

# Hiding in Plain Sight: Obscuring C2s by Abusing CDN Services

 [teamt5.org/en/posts/hiding-in-plain-sight-obscuring-c2s-by-abusing-cdn-services](https://teamt5.org/en/posts/hiding-in-plain-sight-obscuring-c2s-by-abusing-cdn-services)

Cyber Threat Intelligence



Threat Intelligence

[RSS](#)

5.10.2022 Cyber Threat Intelligence

Share:

Cobalt Strike is a well-established red-teaming toolkit that was specifically designed to create feature-rich backdoors in the matter of seconds. However, ever since its inception, the tool has been heavily abused by threat actors all over the world (e.g., Polaris/MustangPanda, APT32/OceanLotus, cybercriminals, etc.) The popularity of Cobalt Strike is primarily due to how powerful its built-in payload generators are, its flexibility in deploying the Beacon to the target machine(s) - and most notably - its ability to disguise the payload traffic in the target network.

Over the last year or so, more and more beacons relying on third-party services as their proxy are being found in the wild. Since the use of these third-party CDNs can allow anonymized C2 and enhance its detection evasion, numerous APTs have also begun using these services. In the following sections, we will go through some of the techniques used by these threat actors to see how one could easily set up these misused configurations.

## Serverless Functions

Other techniques that threat groups have been employing also include the use of third-party services such as Cloudflare Workers. With the use of serverless services, it is possible to setup what is essentially a free reverse proxy to a Cobalt Strike endpoint. When set up properly, these services can provide an additional layer of protection, as looking up the domain will only yield the service's IP instead of the origin server.

The screenshot displays the Cloudflare Workers interface. On the left is a navigation sidebar with options: Websites, Buy Domains (New), Analytics, Pages, Workers (selected), Stream, Images (New), Security Center (Beta), Zero Trust, Bulk Redirects (Beta), Notifications, and Manage Account. The main content area shows a 'production' environment with a table of routes. One route is listed: 'route.moffice365.workers.dev' with a 'Bundled' usage model and a last modified date of 'April 28, 2022'. Below the table are tabs for Resources, Triggers, Logs, Deployments, and Settings. The 'Resources' tab is active, showing a 'Worker' section with a 'Summary' of metrics: 'Metrics aggregated across all custom and workers.dev routes invoking this Worker. Apr 27th - Apr 28th'. A box highlights 'Requests 136'. To the right, there is a 'Requests' section with a sub-header 'Historical requests for route grouped by invocation status.'




Since these services are often set up with sub-domains that the threat actor could freely set up without purchasing a real domain, they could also be set to a somewhat convincing fake domain. In this example, the C2 was set to `route.moffice365.workers.dev` to make the traffic look somewhat related to Microsoft services.

Create a listener.

Name:

Payload:

### Payload Options

HTTP Hosts:    

HTTP Host (Stager):

Profile:

HTTP Port (C2):

HTTP Port (Bind):

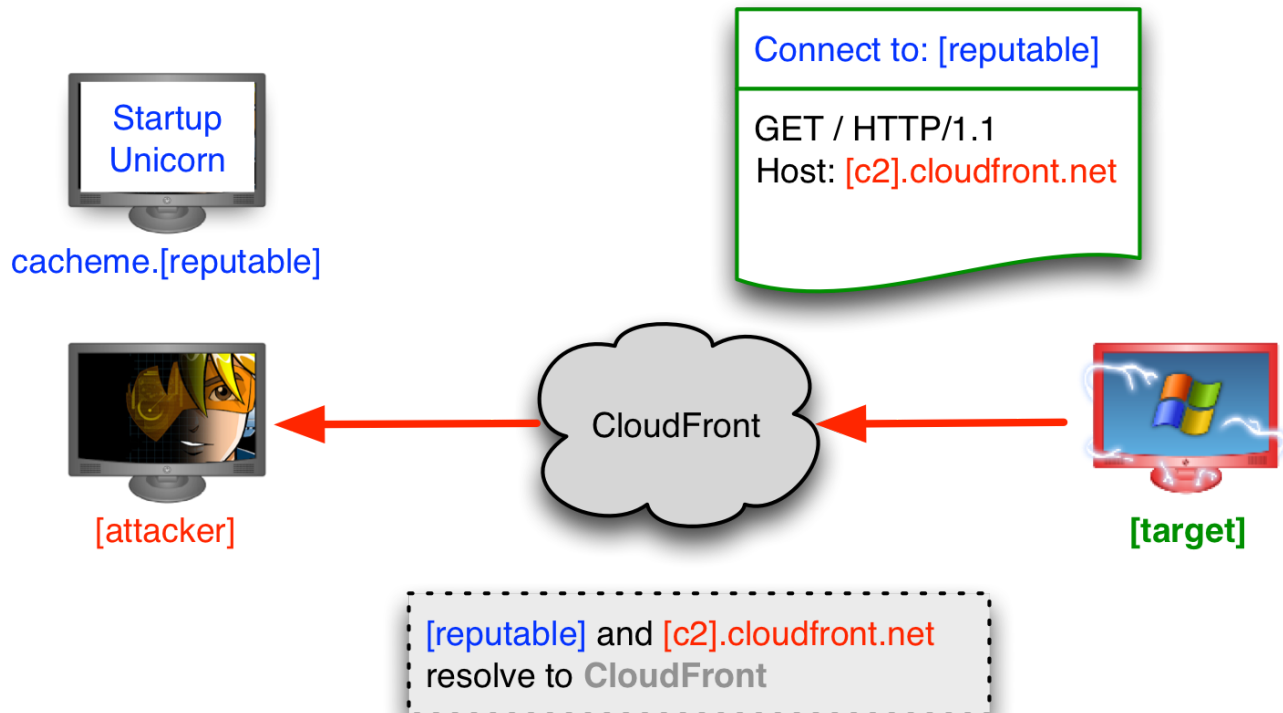
HTTP Host Header:

HTTP Proxy:  

The nature of these serverless functions makes it extremely difficult to attribute the origins of or identify the threat actors. Since the serverless code is only stored on Cloudflare, and its execution is done on Cloudflare's servers, there is not a feasible way to identify the C2 infrastructures.

### Domain Fronting via CDN

Domain fronting is a technique that attempts to disguise the traffic by smuggling data to a well-known service or domain. In other words, similar to the previous technique, domain fronting cannot be feasibly blocked without disrupting services and or examining the inner request. HelpSystems, the company behind Cobalt Strike, had actually done a write-up of Cobalt Strike domain fronting using CloudFront in 2017[1].



Essentially, certain websites employ CDNs like CloudFront to deliver assets on their websites. Under normal circumstances, the `Host` header determines where the request should go - and this header is assigned by the website. However, we can also abuse this "feature" and make our beacon visit a certain domain with a specified Host header that goes to our CDN instead of the standard CDN that the website uses. With HTTPS, this request would be extremely difficult or impossible for the IT admin to block, as the request itself is encrypted, and on the surface, it would look like the user is simply visiting a reputable site.

## Domain Fronting with a Non-existent Domain

The following technique, however, does something slightly different. Some time ago last year, we detected a rather peculiar looking Cobalt Strike payload where its C2 address was set to `pypi.python.org`. Obviously the actor couldn't have possibly obtained the rights to the domain owned by the Python Foundation, and the chance of them pwning the service is also little to none.

```
BeaconType - HTTPS
Port - 443
SleepTime - 1000
MaxGetSize - [REDACTED]
Jitter - 10
MaxDNS - Not Found
PublicKey_MD5 - [REDACTED]
C2Server - pypi.python.org,/latest/pip-check
```

So what gives? It turns out the secret lies within the service that operates the sub-domain and the HTTP header specified in the beacon. The Python Foundation uses Fastly for most of its services. Fastly uses the `Host` header specified within the request to decide which service to redirect the request to internally. For example, if I were to create a service on Fastly named `dl-python.org`, and the `Host` header is set to `dl-python.org` in the beacon, Fastly would then forward the request to the service named as such.

Note the difference between this methodology against the previous domain fronting technique - the `Host` header may give a false impression that the C2 is being pointed towards `dl-python.org`. In reality, it is being resolved as `dl-python.org.global.prod.fastly.net` when going through Fastly's services. The service then internally forwards the request to `my-c2domain.com` as configured by the threat actor. This trick may lure threat investigators into investigating a completely different domain that may or may not be relevant to the incident.

## Conclusion

In this post, we have covered three ways that threat actors can abuse Cobalt Strike's flexibility in payload communication: misuse of serverless functions and various abuses of CDN domain fronting. The resilience of Cobalt Strike is something that is beloved (or despised) by many because of this very reason. It is clear that the red-teaming toolkit is not going anywhere anytime soon as more and more creative strategies are discovered and employed.

## References

---

[1] Mudge, R. (2017, February 6). High-reputation Redirectors and Domain Fronting—Cobalt Strike Research and Development. → [link](#)

\*Image courtesy of [Pexels](#)

Share:

