Tor: Finding the Hidden Shallots

João Marques

University of Amsterdam

joao.marques@os3.nl

July 5, 2018

Overview

Why this project?

Hidden Services importance (for the **service provider**):

- Anonymity
- Freedom

Why this project?

Hidden Services importance (for the **service provider**):

- Anonymity
- Freedom

Consequences of above values:

- legitimate - Uncensored news website/blog - important to secure
- illegitimate - C&C Servers / Uncontrolled markets - Extract intel / monitor

Previous Research

In 2013 a paper by Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann was published, titled: **Trawling for tor hidden services: Detection, measurement, deanonymization**

They were very successful and gave recommendations to stop the acquisition of Hidden services, and targeted attacks

Previous Research

In 2013 a paper by Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann was published, titled: **Trawling for tor hidden services: Detection, measurement, deanonymization**

They were very successful and gave recommendations to stop the acquisition of Hidden services, and targeted attacks

Despite the work done:

- No extraction method
- No tools
- Requires verification for changes

Research Question

How feasible is the acquisition of hidden service links (onion links)?

Research Question

How feasible is the acquisition of hidden service links (onion links)?

- What is the state of the current specification?
- How are protection mechanisms used/applied?
- What protocols are still used in the wild?
- Are these protocols safe?
- How can we extract from unsafe ones?

What is the The Onion Rounting (Tor) Network?

Tor Network

The tor network is an **Overlay Network** that aims to provide the **user** with:

- Privacy
- Anonymity

Tor Network

The tor network is an **Overlay Network** that aims to provide the **user** with:

- Privacy
- Anonymity

Tor: How does it work?

For the Tor network to work it makes use of 3 types of relays/nodes:

Tor: How does it work?

For the Tor network to work it makes use of 3 types of relays/nodes:

- **Guard Node** - First node of the circuit created by the client and where traffic enters the Tor Network
- **Middle Node**
- **Exit Node**

Tor: How does it work?

For the Tor network to work it makes use of 3 types of relays/nodes:

- **Guard Node**
- **Middle Node** - Second node of the circuit, it relays the traffic between the guard node and the exit node
- **Exit Node**

Tor: How does it work?

For the Tor network to work it makes use of 3 types of relays/nodes:

- **Guard Node**
- **Middle Node**
- **Exit Node** - Third and last Node of the circuit, where the traffic gets unencrypted and sent to the destination

Tor: How does it work?

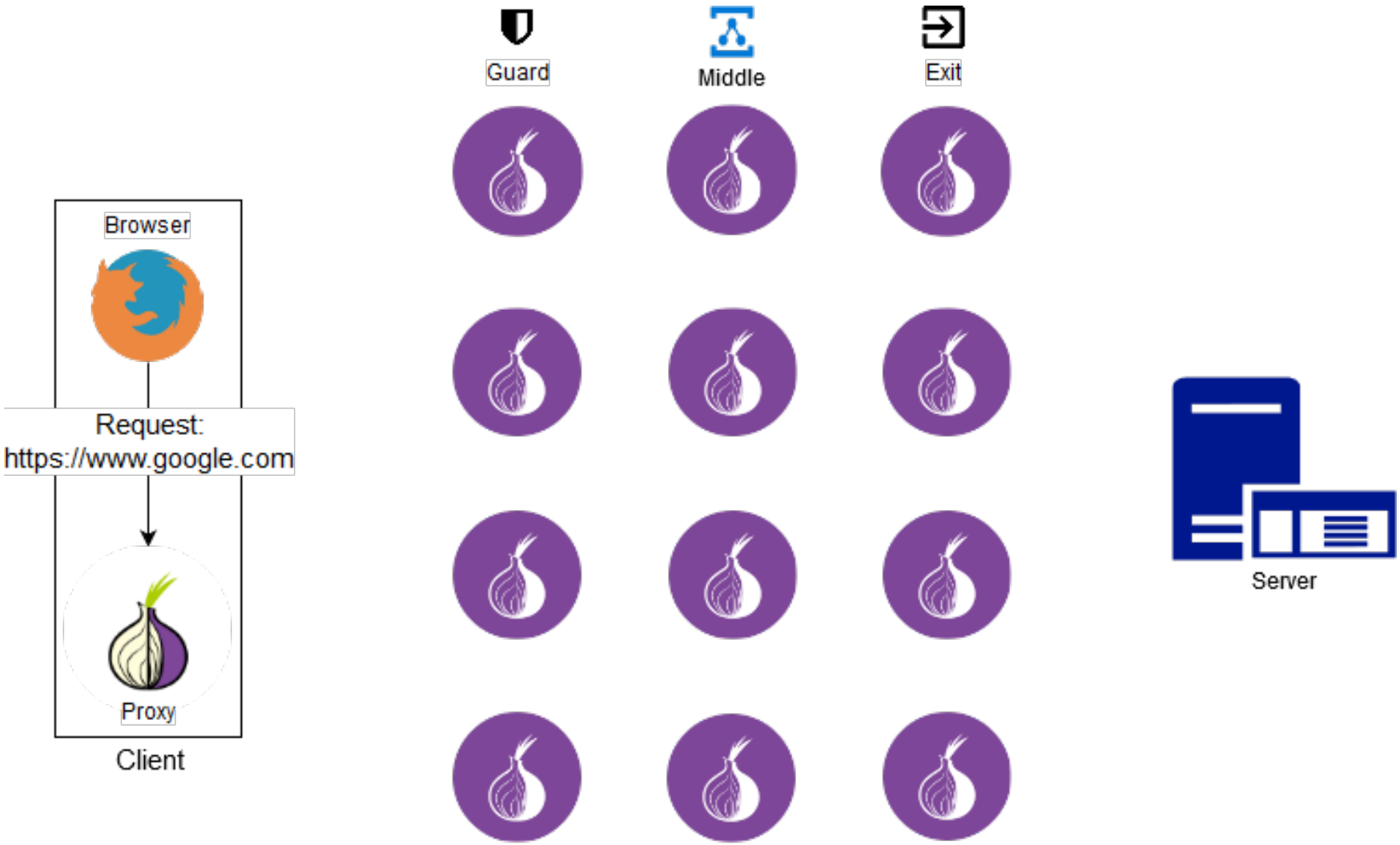

Figure: Tor browser requests page to proxy

Tor: How does it work?

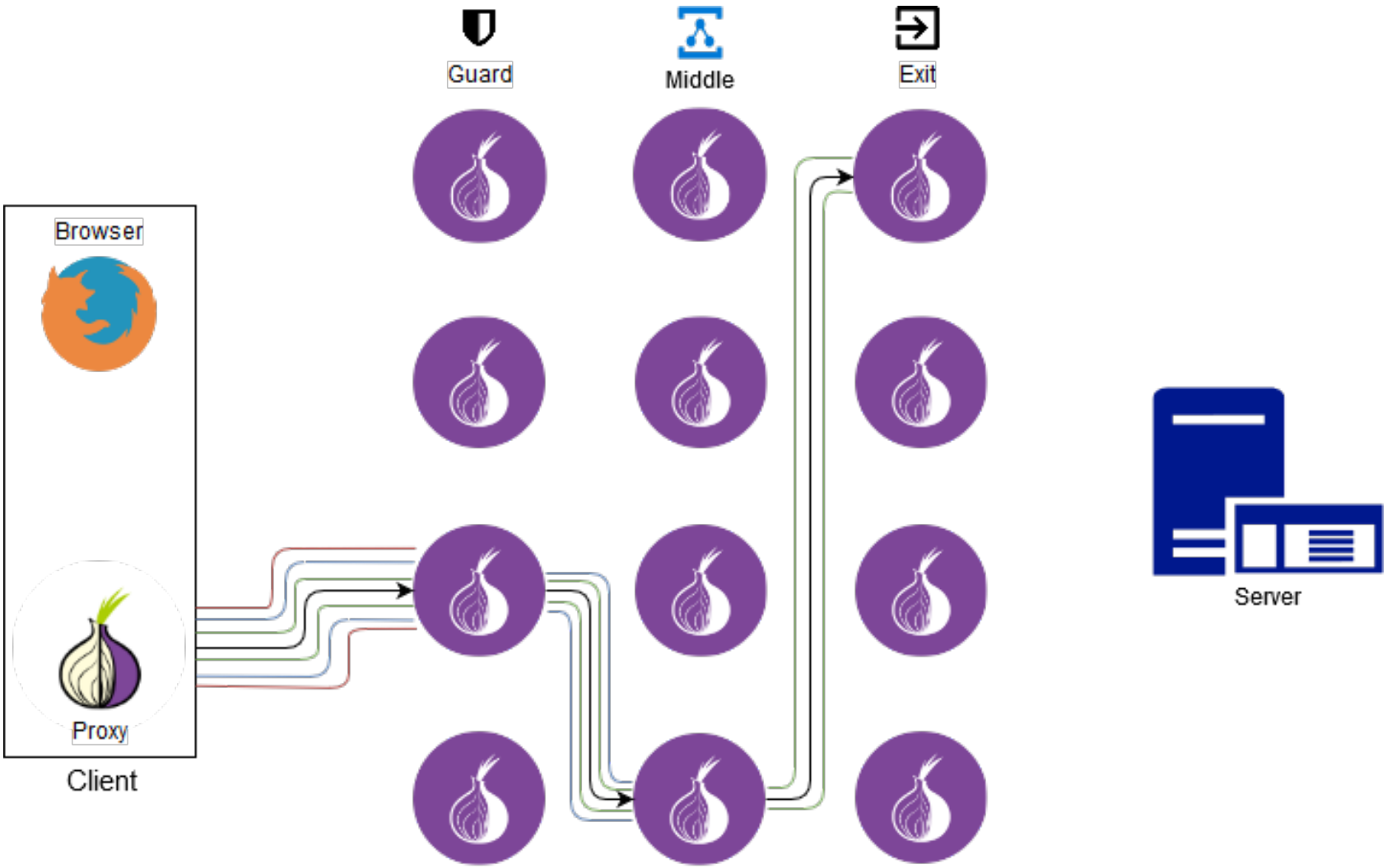

Figure: Tor proxy negotiates encryption layer with each node

Tor: How does it work?

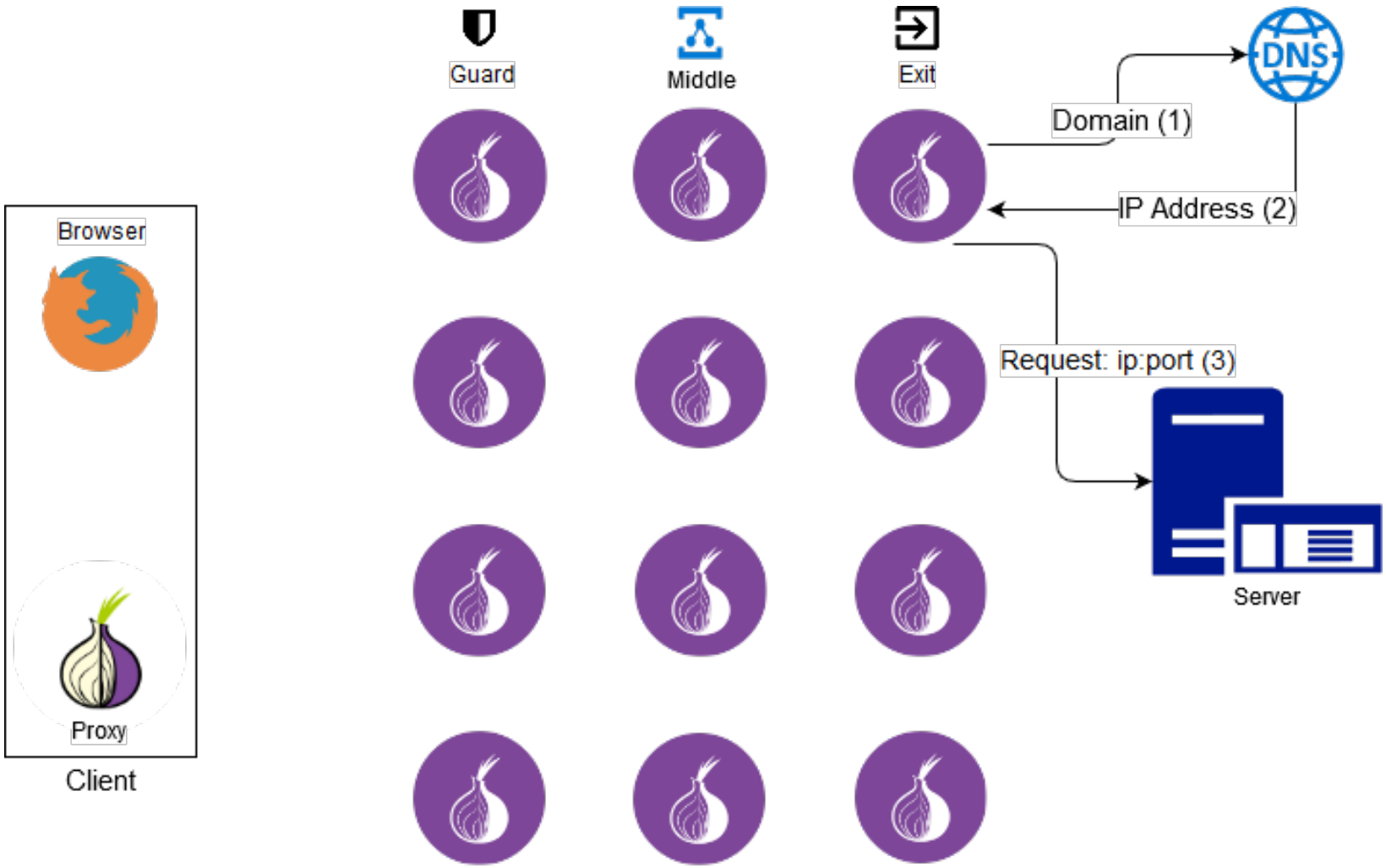

Figure: Exit node communicates on the user's behalf

Tor: How does it work?

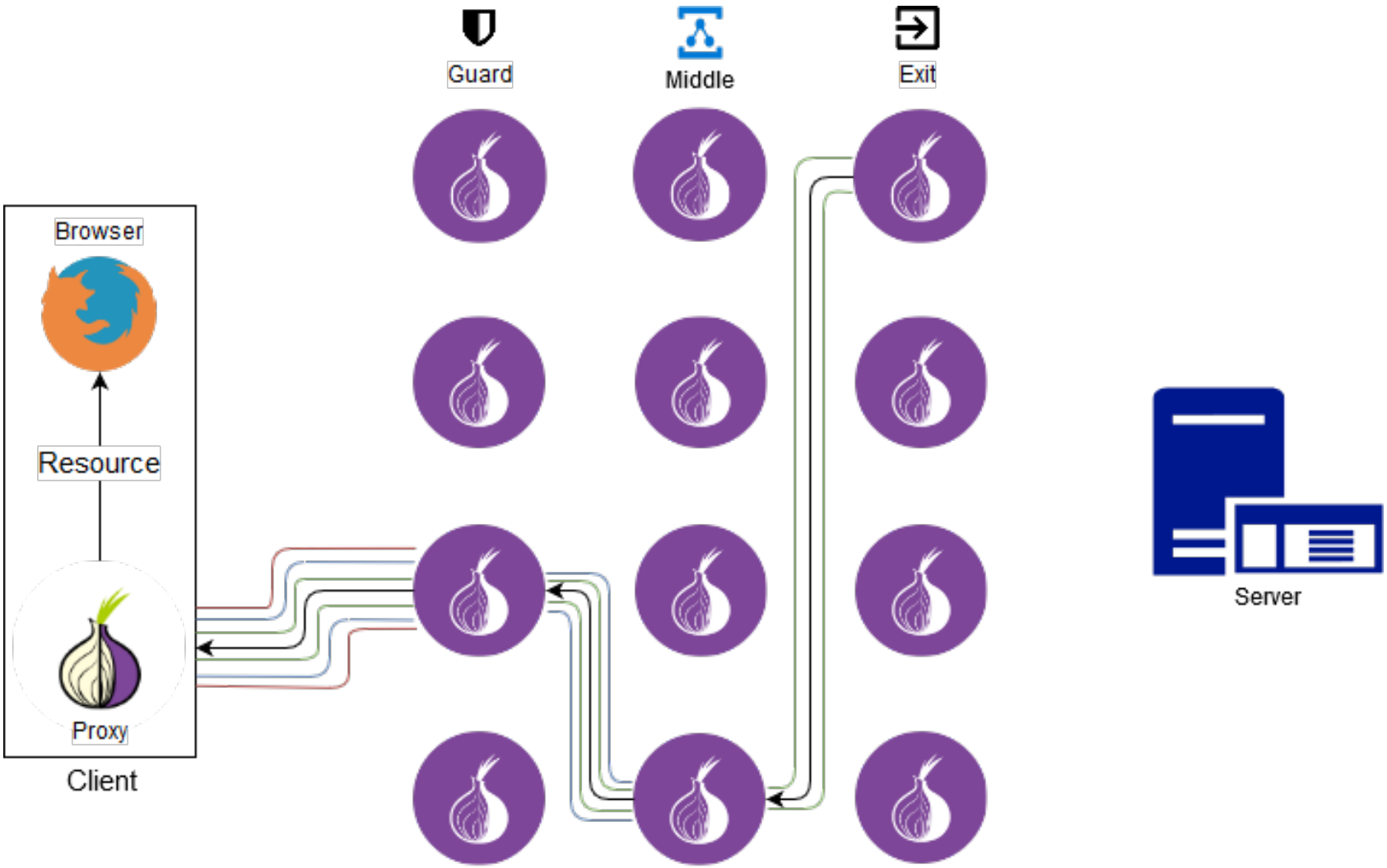

Figure: Data gets relayed back to the client

How does it work?

This provides anonymity to the client... but what about the **server**?

HS: How does it work?

Distributed Hash Table (DHT):

- Group of servers
- Each server holds a list of descriptors
- Descriptors contain information on how to contact the service

HS: How does it work?

The publishing of the Hidden Service

HS: How does it work?

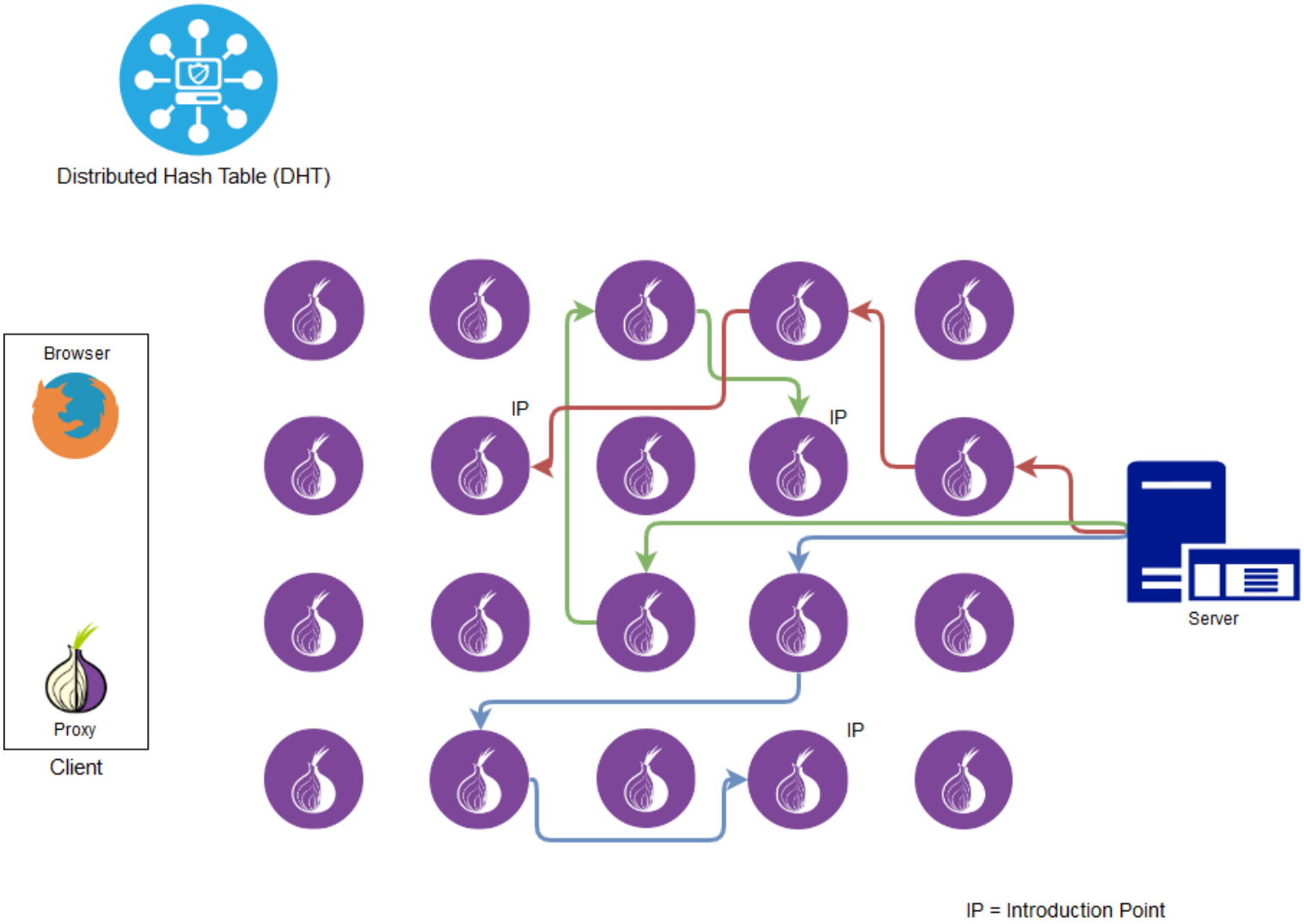

Figure: Server selection of Introduction Points

HS: How does it work?

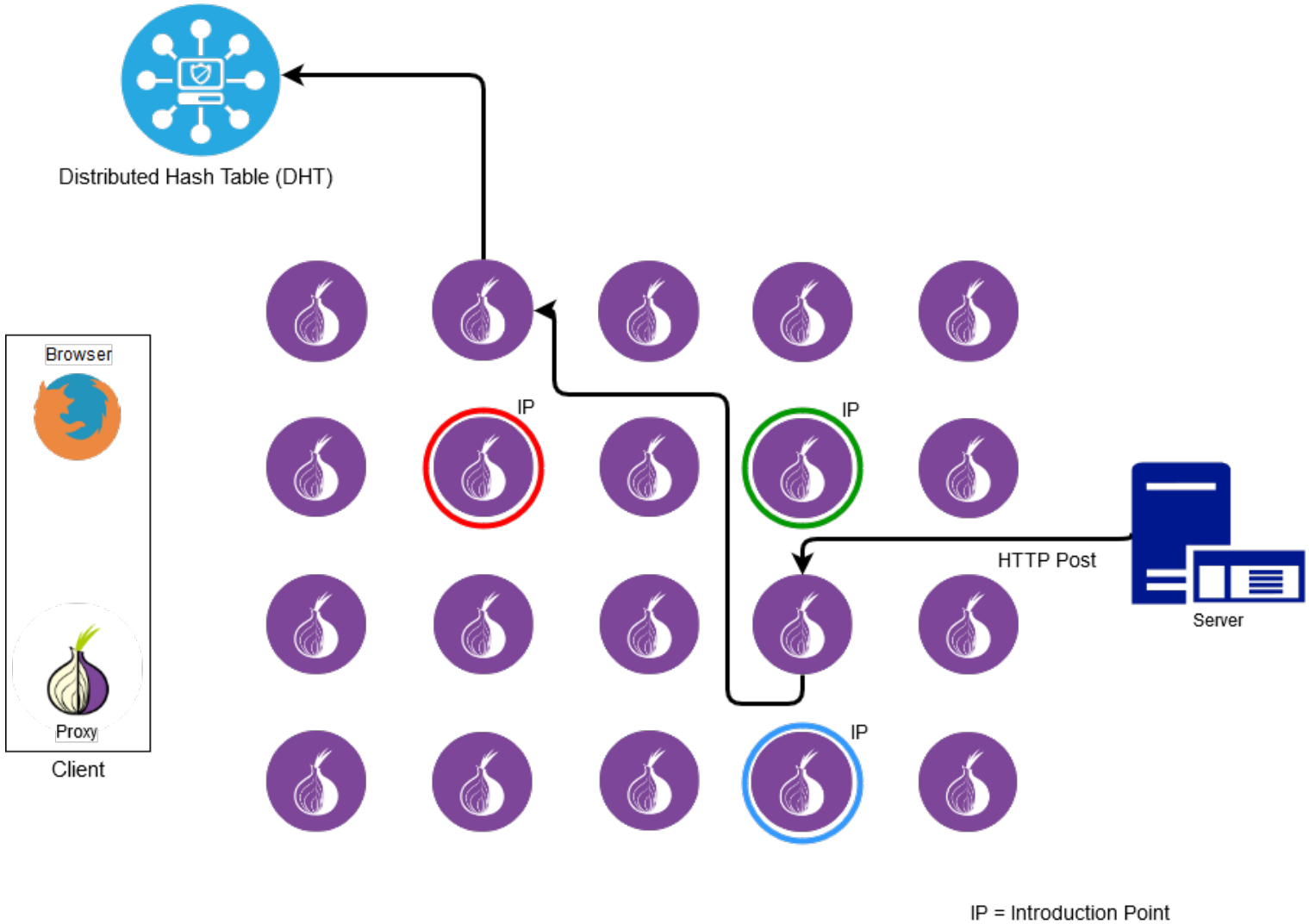

Figure: Server publishing descriptor to DHT

Client connection to hidden service

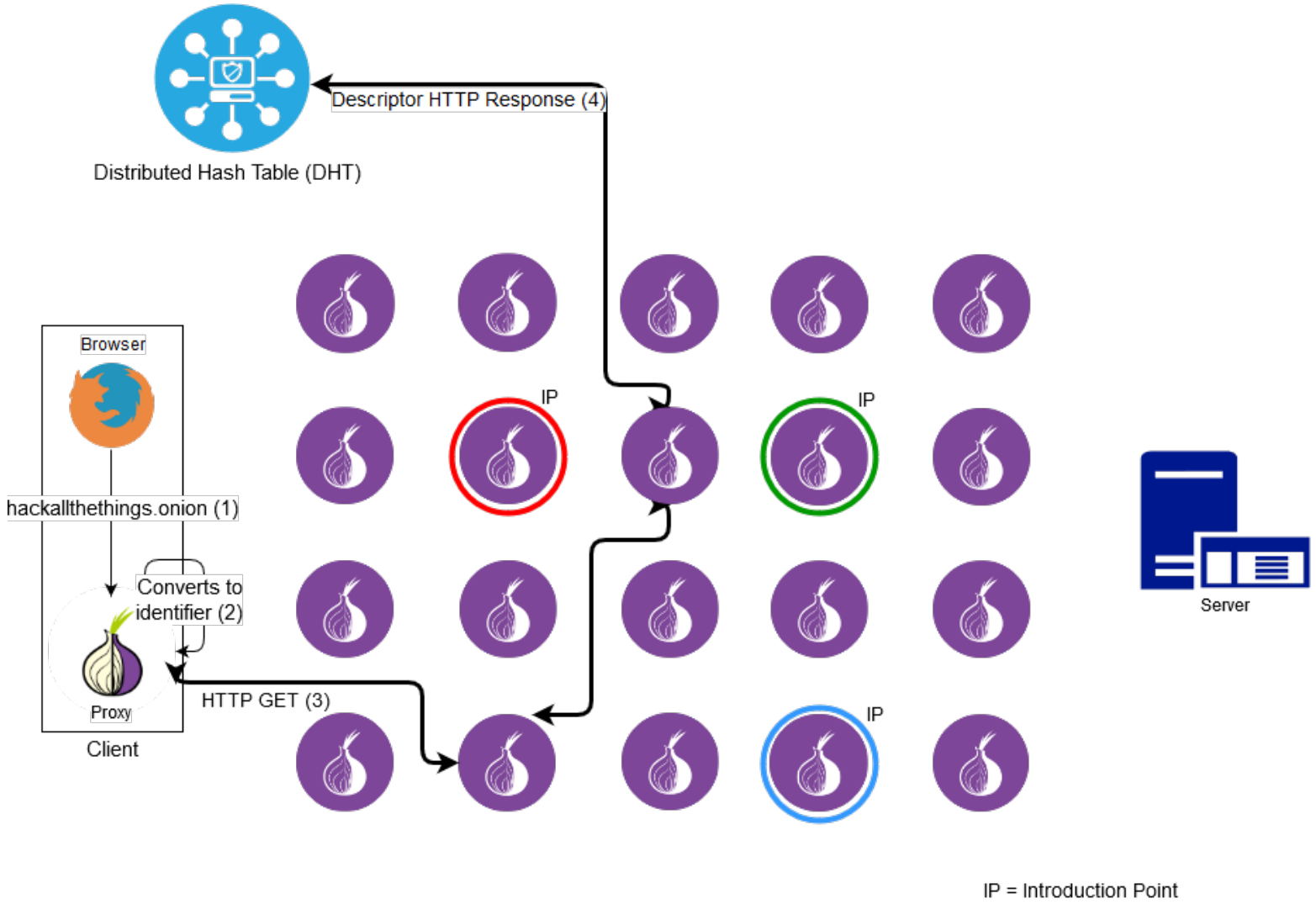

Figure: From browser request to receiving the descriptor from the DHT

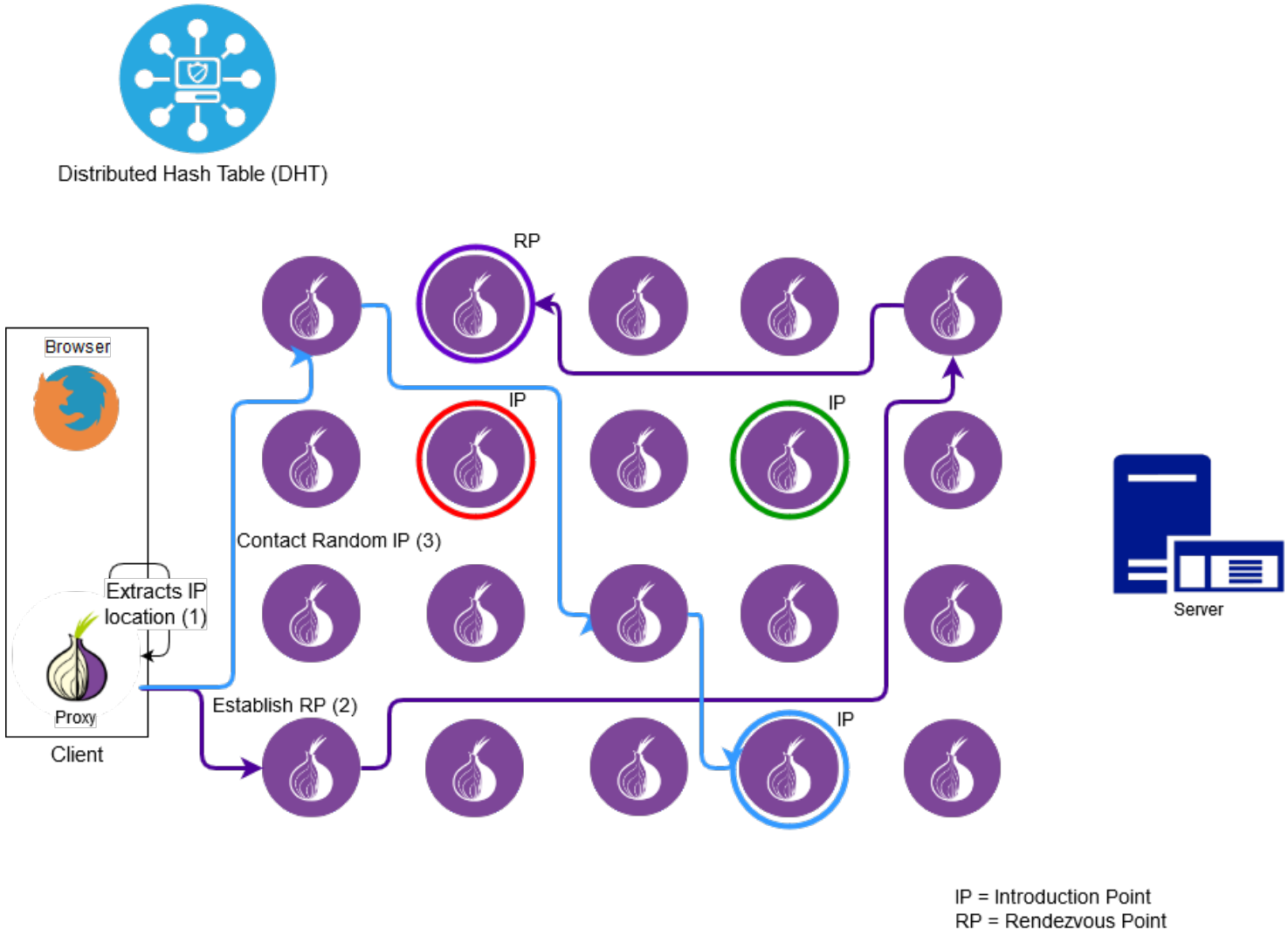

Figure: Rendezvous Point selection and contacting the Hidden Service

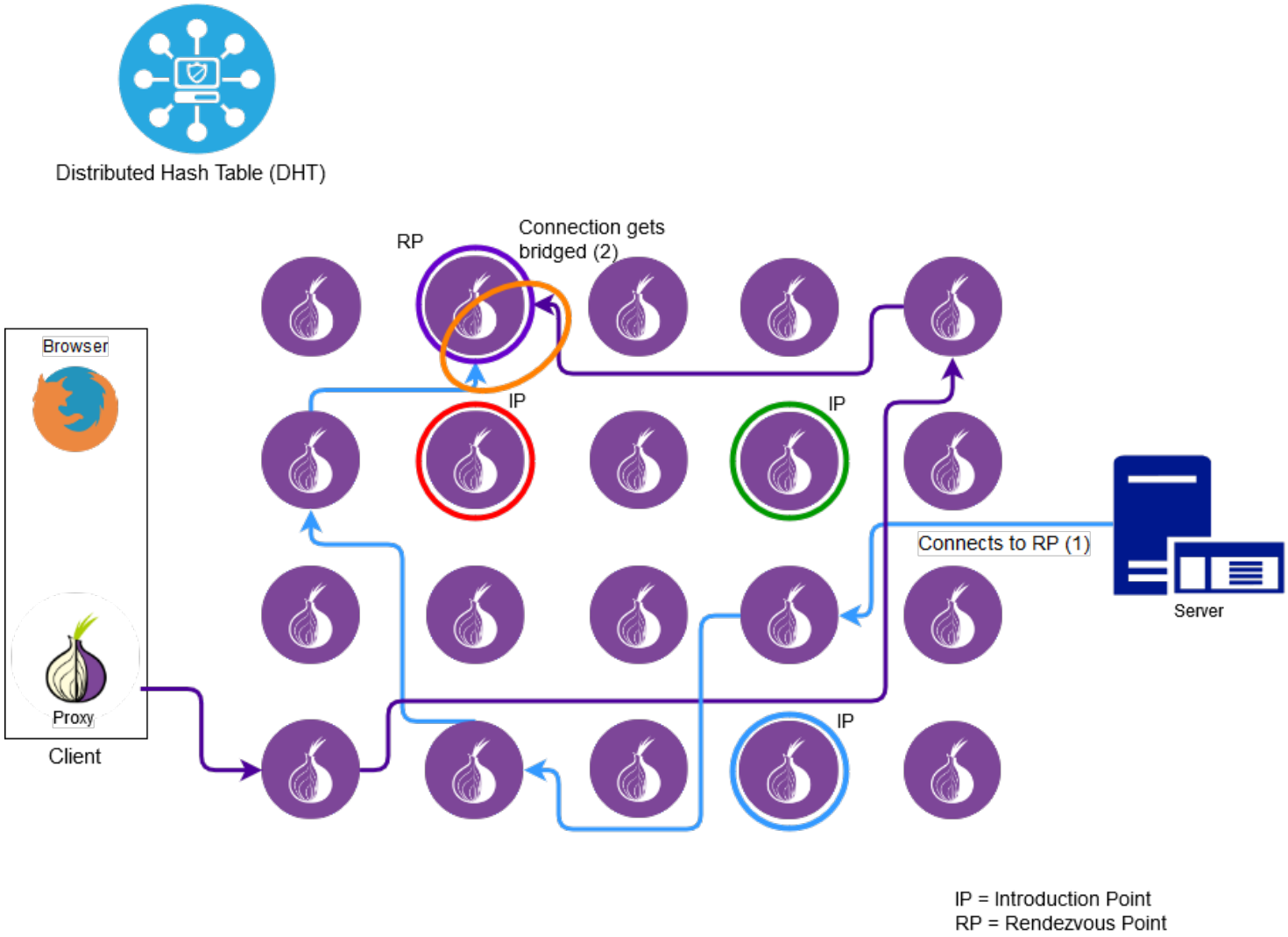

Figure: Server connection to RP and bridging of both circuits

HS: The protocol specified

Protocol received several changes through out the project lifetime. The protocol versions are:

- V0
- V2 (0.2.0.10-alpha+)
- V3 (0.3.0.8)

HS: The protocol specified

Protocol received several changes through out the project lifetime. The protocol versions are:

- V0
  - First version
  - No encryption
  - Requests made to HSDir directly with onion link (Supposed to be **Hidden!!**)
  - Deprecated in 0.2.2.1-alpha...no more V0 legacy ;-)
- V2 (0.2.0.10-alpha+)
- V3 (0.3.0.8)

HS: The protocol specified

Protocol received several changes through out the project lifetime. The protocol versions are:

- V0
- V2 (0.2.0.10-alpha+)
  - Second version
  - Encrypted Introduction points, but link still encoded in the clear text part
  - 16 characters link - yyhws9optuwiwsns.onion
- V3 (0.3.0.8)

HS: The protocol specified

Protocol received several changes through out the project lifetime. The protocol versions are:

- V0
- V2 (0.2.0.10-alpha+)
- V3 (0.3.0.8)
  - Current version
  - Clear text metadata for identification of descriptor
  - Rest encrypted using a derivation of the onion link
  - 56 characters link - l5satjgud6gucryazcyvyvhuxhr74u6ygigiuyixe3a6ysis67ororad.onion

HS: The protocol specified

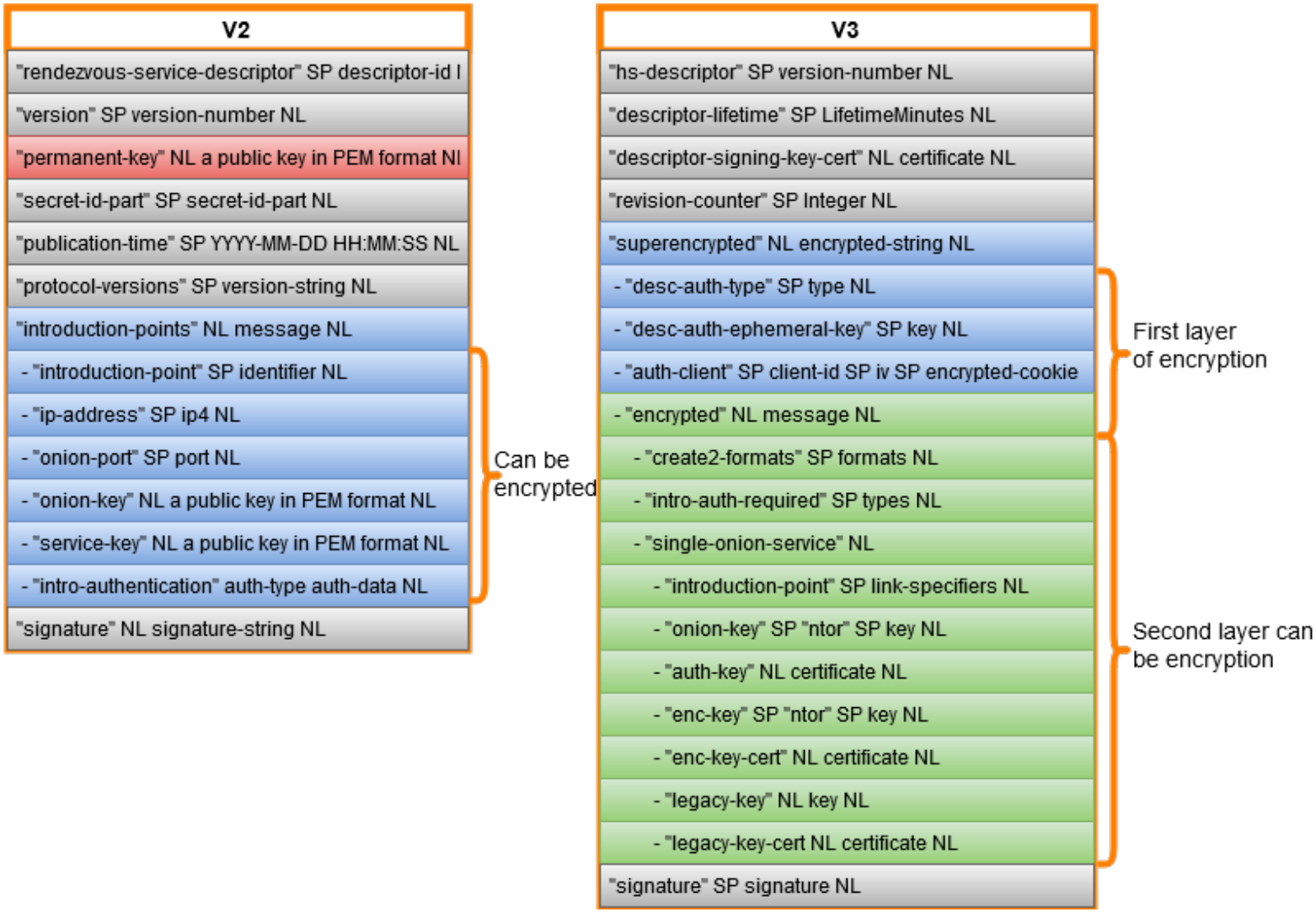

Figure: Differences between V2 and V3 descriptor

Method

Several routes to acquire the onion links:

- Scrapping
- Bruteforcing
- Sniffing
- Dumping Memory from the HSDir

Method

Several routes to acquire the onion links:

- Scrapping
  - Time consuming
  - Only links that have been shared in public domain
- Bruteforcing
- Sniffing
- Dumping Memory from the HSDir

Method

Several routes to acquire the onion links:

- Scrapping
- Bruteforcing
  - Infeasible - V3
  - Time - V2
- Sniffing
- Dumping Memory from the HSDir

Method

Several routes to acquire the onion links:

- Scrapping
- Bruteforcing
- Sniffing
  - Impossible
- Dumping Memory from the HSDir

Method

Several routes to acquire the onion links:

- Scrapping
- Bruteforcing
- Sniffing
- Dumping Memory from the HSDir
  - Requires HSDir (flag acquired 4 days from last down (Requires Stable flag which takes 5 days))
  - Impossible - V3

Memory Dumps

Dumping Memory - Very fruitful, V2 descriptors successfully extracted and decoded to acquire the onion link

Memory Dumps

Dumping Memory - Very fruitful, V2 descriptors successfully extracted and decoded to acquire the onion link

Created a proof of concept program for automating hourly memory dumps of multiple Tor proxys

Memory Dumps

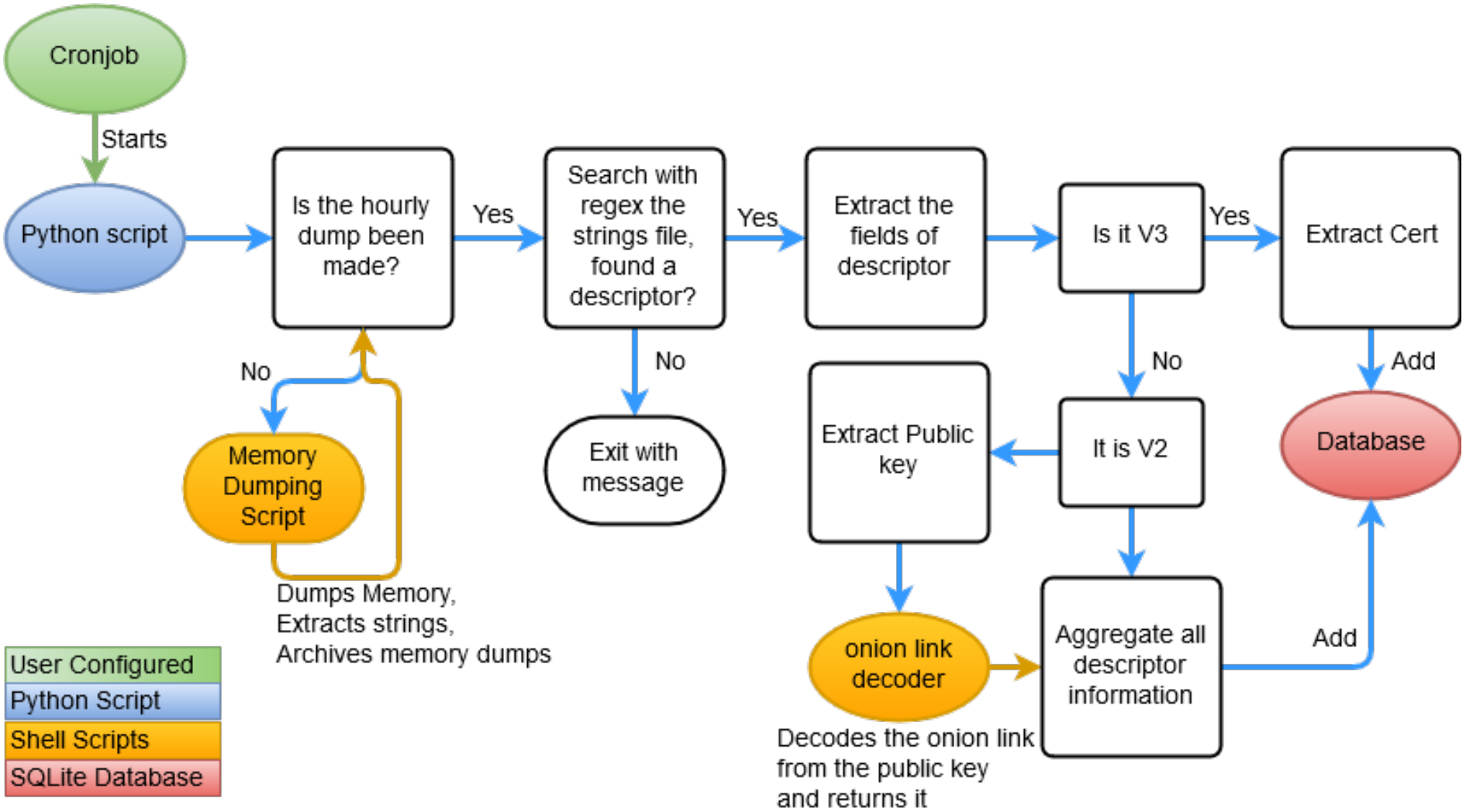

Figure: Process flow diagram of the link extraction PoC

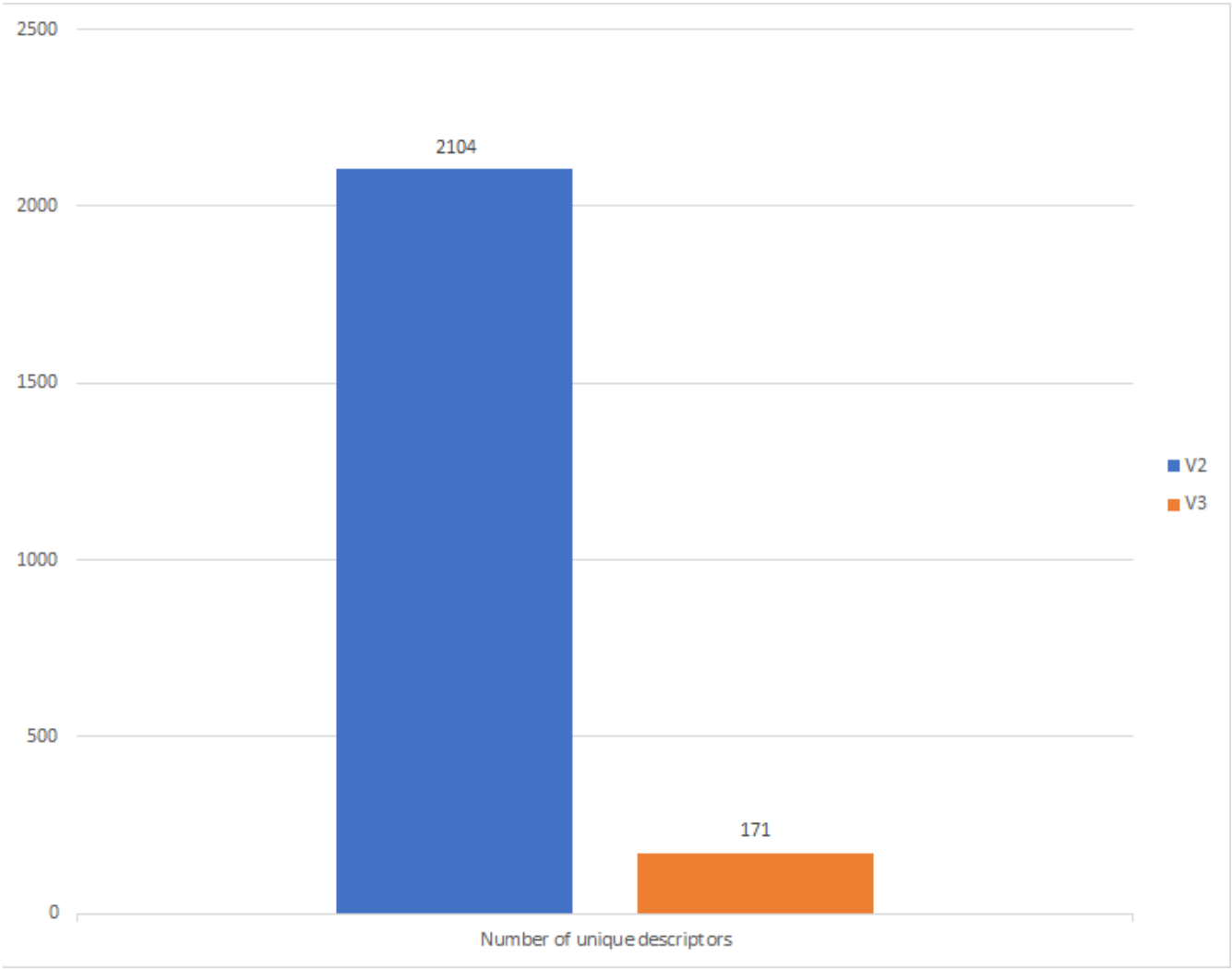

Figure: Graph showing the number of unique descriptors extracted in 5 days

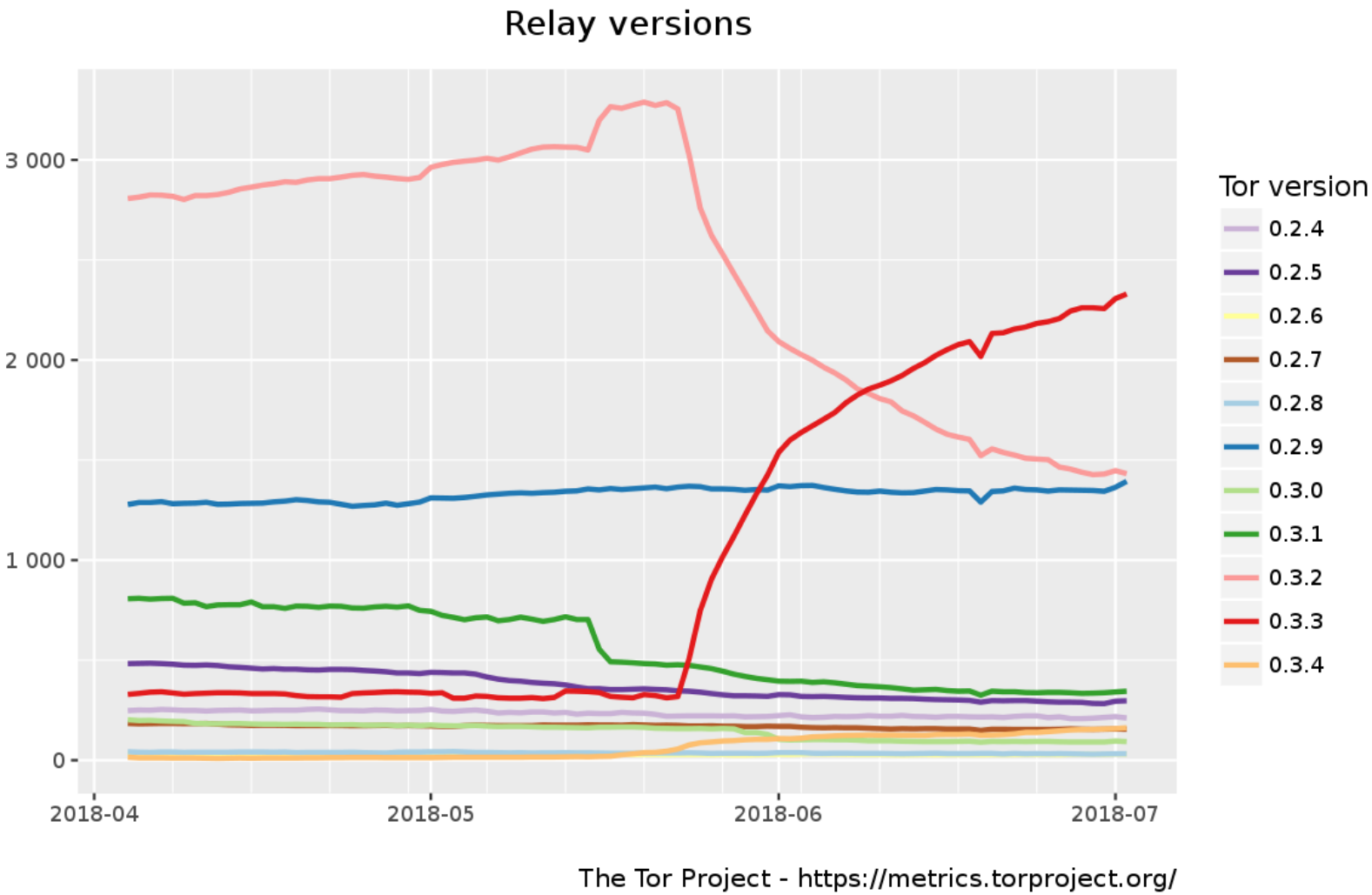

Figure: Graph showing tor versions currently being run: V2≥0.2.0.10 - V3≥0.3.0.8 (15/18)

In conclusion...

How feasible is the acquisition of hidden service links (onion links)?

We can conclude from the findings that:

- 2104 unique V2 links five days of running memory dumps from the 105069 reported by tor metrics [1]
- Two relays for less than 26 euros - Very good cost/efficiency balance
- V2: Even though IP encryption enabled, the encoded links are always present on the clear
- V3 Enabled Relays != V3 > V2

[1] `https://metrics.torproject.org/hidserv-dir-onions-seen.html`

In conclusion...

But **hidden services** are supposed to be **hidden** unless specifically gived the address. So to solve this the recommendation is to simply:

- Use the latest features of the software
- Deprecate the V2 protocol
- If not possible use V2 IP encryption

What can still be done

With results aggregated, this stage becomes a stepping stone for targeted intel extraction such as:

- Verifying which links are alive (big portion could be on demand file sharing, short lived hidden services)
- Identifying type of service running behind the onion link

What can still be done

With results aggregated, this stage becomes a stepping stone for targeted intel extraction such as:

- Verifying which links are alive (big portion could be on demand file sharing, short lived hidden services)
- Identifying type of service running behind the onion link

With some research into how to capture the requests for V2 descriptors:

- Easy to convert from from link to id
- Correlate id captured to addresses acquired
  - Possibly discerning traffic to previously discovered C&Cs

Questions?

References


Biryukov, Alex and Pustogarov, Ivan and Weinmann, Ralf-Philipp (2013)
Trawling for tor hidden services: Detection, measurement, deanonymization
Security and Privacy (SP), 2013 IEEE Symposium on pp.80 – 94.