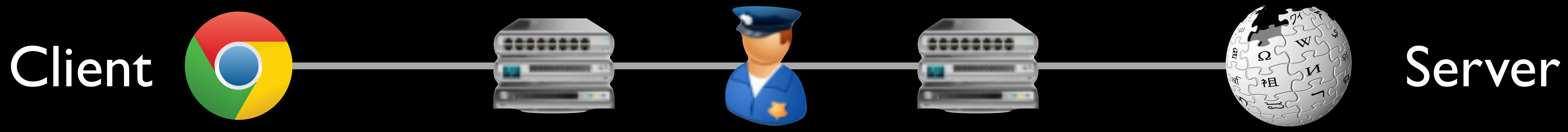
Geneva:
Evolving Censorship Evasion

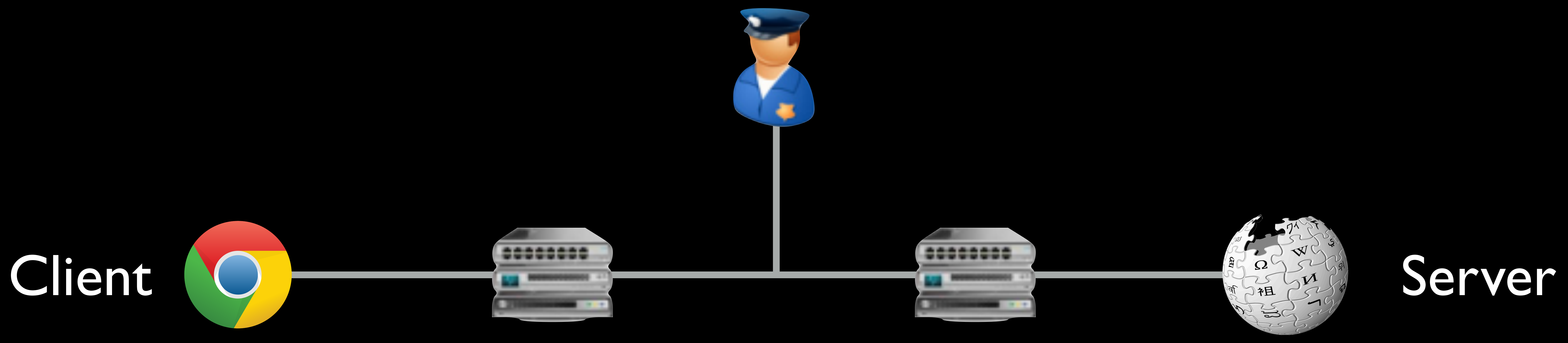
Kevin Bock

UNIVERSITY OF MARYLAND

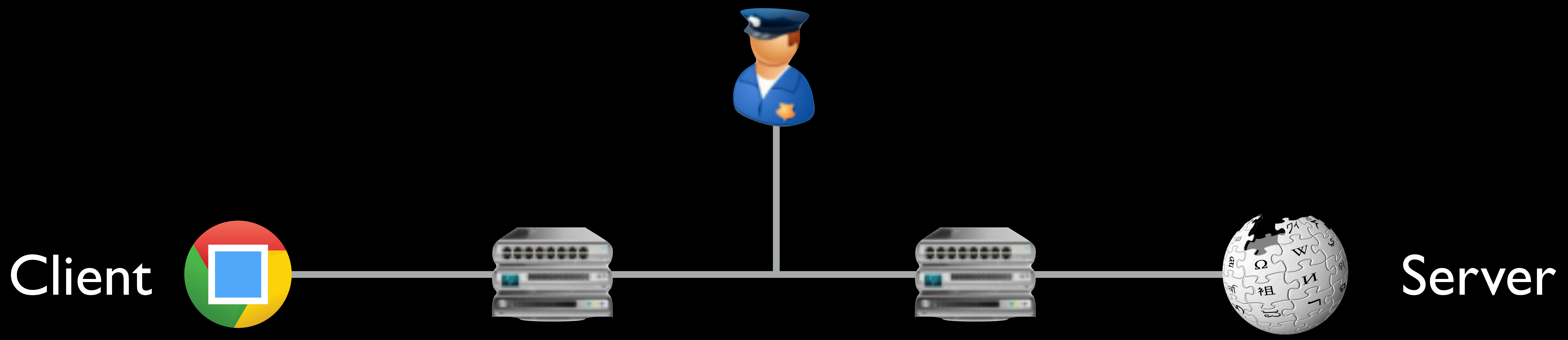
In-network censorship by nation-states



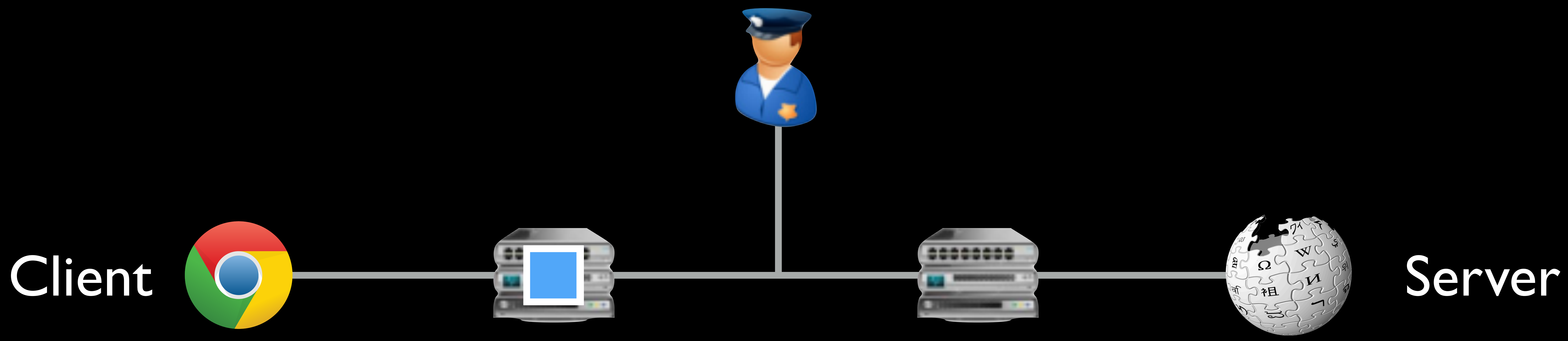
In-network censorship by nation-states



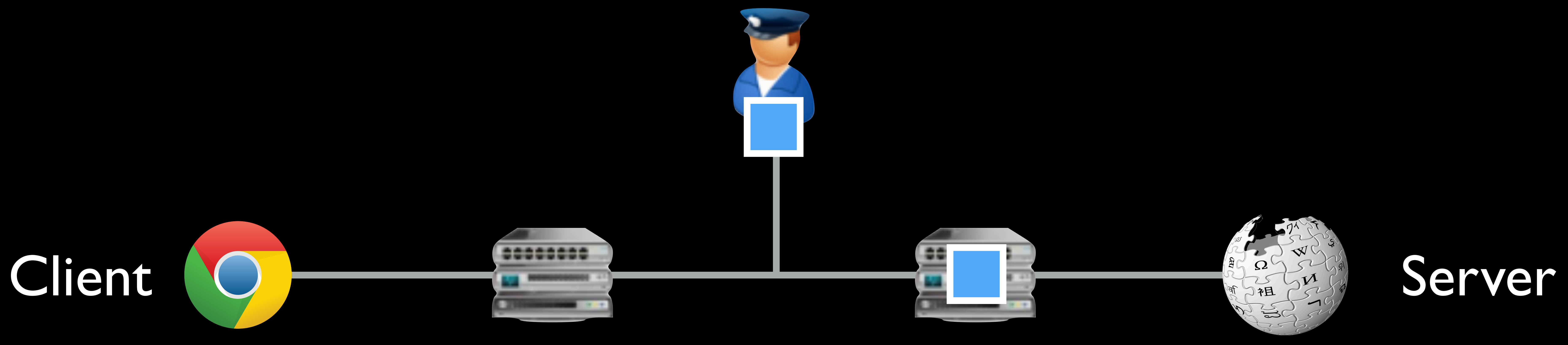
In-network censorship by nation-states



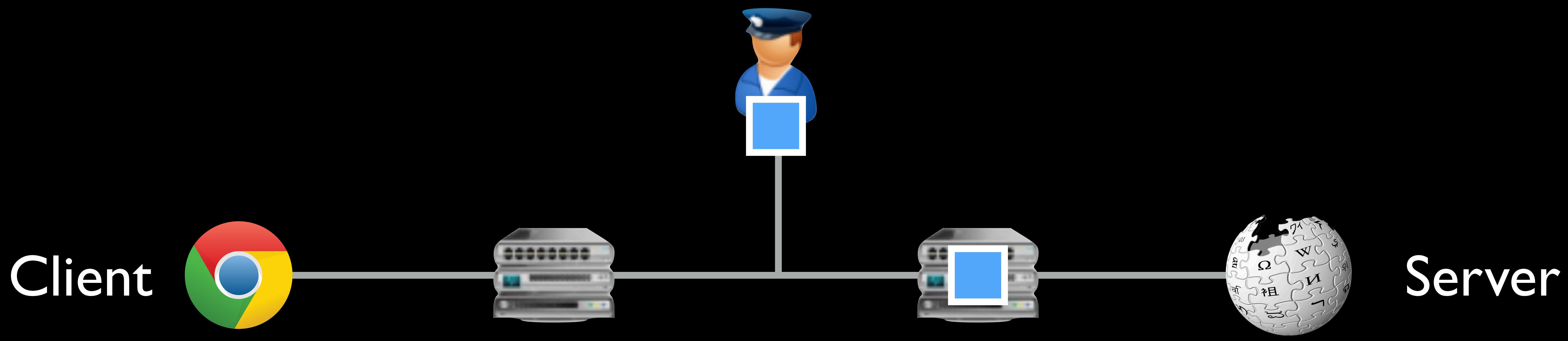
In-network censorship by nation-states



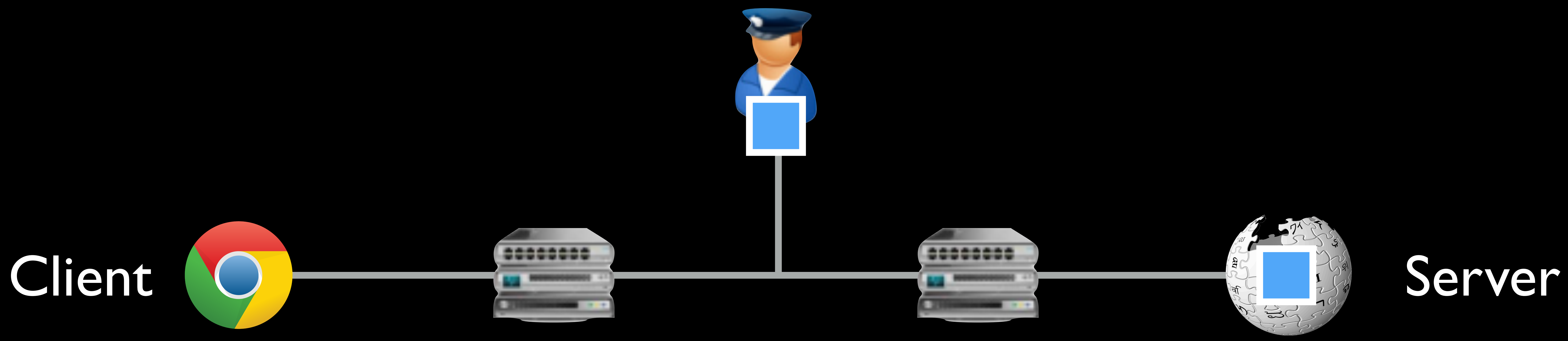
In-network censorship by nation-states



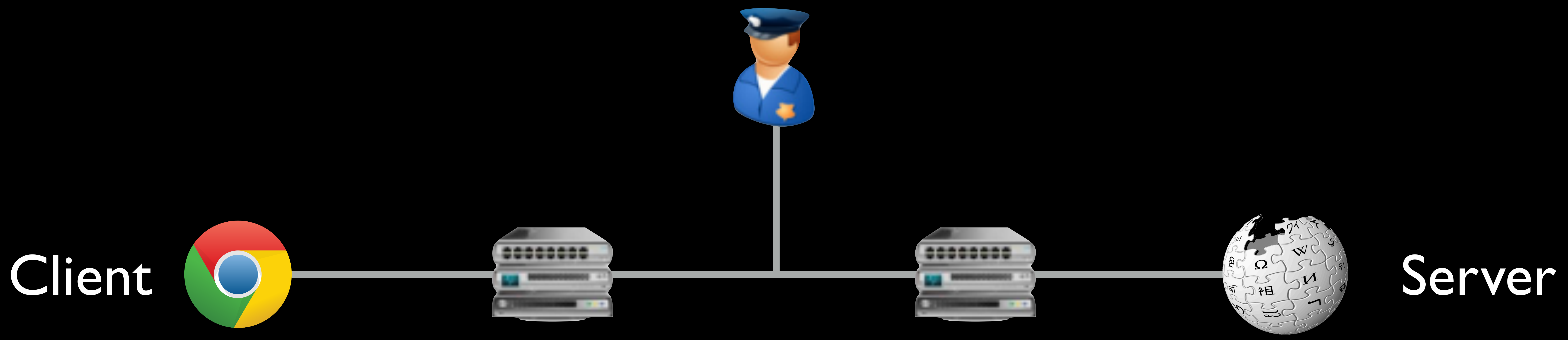
In-network censorship by nation-states



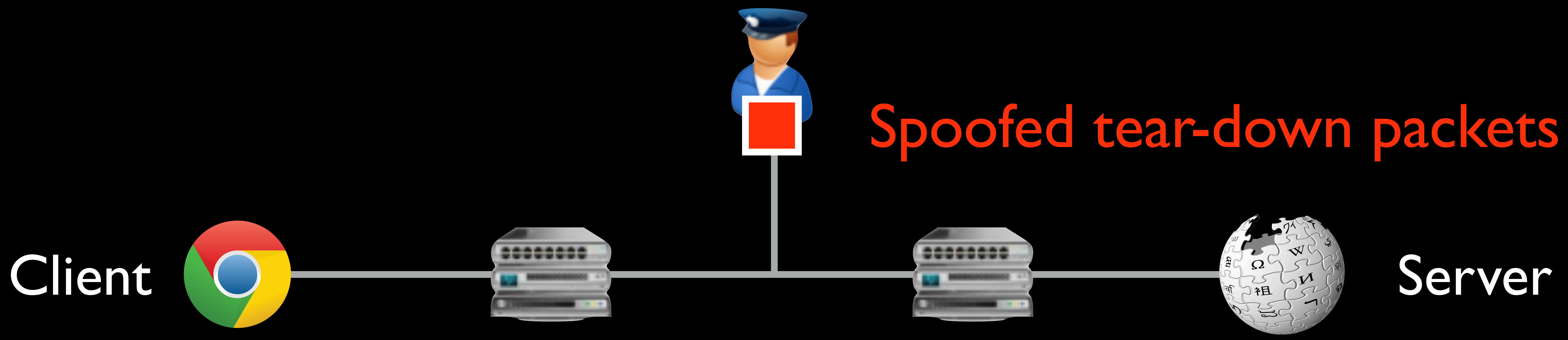
In-network censorship by nation-states



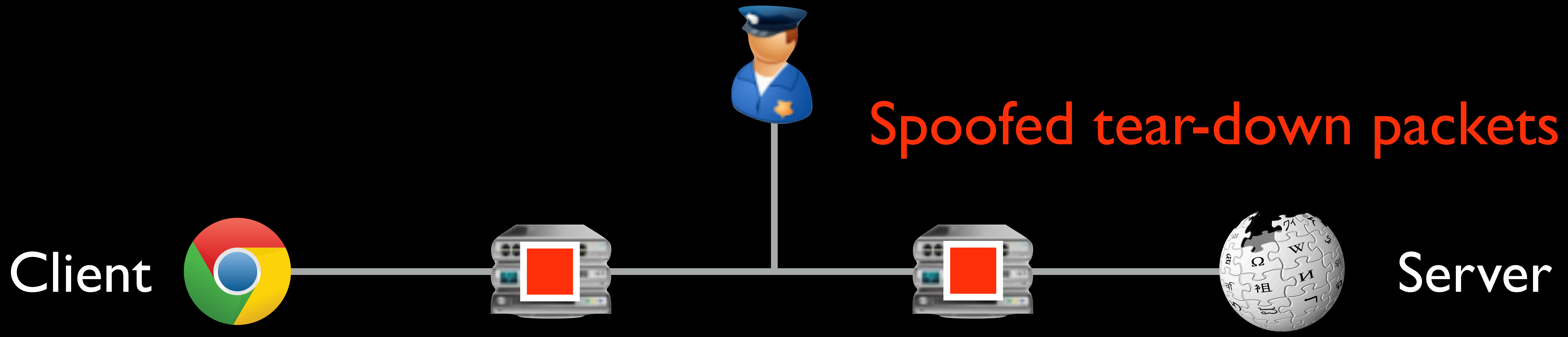
In-network censorship by nation-states



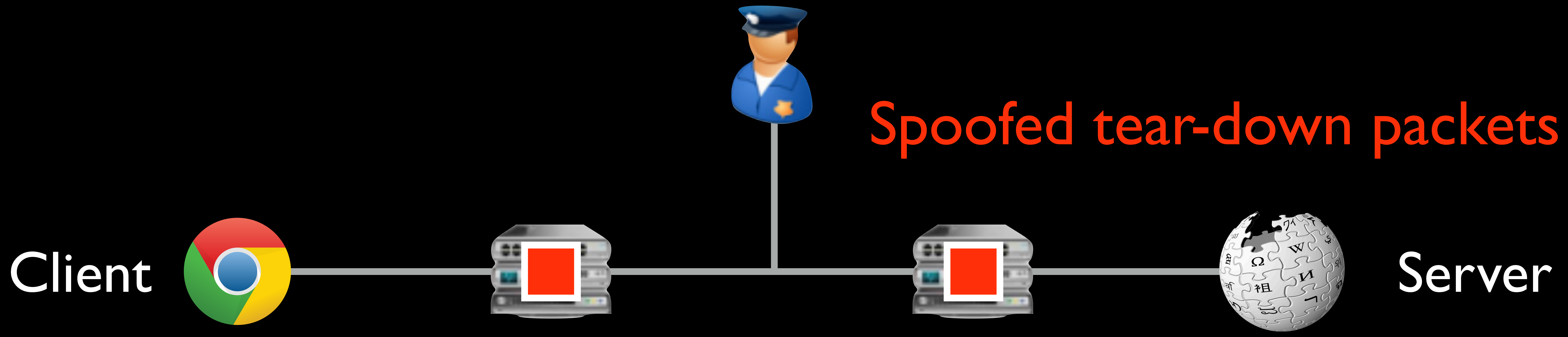
In-network censorship by nation-states



