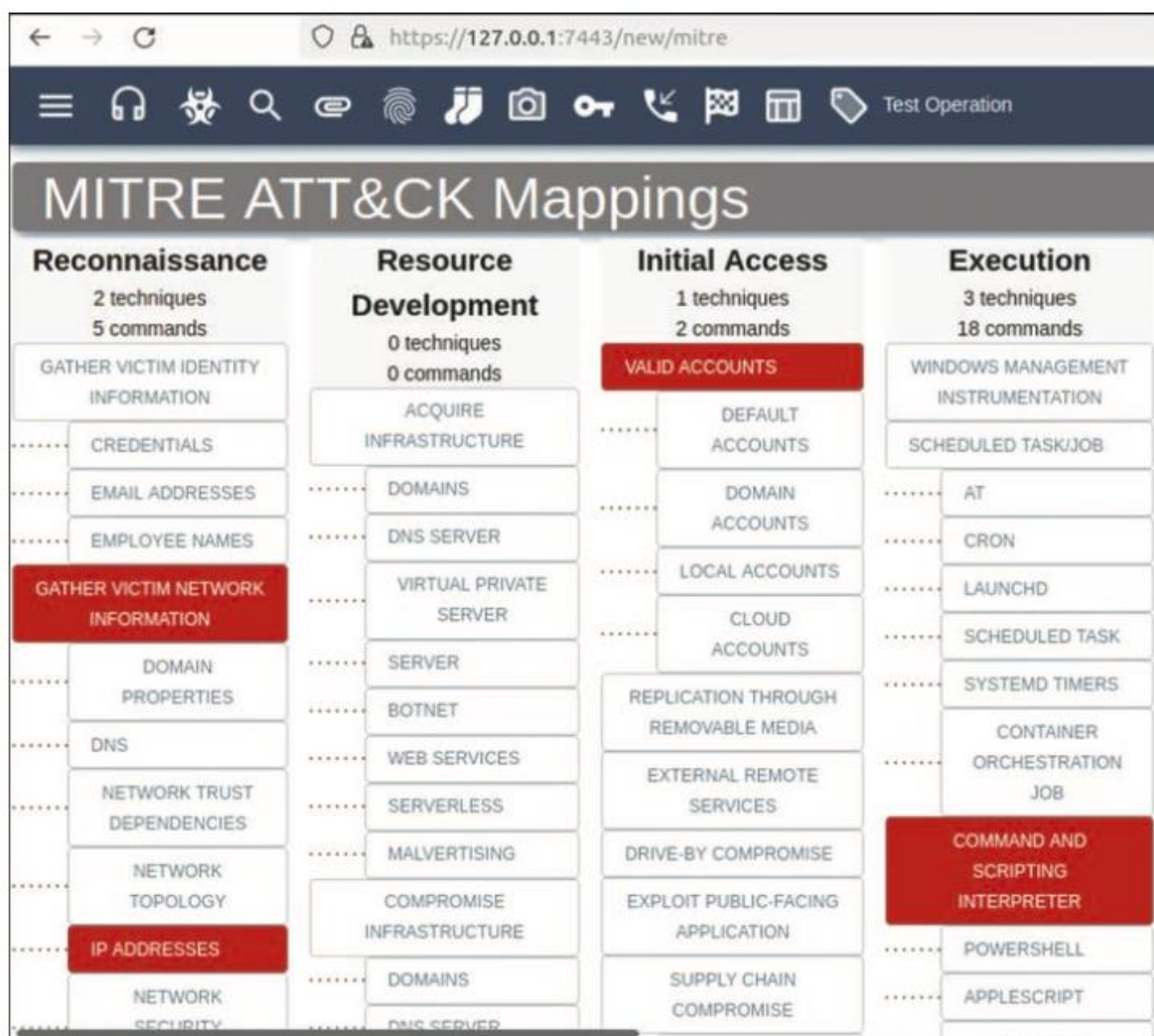


Figure 3-24

Possible Commands from Apollo Agent Mapped to MITRE

In Figure 3-25, the Apollo agent has an **ifconfig** command option, which maps back to T1590.005–Gather Victim Network Information: IP Addresses.

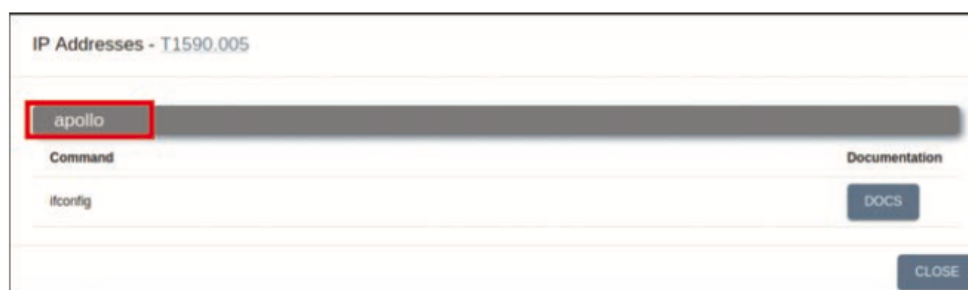


Figure 3-25

Command Mapped to T1590.005

Pour générer un rapport, accédez au drapeau (flag) ou rendez-vous manuellement à l'URL suivante : <https://127.0.0.1:7443/new/reporting>.

