Cyclops: a likely replacement for BellaCiao

By Cyber Threat Research Team

Published: 2024-08-14 · Archived: 2026-04-06 01:27:15 UTC

Inside The Lab

Published on 14 August, 2024 19min

Identifier: TRR240801.

Summary

This report introduces Cyclops, a newly discovered and previously undocumented malware platform written in Go which dates back to December 2023, and that we believe has been deployed against targets in the Middle-East in 2024. Cyclops allows operators to execute arbitrary commands on the target's file system, as well as pivot inside

the infected network. Notably, Cyclops is controlled through a HTTP REST API which is exposed to operators within an SSH tunnel.

Based on our research, we assess that Cyclops was likely developed as a sucessor to the BellaCiao[1] malware. We attribute this new platform to “Charming Kitten” (also known as APT 35) due to significant overlaps in TTPs and targeting. Charming Kitten recently made the headlines due to accusations of attempting to interfere with the US elections[2].

As far as we know, there are only a limited number of samples of this family, with evidence suggesting its development was completed in December 2023. This recent emergence and limited prevalence indicate that Cyclops may still be in its early stages, and hope that this report allows the community to further detect, analyze and potentially curb its spread.

Background

While hunting for in-the-wild malicious implants in late July 2024, we identified a poorly detected binary (SHA-256 `fafa68e626f1b789261c4dd7fae692756cf71881c7273260af26ca051a094a69`, see Fig. 1) that requested the resolution of a hostname we could associate with the BellaCiao implant[1].

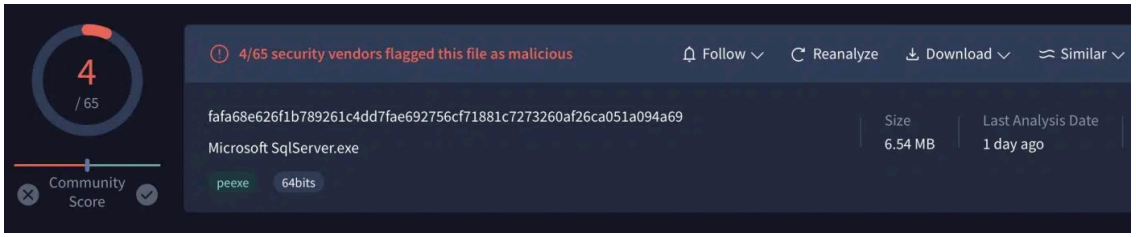

Figure 1 – Poor detection of the identified binary on a public online multiscanner service, as of July 30, 2024

Further analysing the suspicious binary (see the “Cyclops” title below) and pivoting from related data unveiled more ties with BellaCiao and Charming Kitten[4]-associated activities.

Infection chain

We miss information about the context under which the identified binary has been deployed.

However, from previous reporting[1], original name of the identified binary (`Microsoft SqlServer.exe`) as well as our own experience investigating BellaCiao incidents, we believe with medium confidence that Cyclops could be deployed on servers following an exploitation of vulnerable services.

In particular, we previously analysed infection cases where BellaCiao samples were deployed from ASP .NET webshells, following the exploitation of Exchange Web servers vulnerabilities.

Additionally, because we believe with medium to high confidence that Cyclops is a replacement for BellaCiao, we also believe with medium confidence that Cyclops could be deployed from previous BellaCiao infections.

Cyclops

Filename	`Microsoft SqlServer.exe`
Compiler	Go 1.22.4
Hash (SHA256)	`fafa68e626f1b789261c4dd7fae692756cf71881c7273260af26ca051a094a69`

Cyclops, as seemingly named by its authors, is a complex malware platform written in Go language. It aims at reverse-tunelling a REST API to its command and control (C2) server for the purposes of controlling targeted machines. It uses the go-svc library to be able to run as a service if needed, which could be its preferred persistence method on Windows.

It allows operators to run arbitrary commands, manipulate the target's filesystem, and use the infected machine to pivot into the network.

Libraries dependencies and associated versions being embedded in this Go binary sample (at compilation time), one can determine that the newest external dependency for the sample (google/uuid v1.5.0) has been released in December 12, 2023. This indicate that this exact sample development ended in December 2023 at the earliest. Additionally, Go compiler version for this sample is set to 1.22.4, which according to the official Go repository, has been released in June 4, 2024. This date information indicates that the specific sample we analysed could not have been deployed before June 2024.