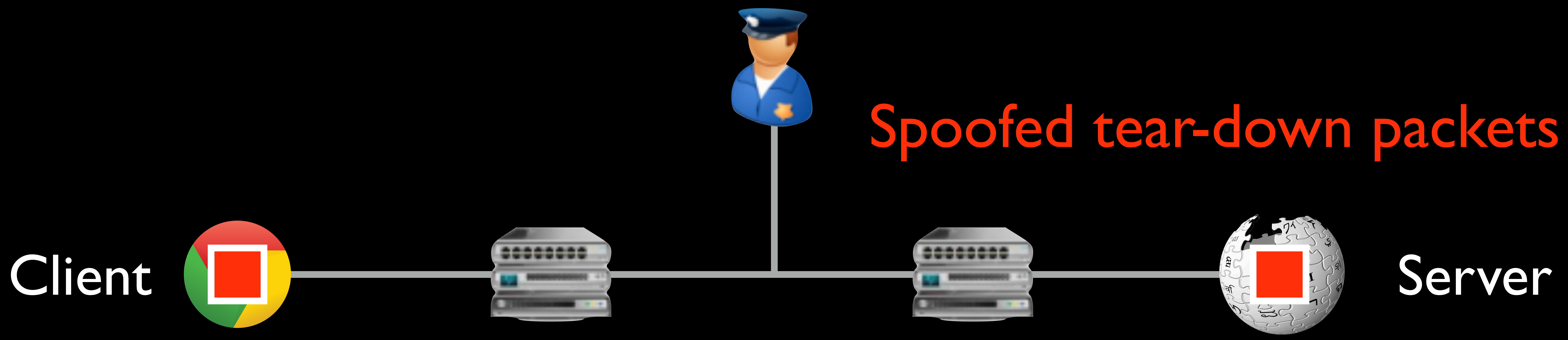
In-network censorship by nation-states



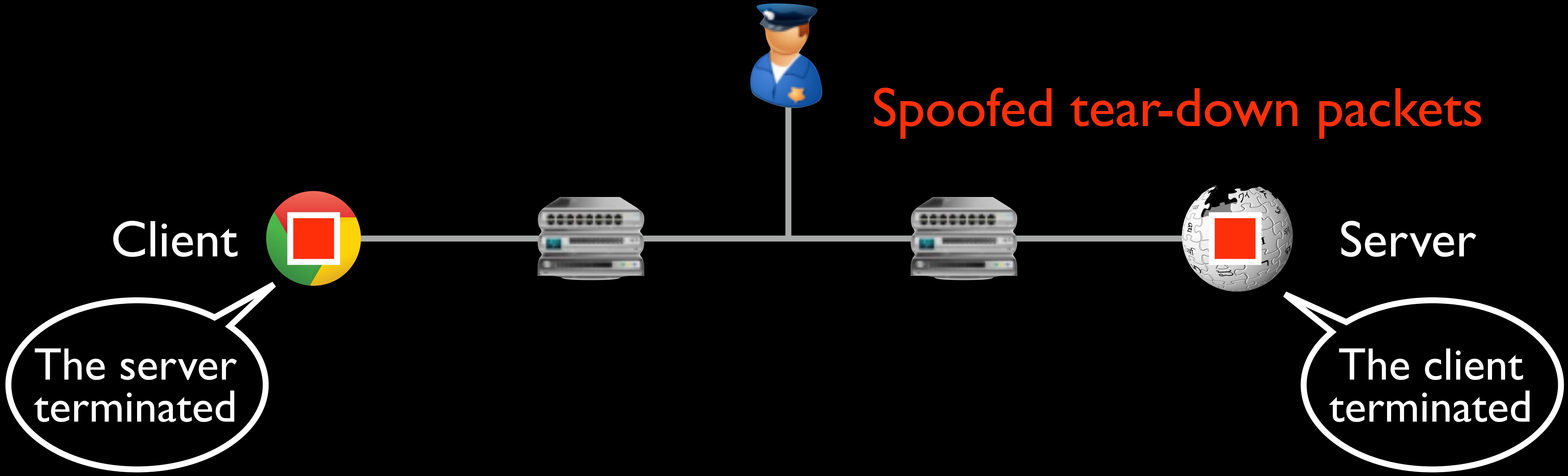
In-network censorship by nation-states



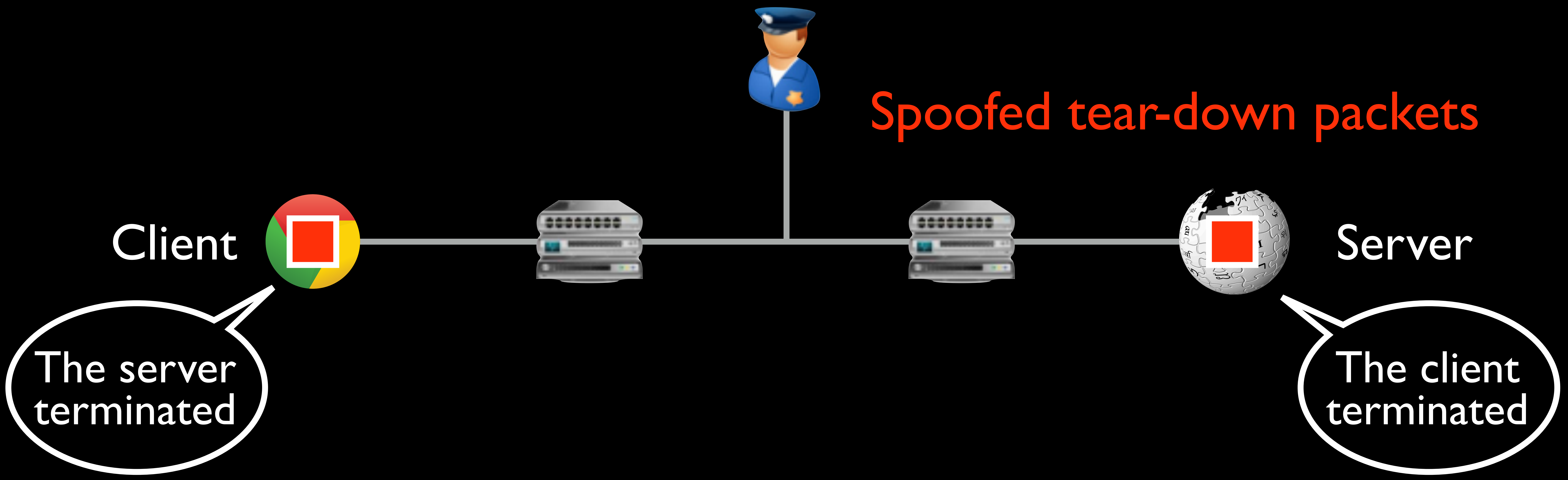
In-network censorship by nation-states



In-network censorship by nation-states

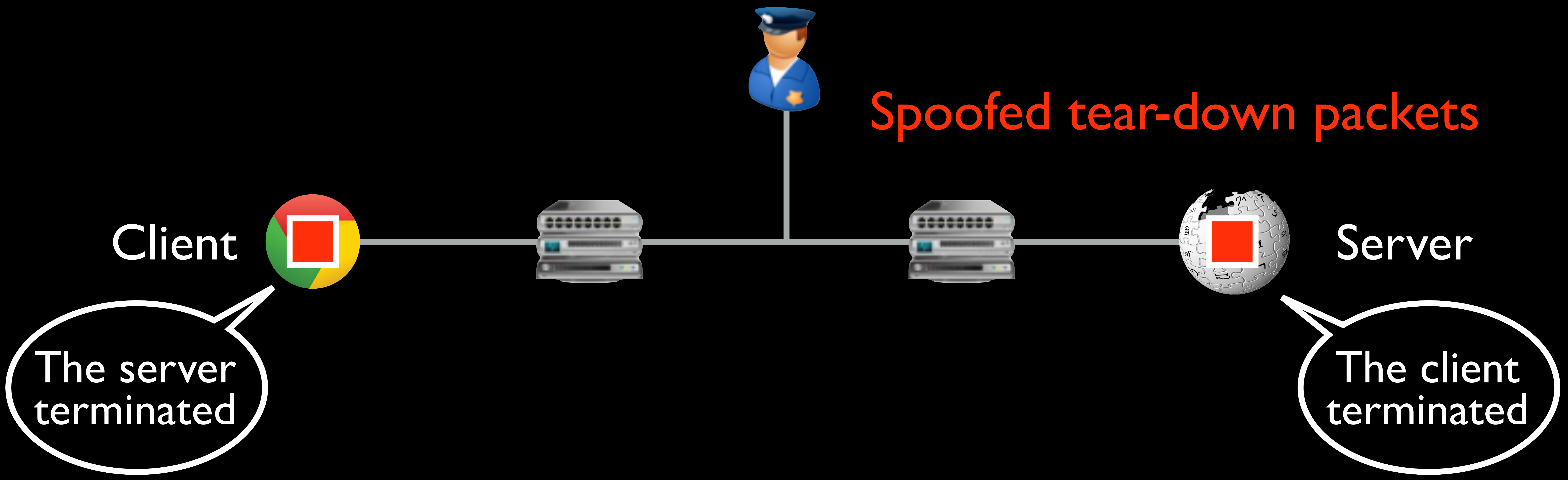


In-network censorship by nation-states



Requires *per-flow state*

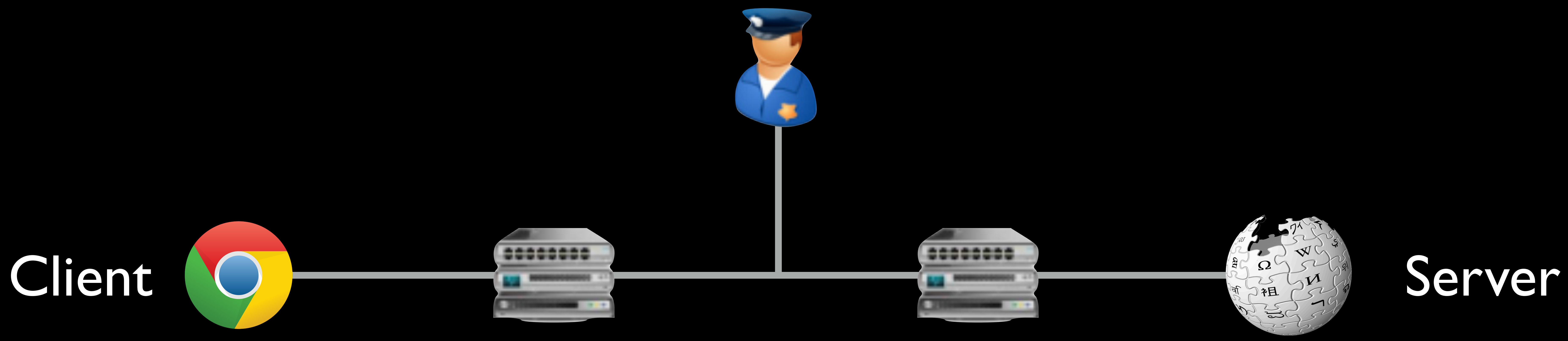
In-network censorship by nation-states



Requires *per-flow state*

Censors necessarily *take shortcuts*

In-network censorship by nation-states

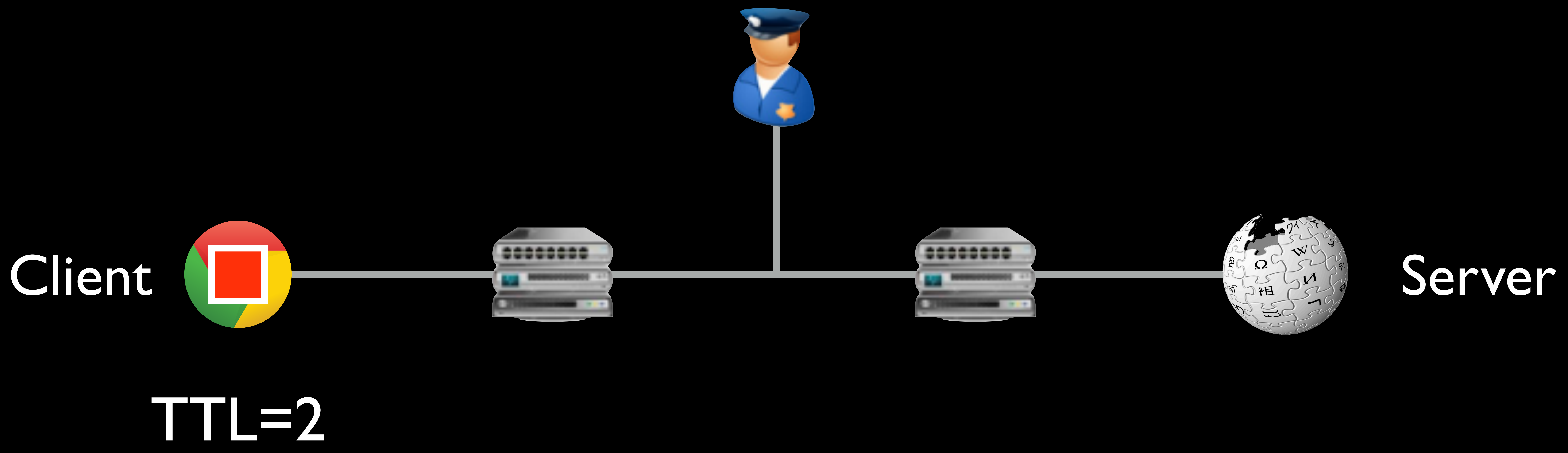


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

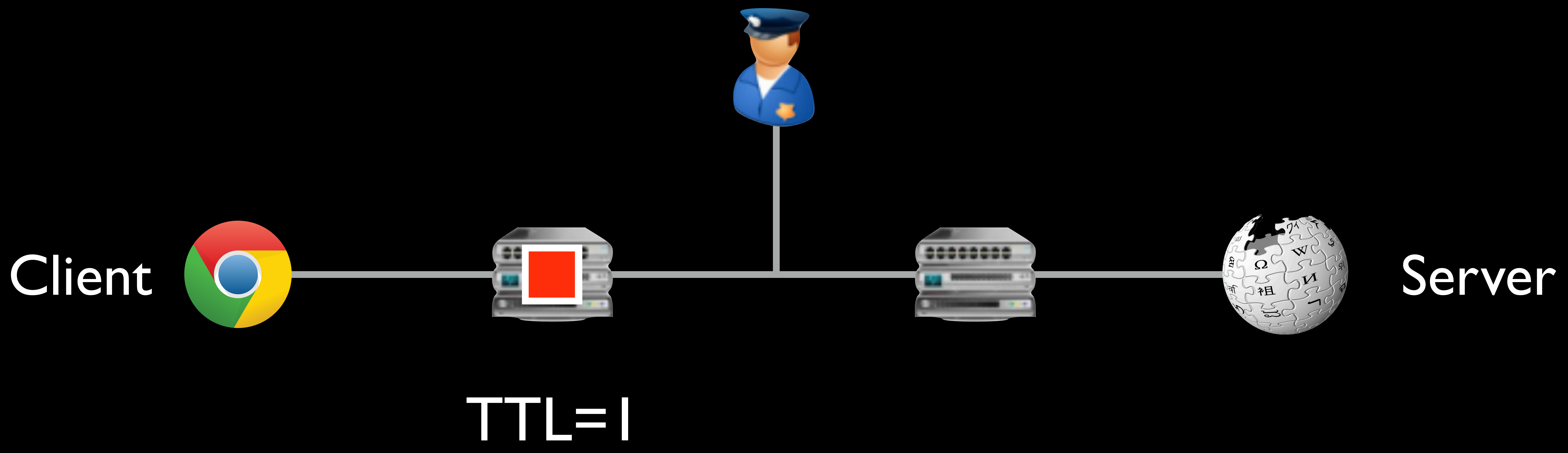


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

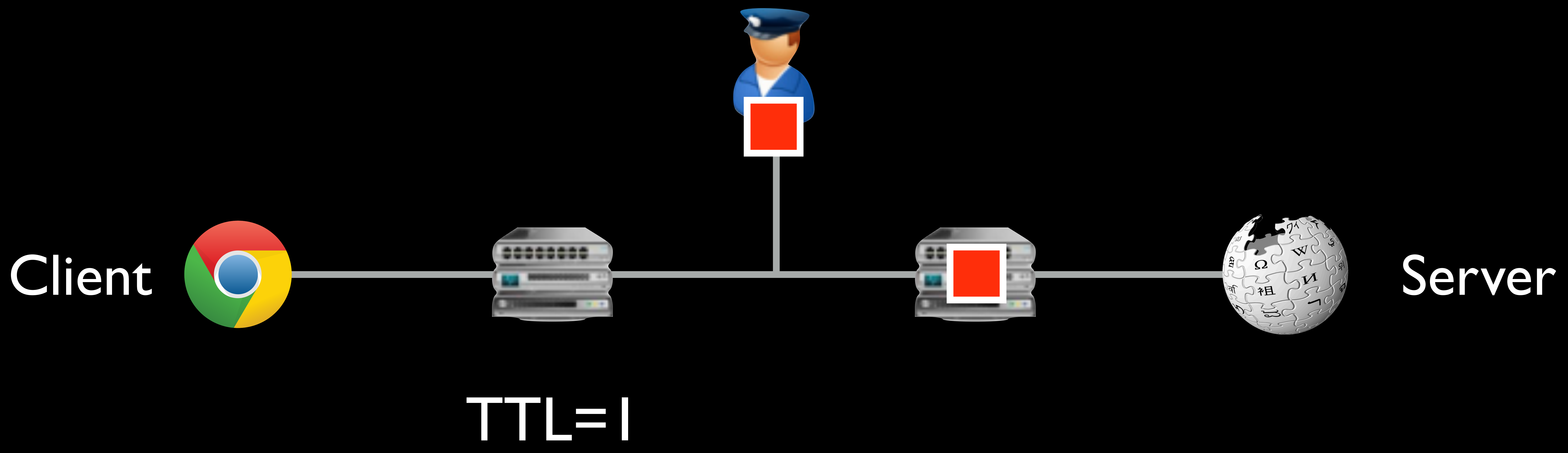


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

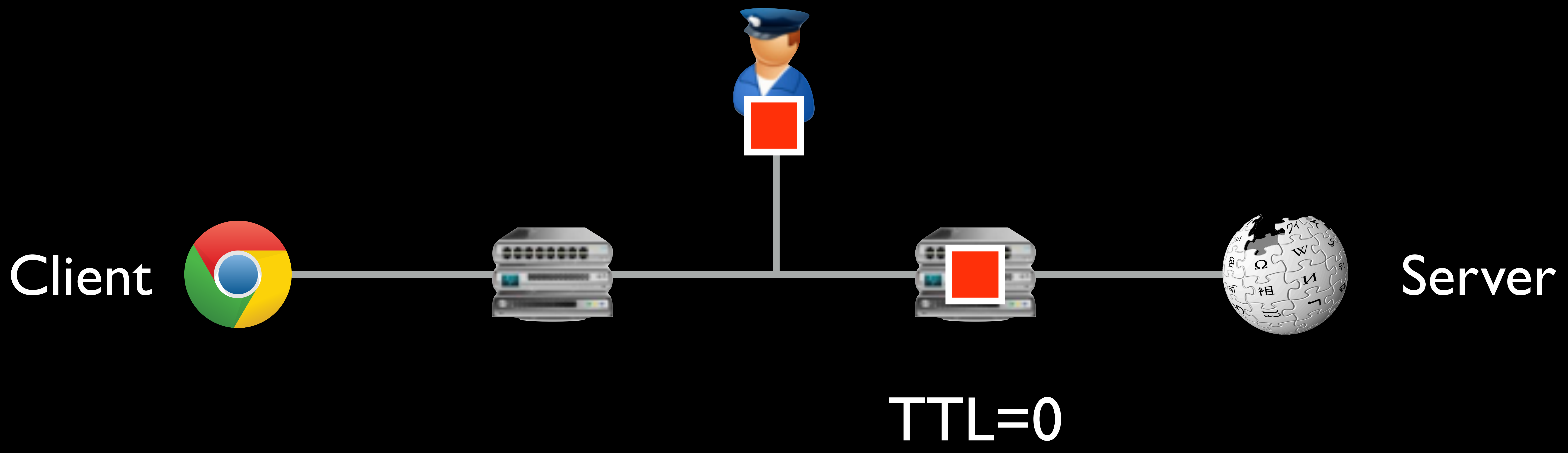


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

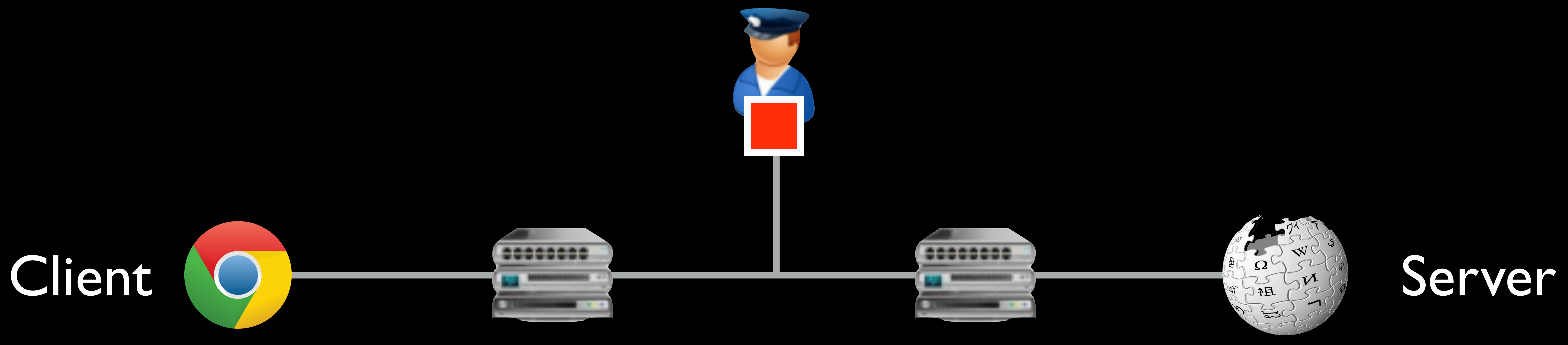


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

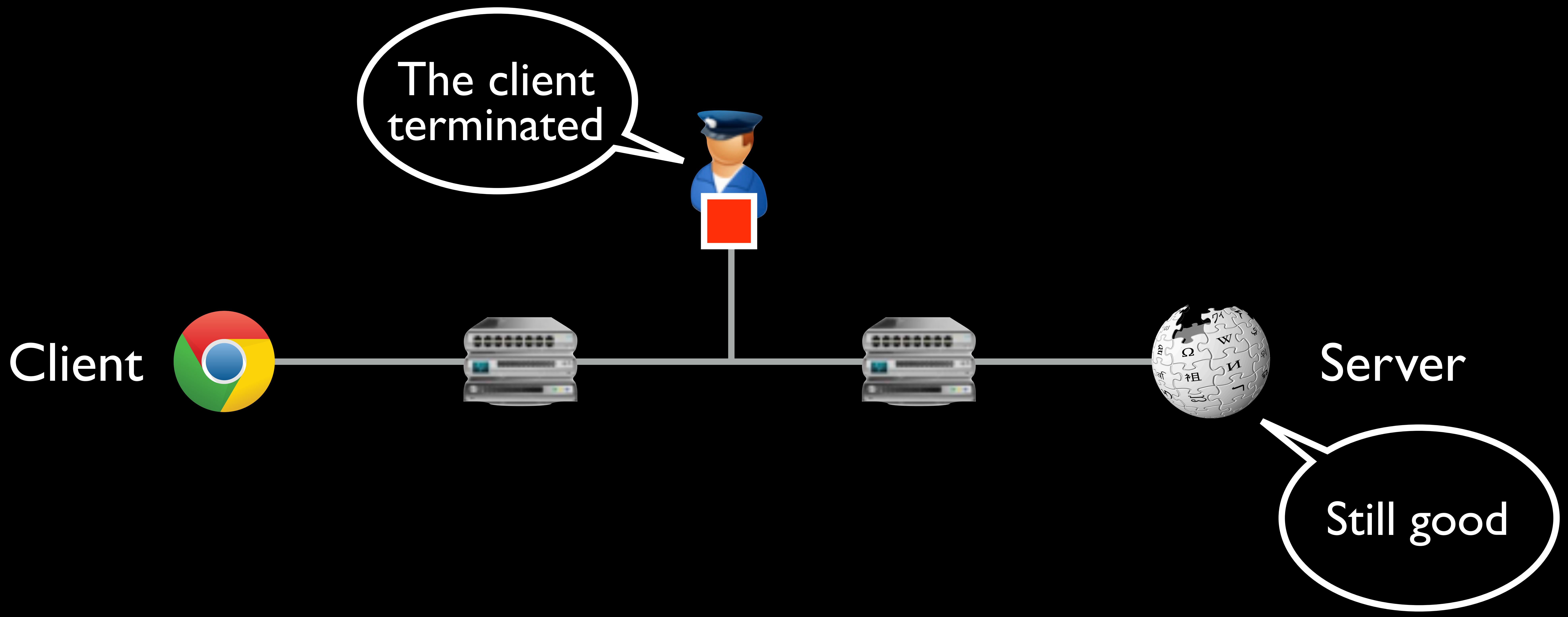


Requires *per-flow state*

Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