Vous pouvez choisir de générer le rapport au format HTML ou JSON. Dans cet exemple (voir les figures 3-26 et 3-27), nous générerons un rapport HTML. Il est également recommandé d'inclure les options de couverture MITRE. Si nécessaire, vous pouvez choisir d'exclure certains noms d'hôtes ou utilisateurs.

Mythic Report Generation

Report Option

Output Format: html

Selected Values

Include MITRE ATT&CK Coverage Per Task: ☒

Include MITRE ATT&CK Coverage Overview: ☒

Exclude Users: Excluded Usernames

Exclude Hosts: Excluded Hostnames

Exclude IDs: Excluded Callback IDs

GENERATE

Figure 3-26
Mythic Report Creation

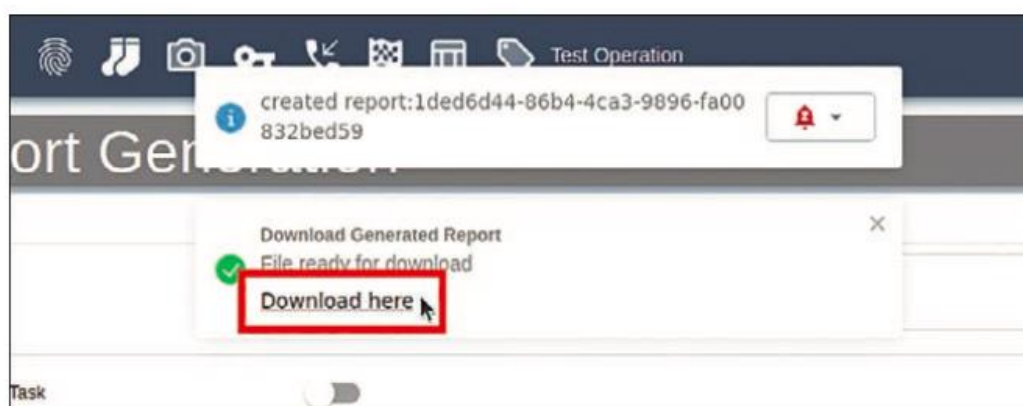


Figure 3-27
Downloading a Mythic Report

Once the report is downloaded, you have detailed information on the operation metrics, callback, MITRE mappings, and artifacts, as illustrated in Figure 3-28.

	
1. Assigned Operators	
The following table lists out all of the operators assigned to the operation and their roles in the assessment.	
Operator	Role
mythic_admin	lead
2. Operation Metrics	
The following table lists out some metrics for the operation.	
Metric	Value
Compromised Users	[REDACTED] testing
Compromised Hosts	[REDACTED]
Domains Accessed	[REDACTED]
Total Callbacks	5
Total High Integrity Callbacks	0
Total Tasks Issued	8
3.1 New Callback 1	
Callback Info	Value
User	[REDACTED] Testing
Host	[REDACTED]
PID	5664
IPs	[REDACTED] 236
External IP	
Process Name	apollo
Description	Created by mythic_admin at 2024-06-16 22:33:03 Z
OS/Arch	Windows 10 Enterprise 2009 6.2.9200.0/x64
Domain	[REDACTED]
Integrity Level	MEDIUM
Initial Checkin	2024-06-16 23:05:09 Z
Last Checkin	2024-06-16 23:08:28 Z
C2 Channels	http

Figure 3-28
Mythic Report Output

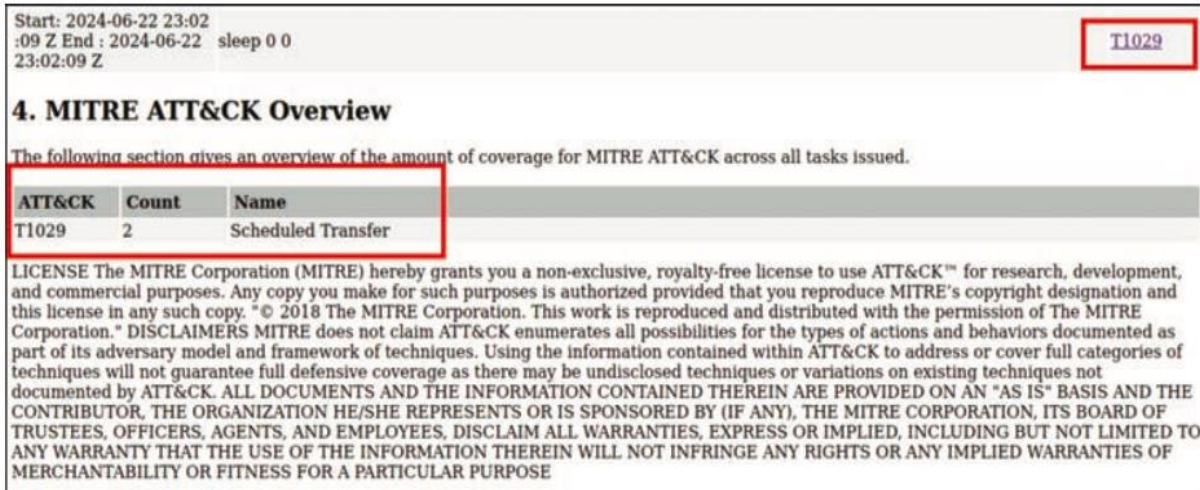


Figure 3-29
Mythic Report MITRE ATT&CK Mappings