SSH tunneling

On startup, the program loads an embedded configuration (encrypted with AES-128 CBC), which contains information about its C2 server and initial connection:

```
{
   "StartDelay": 5000,
   "SonarConfigs": {
       "Cycle": 1800000,
       "HostName": "lialb.autoupdate[.]uk",
       "HostNameFormat": "%s.%s",
       "ExpectedAddress": [REDACTED]
   },
   "BeamConfigs": {
       "BeamAgent": "SSH-2.2-OpenSSH_for_Windows_8.1",
       "UserName": [REDACTED],
       "Password": [REDACTED],
       "Host": "88.80.145[.]126:443",
       "LocalAddress": "127.0.0.1:9090",
       "RemoteAddress": "127.0.30.3:9090",
       "Retry": 10
   }
}
```

As a possible anti-analysis measure, Cyclops starts by resolving a random “validation” hostname within the subdomain of the `HostName` configuration field (i.e., `iuxyf.lialb.autoupdate[.]uk`). If the hostname doesn’t exist, or doesn’t resolve to an IP address matched by the regular expression contained in `ExpectedAddress` field, the malware enters a sleep cycle and tries again later. Otherwise, the malware starts its built-in HTTPS server, and forwards the corresponding port to the C2 server via the SSH connection (parameter `LocalAddress` of the configuration).

Local HTTPS server

As part of its startup phase, Cyclops loads a second AES-128-encrypted configuration blob which controls the behavior of its internal HTTPS server:

```
{
   "BindAddress": "127.0.0.1:55561",
   "TLS":
   {
       "CertPEMBlock": [base64-encoded certificate],
       "KeyPEMBlock": [base64-encoded key]
   },
   "Users": {
       "[REDACTED username]": "[REDACTED SHA-256 hash]"
   }
}
```

The developers use a modified version of the gorilla/mux package to handle incoming HTTPS requests coming via Go’s build-in `net/http` web server. The server starts listening on the `BindAddress` using the TLS certificate and key provided in the configuration.

One of the main changes made to `gorilla/mux` was adding support for basic HTTP authentication. This feature was manually implemented so that only requests containing the credentials in the `Users` section of the configuration are accepted. It is worth noting that their implementation is not RFC-compliant as the credentials are not base64-encoded. The header expected by Cyclops looks like:

```
Authorization: Basic username:cleartextpassword
```

…instead of:

```
Authorization: Basic dXNlcm5hbWU6Y2xlYXJ0ZXh0cGFzc3dvcmQ=
```

The (self-signed) TLS certificate bundled with this implant has the following characteristics, confirming a likely end of development in December 2023:

```
 Serial Number: 9e:22:f8:bb:63:88:5e:d0:71:8f:f6:61:7c:17:ec:e5
Signature Algorithm: sha256WithRSAEncryption
```

```
Issuer: O=mkcert development CA, OU=deathpact@fedora (DeathPact), CN=mkcert deathpact@fedora (DeathPact)
...
Validity
    Not Before: Dec 20 09:22:46 2023 GMT
    Not After : Mar 20 08:22:46 2026 GMT
```

REST API control channel

Finally, as the HTTPS service becomes reachable through the SSH tunnel, it may start receiving orders from the operators on a single endpoint: `/api/v3/update`. Only POST requests are accepted and the payload must be placed in a multipart file named `resume`. Cyclops relies on a custom protocol which can be broken down like so:

Size (bytes)	Name (ours)	Description
36		Unused
4	`command_description_size`	Size of the next field (network byte order)
`command_description_size`	`command_description`	The requested command passed as a JSON object
Until the end of the packet	`command_arguments`	The parameters to give to the command, also as a JSON object

The `command_description` field is a simple object of the following format:

```
 {
    "type": "test"|"review"|"storage"|"upload"|"download"|"pf"|"server",
    "syncresult": true|false
}
```

When Cyclops receives such requests, it uses `type` as a key in a map of all available command handlers. For instance, the type `download` will cause an instance of `APIDownload` to be created, and its method `Update` to be invoked receiving `command_arguments` as an argument. The second parameter, `syncresult`, controls whether the client expects a synchronous or asynchronous response ; for more details about this feature, see the description of the `storage` command below.

A high-level description of the available commands follows:

Type	Description
`test`	Does nothing, and in fact crashes the worker because the developers failed to initialize the object containing the response, leading to a nil dereference.
`review`	Arbitrary command execution with Go's `os.exec` package.
`storage`	Controls the in-memory store containing the results of asynchronous commands.
`upload`	Sends a file to the targeted machine.
`download`	Retrieves a file from the targeted machine.
`pf`	Sets up port forwarding via SSH tunnels. Should be run asynchronously or hangs the worker.
`server`	Waits 5 seconds and shuts the HTTPS server down.

After executing a command, Cyclops returns two objects in JSON format. The first one represents the API result (whether the command failed, start and end time, response size, etc.) as well as a command-specific result object containing the actual response. An example API result would be:


```
 {
    "ISerialize":null,
    "uuid":"650ee54f-558e-11ef-8ddb-000c29fbee52",
    "isDeferred":true,
    "status":0,
    "fileSize":148,
    "currentSize":148,
    "hasError":false,
    "error":"%!s(u003cnilu003e)",
    "type":"review",
    "startDate":"2024-08-08T13:59:17.3233476Z",
    "doneDate":"2024-08-08T13:59:17.4461401Z",
    "done":true
}
```


Most fields are either self-explanatory or irrelevant. The UUID is not a target identifier but a GUID generated for each request using the `google/uuid` package, used to retrieve asynchronous command results. A sample Python script which can be used to interact with a running instance of Cyclops is provided in Appendix.

Review command

The review command, providing command execution, expects the following parameters:

Name	Description
`dir`	The directory containing the program to execute

Name	Description
`file`	The name of the executable to run
`timeout`	Maximum time to wait for the program's termination
`args`	A list of strings to be used as arguments for the program
`envs`	A dictionary of environment variables to provide to the progam

Results are returned in the following format:

Name	Description
`code`	Exit code of the program
`out`	A list containing the output of the program executed. Each entry represents a single line and is a dictionary with two keys: `time` (timestamp at which the line was captured) and `content` (the corresponding data). See example below.
`err`	An error that occurred, or `null`
`pid`	The PID of the program that was executed
`isDone`	Whether the program is finished running

To clarify the structure of the `out` value, sample output for the execution of `cmd.exe` is reproduced here:

```
{
   // [...]
   "out": [
       {
           "time":"2024-08-08T13:59:17.4345436Z",
           "content":"Microsoft Windows [Version 10.0.19044.2251]"
       },
       {
           "time":"2024-08-08T13:59:17.4345436Z",
           "content":"(c) Microsoft Corporation. All rights reserved."
       }
       // ...
   ]
   // [...]
}
```

Download command

A command used to send a file from the infected machine to the C2 server. This command only requires a simple argument in the form of:

```
 {
    "path": "[Local file to send to the C2]"
}
```

The response contains the two usual JSON objects (API response and command response) which do not contain any information of particular interest. The raw bytes of the requested file is placed right after them:

Size	Name (ours)	Description
4	`api_result_size`	Size of the API response in bytes, network order
`api_result_size`	`api_response`	JSON object representing the status of the API request (see above)
4	`command_result_size`	Size of the command response in bytes, network order. This field is omitted in all other commands.
`command_result_size`	`command_result`	JSON object representing the command result: whether it was successful, the path of the requested file, etc.
Until the end of the file	`data`	Raw bytes of the requested file. This field is omitted in all other commands.

Upload command

Writes an arbitrary file on the infected machines's filesystem. The expected command argument is:

```
 {
    "path": "[Path to where the file should be written]"
}
```

Since this command needs to receive arbitrary data to write into the file, it expects a slightly different command description format that allows it to find the offset of the file data:

Size (bytes)	Name (ours)	Description
36		Unused
4	`command_description_size`	Size of the next field (network byte order)

Size (bytes)	Name (ours)	Description
`command_description_size`	`command_description`	The requested command passed as a JSON object
4	`optional_command_args_size`	Size of the next field (network byte order).
`optional_command_args_size`	The parameters to give to the command, also as a JSON object	
Until the end of the packet	`payload`	Raw bytes of the file to write.

The responses produced by this command does not contain any particularly noteworthy information.

Storage command

Commands submitted to Cyclops can either be synchronous (the client waits until a result is returned), or asynchronous (the clients receives a GUID corresponding to the request and polls until a result becomes available). Asynchronous jobs get tracked via an in-memory registry which keeps a record of the ongoing jobs based on their UUID and their status. A worker checks every 5 seconds if a finished job is more than 5 minutes old, and if so, deletes it from the registry.

The command allows operators to check the contents of the registry and obtain results for finished commands. Its parameters are:

```
{
   "type": "[Subcommand type]",
   "args": ""
}
```

The `list` subcommand doesn't require any arguments and simply returns the contents of the registry in a reponse object such as:

```
{
   "ISerialize":null,
   "items": [
       {
           "ISerialize":null,
           "uuid":"3966c1ea-568c-11ef-b418-000c29fbee52",
           "isDeferred":true,
           "status":0,
           "fileSize":149,
           "currentSize":149,
```

```
            "hasError":false,
            "error":"%!s(u003cnilu003e)",
            "type":"review",
            "startDate":"2024-08-09T20:16:15.3046826Z",
            "doneDate":"2024-08-09T20:16:15.3933832Z",
            "done":true
        },
        // ...
    ]
}
```

The `update` subcommand returns the result associated with a given UUID; `args` should be a JSON object like `{"id": "UUID"}`. If the corresponding UUID is found in the store, Cyclops returns the result of that asynchronous command. Finally, the `kill` subcommand terminates a worker that still hasn't produced a result.

Pf (port-forwarding) command

A command to forward TCP ports via SSH tunnels, matching the semantics of the `-R` or `-L` options of SSH.

```
 {
    "serverAddr": "[The server to connect to, in the form IP:port]",
    "direction": "L"|"R", // For local or remote forwarding.
    "userName": "[The username used to log into the server]",
    "password": "[The password used to log into the server]",
    "localAddr": "[The address to use on the infected machine]",
    "remoteAddr": "[The address to use on the C2]"
}
```

For instance, the arguments:

```
 {
    // [...]
    "direction": "L",
    "localAddr": "127.0.0.1:9091",
    "remoteAddr": "127.0.0.1:9092"
}
```

…would allow the infected machine to connect to any service listening on the remote host's port `9092` by connecting to `127.0.0.1:9091`.

Server command

This command doesn't take any arguments (despite traces in the code of receiving `type` and `args` like `storage`) and simply shuts the HTTPS server down after 5 seconds. We were not able to identify any way to restart it afterwards, the command and control channel becoming unavailable at the same time as the web server.

We speculate this is a broader reconfiguration command for the webserver that is still under implementation.

Infrastructure

The identified Cyclops sample caught our attention because it requires the resolution of a hostname (`<random letters>.lialb.autoupdate[.]uk`) from the `autoupdate[.]uk` domain, which in turns uses the glued `88.80.145[.]126` IP as authoritary name server (NS) since mid-May 2023. This last IP was also associated with the NS of a publicly documented[1] BellaCiao C2 domain (`mail-updateservice[.]info`).

We particularly looked at NS records and associated IP addresses, given both Cyclops and BellaCiao rely on DNS resolutions as a validation flag to continue running. This implies operators must control DNS resolutions for associated domains – which can be easily done with operator-owned NS servers.

Domain	Registration	NS
`autoupdate[.]uk` (Cyclops validator)	NameSilo (2023-04-26)	`ns1` and `ns2.autoupdate[.]uk`, resolves to `88.80.145[.]126` (ASN 44901 – Belcloud) since 2023-05-19.
`mail-updateservice[.]info` (Publicly known BellaCiao C2)	NameSilo (2022-12-24)	`ns1` and `ns2.mail-updateservice[.]info`, resolved to `88.80.145[.]126` (ASN 44901 – Belcloud) from 2023-08-28 to 2023-12-23.

Pivoting from `88.80.145[.]126`, we identified additional and possibly linked IP addresses and domains.

Domain/IP	Details
`88.80.145[.]93`	ASN 44901 – Belcloud. Same (self-signed) TLS certificate as `88.80.145[.]126` for the RDP service.
`88.80.145[.]122`	ASN 44901 – Belcloud. Same services on the same unusual ports as `88.80.145[.]126`. Resolution for NS of known BellaCiao validator domains (such as `maill-support[.]com`, `twittsupport[.]com` and `msn-service[.]co`), starting 2022-11-25.
`88.80.145[.]137`	ASN 44901 – Belcloud. Resolution for `ns1` and `ns2.freeheadlines[.]top`, between 2024-02-01 and 2024-03-14.
`88.80.145[.]132`	ASN 44901 – Belcloud. Resolution for `ns1` and `ns2.servicechecker[.]top`, between 2023-06-07 and 2024-03-06.
`servicechecker[.]top`	Registered at NameSilo (2023-06-07), also using IP `88.80.145[.]126` as NS (starting 2024-03-09).
`servicesupdate[.]info`	Registered at NameSilo (2023-04-26), also using IP `88.80.145[.]126` as NS (from 2023-05-15 to 2024-04-18). Domain name looks like a known

Domain/IP	Details
	BellaCiao validator (`mail-updateservice[.]info`).
`mail-update[.]info`	Registered at NameSilo (2022-12-24), also using IP `88.80.145[.]126` as NS (from 2023-08-28 to 2023-12-23). Domain name looks like known BellaCiao validator (`mailupdate[.]info`).
`freeheadlines[.]top`	Registered at NameSilo (2024-02-01), using IP `88.80.145[.]122` as NS (from 2024-03-16 to 2024-07-26).
`servicepackupdate[.]info`	Registered at NameSilo (2023-05-22), using IP `88.80.145[.]122` as NS (from 2023-05-23 to 2024-05-20).
`systemupdate[.]info`	Registered at NameSilo (2023-05-22), using IP `88.80.145[.]122` as NS (from 2023-05-22 to 2024-04-25).

Targets

As we initially identified a single sample of the Cyclops implant from an online multiscanner service, we have very limited information about Cyclops targets.

We noticed that publicly available and documented BellaCiao samples or associated C2 infrastructure always contained information about associated targets (such as names, domain names or IP addresses).

Analyzing Cyclops and associated infrastructure while considering the operators and developers may have followed the same practice, we determined with medium to high confidence that Cyclops targeted a non-profit organization which supports innovation and entrepreneurship in Lebanon, as well as a telecommunication company in Afghanistan.

Attribution

We first decided to focus on the Cyclops sample because the associated domain could be linked with known BellaCiao infrastructure, as we demonstrated in a previous title.

While analyzing Cyclops, we noticed other similarities with publicly documented BellaCiao samples and related activities:

- Both Cyclops and BellaCiao use a hostname resolution result to control implants execution flow, and both implants generate hostnames containing random caracters and a target-specific explicit identifier;
- Both implants aim at starting a malicious HTTP service for the operators to send commands (dropped Webshell or PowerShell payload for BellaCiao, embedded service for Cyclops). The HTTP services are additionally exposed on unusual high port numbers, and executing commands through such services is password-protected;
- Implant filenames (poorly) impersonate server or update services (such as `Exchange Agent Diagnostic Services.exe` for BellaCiao, `Microsoft SqlServer.exe` for Cyclops);

- Both implants' operators rely on SSH tunnels (embedded feature for Cyclops, additional tools such as Plink required for BellaCiao) to access created HTTP services on targeted computers;
- Last but not least, both implants' operators use the same usernames and/or passwords as part of their HTTP or SSH services access control.

Additionaly, we determined that the development for the Cyclops sample that we identified likely ended in late 2023, while to the best of our knowledge, BellaCiao usage seemingly stopped between late June 2023 and December 2023. As a result, we believe with medium to high confidence that Cyclops is a replacement for BellaCiao, and is likely developed and/or operated by the same threat actor.

BellaCiao has been publicly associated with Charming Kitten[1,5]. We notice that both BellaCiao and Cyclops operators indeed rely on practices which are leveraged by other threat actors in the "Kitten" family, and that Cyclops targets we know of could match Islamic Revolutionary Guard Corps' (IRGC) interests: extending Iran's influence across the Middle East (and beyond), including in Lebanon, where Hezbollah, Iran's close ally, operates from.

However, related practices (such as network communication tunneling, custom implants, and HTTP services usage) are quite common for many threat actors. While these align with known tactics of the "Kitten" family of threat actors, they are not unique to this group. Furthermore, the two targets we identified are not enough to establish any clear victimology focus or definitively attribute these actions to the IRGC. More data and evidence would be needed to draw firmer conclusions about the threat actor's identity and motivations.

Conclusion

This research shows an increase in proficiency for Charming Kitten, with the discovery of a new and well-designed malware platform. Following publications on BellaCiao, we believe the threat actor took action and started developing new malware to replace their burnt tools – thus confirming the impact of threat intelligence on adversary operations.

The choice of Go for the Cyclops malware has a few implications. Firstly, it confirms the popularity of this language among malware developers. Secondly, the initially low number of detections for this sample indicates that Go programs may still represent a challenge for security solutions. And finally, it is possible that MacOS and Linux variants of Cyclops were also created from the same codebase and that we have yet to find them.

Overall, based on our investigations, the prevalence of Cyclops appears to be highly limited at present. By releasing this research, we aim to enable the broader community to identify additional samples and potentially discover more about the scope of Charming Kitten's recent operations.

Appendix

Indicators of compromise (IOCs)

Associated IOCs are also available on our GitHub repository.

Hashes (SHA-256)

```
fafa68e626f1b789261c4dd7fae692756cf71881c7273260af26ca051a094a69|Cyclops
```

Domains

```
autoupdate[.]uk|Cyclops validator
```

IP Addresses

```
88.80.145.126|Cyclops SSH C2 and validator NS
```

URLs

```
hxxps://127.0.0.1:55561/api/v3/update|Cyclops REST API endpoint
```

Possibly associated domains

```
 servicechecker[.]top|Possible BellaCiao or Cyclops validator
servicesupdate[.]info|Possible former BellaCiao or Cyclops validator
mail-update[.]info|Possible former BellaCiao or Cyclops validator
freeheadlines[.]top|Possible former BellaCiao or Cyclops validator
servicepackupdate[.]info|Possible former BellaCiao or Cyclops validator
systemupdate[.]info|Possible former BellaCiao or Cyclops validator
```

Possibly associated IP Addresses

```
 88.80.145[.]93|Possible BellaCiao or Cyclops infrastructure
88.80.145[.]122|Possible BellaCiao or Cyclops infrastructure
88.80.145[.]137|Possible former BellaCiao or Cyclops infrastructure
88.80.145[.]132|Possible former BellaCiao or Cyclops infrastructure
```

Yara rules

```
 rule charmingkitten_cyclops
{
    meta:
        description = "Detects Cyclops Golang Malware"
        references = "TRR240801"
        hash = "fafa68e626f1b789261c4dd7fae692756cf71881c7273260af26ca051a094a69"
        date = "2024-08-05"
        author = "HarfangLab"
```

```
        context = "file"
    strings:
        $go = " Go build ID: \"" ascii
        $a1 = "dep\tback-service\t(devel)" ascii fullword
        $a2 = "/brain-loader-enc.gox00" ascii
        $a3 = "back-service/go-mux/api" ascii
        $a4 = "/JD-M42KItJncJfqb38qh/" ascii
    condition:
        filesize > 2MB and filesize < 20MB
        and (uint16(0) == 0x5A4D)
        and $go
        and (2 of ($a*))
}
```

Suricata rules

```
 alert tcp $EXTERNAL_NET any -> $HOME_NET [80,443,1024:] (msg:"Invalid RFC7617 Basic Auth in HTTP POST"; flow:es

alert tcp $EXTERNAL_NET any -> $HOME_NET [80,443,1024:] (msg:"Cyclops HTTP API Request"; flow:established,to_ser
```

Python script to interact with Cyclops

The associated Python code snippet is available on our GitHub repository.

---

1. https://www.bitdefender.com/blog/businessinsights/unpacking-bellaciao-a-closer-look-at-irans-latest-malware/ ↩ ↩ ↩ ↩ ↩
2. https://cdn-dynmedia-1.microsoft.com/is/content/microsoftcorp/microsoft/final/en-us/microsoft-brand/documents/5bc57431-a7a9-49ad-944d-b93b7d35d0fc.pdf ↩
3. https://www.microsoft.com/en-us/security/blog/2024/01/17/new-ttps-observed-in-mint-sandstorm-campaign-targeting-high-profile-individuals-at-universities-and-research-orgs/ ↩
4. https://securelist.com/apt-trends-report-q3-2023/110752/ ↩