In-network censorship by nation-states

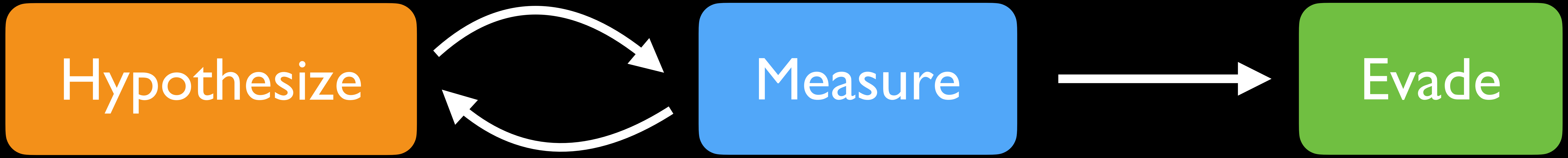


Requires *per-flow state*

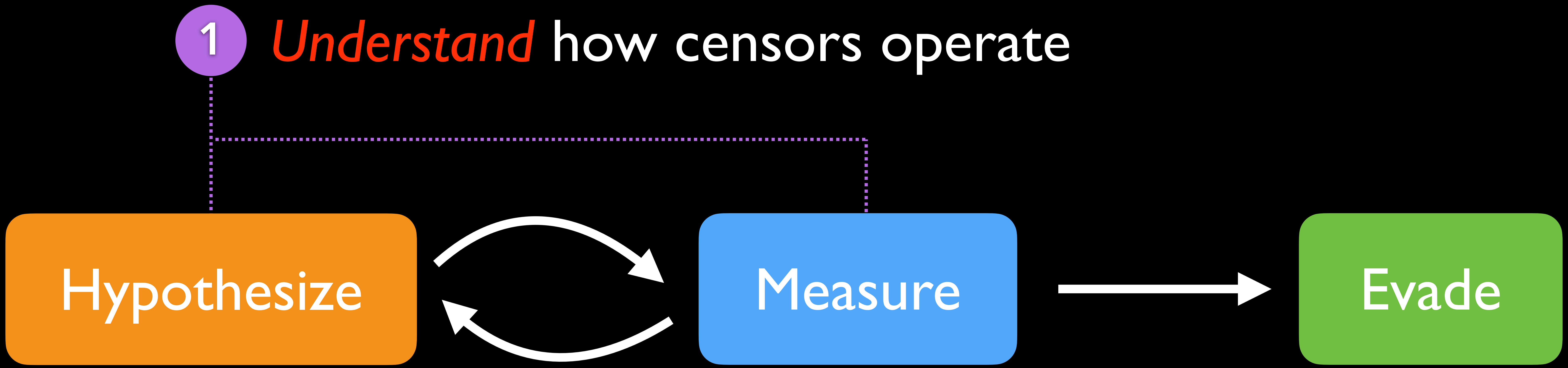
Censors necessarily *take shortcuts*

Evasion can take advantage of these shortcuts

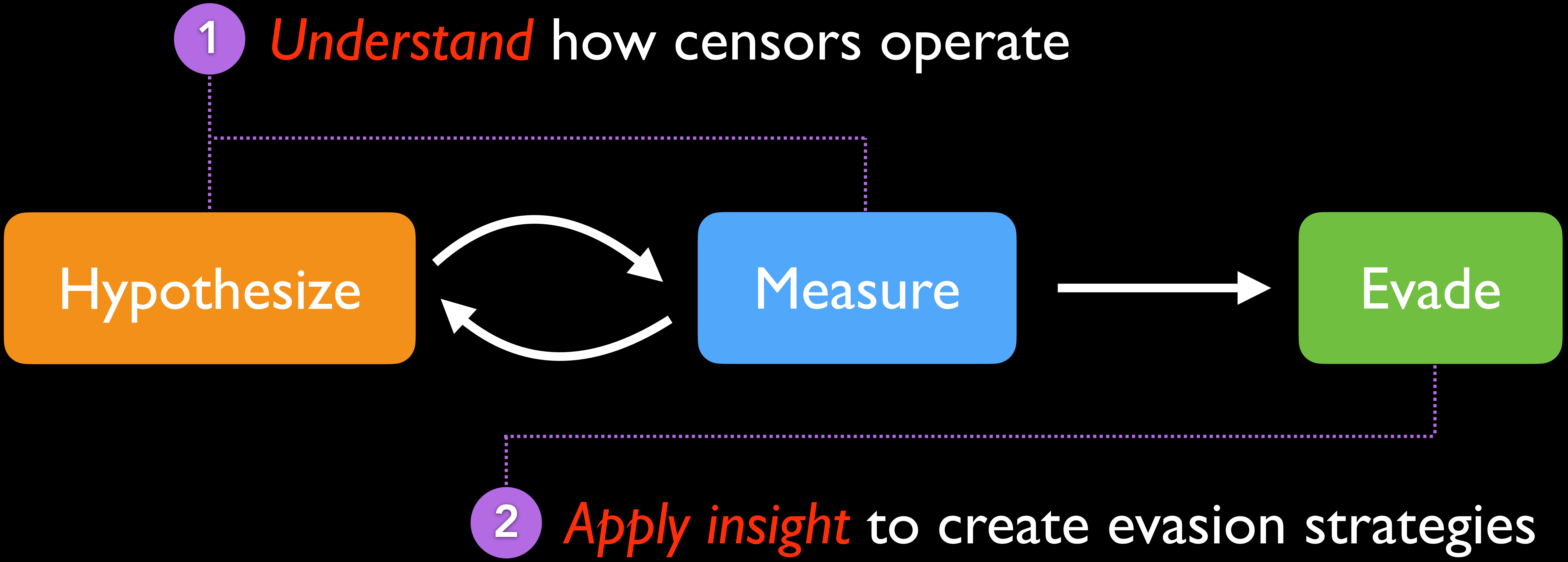
Censorship evasion research



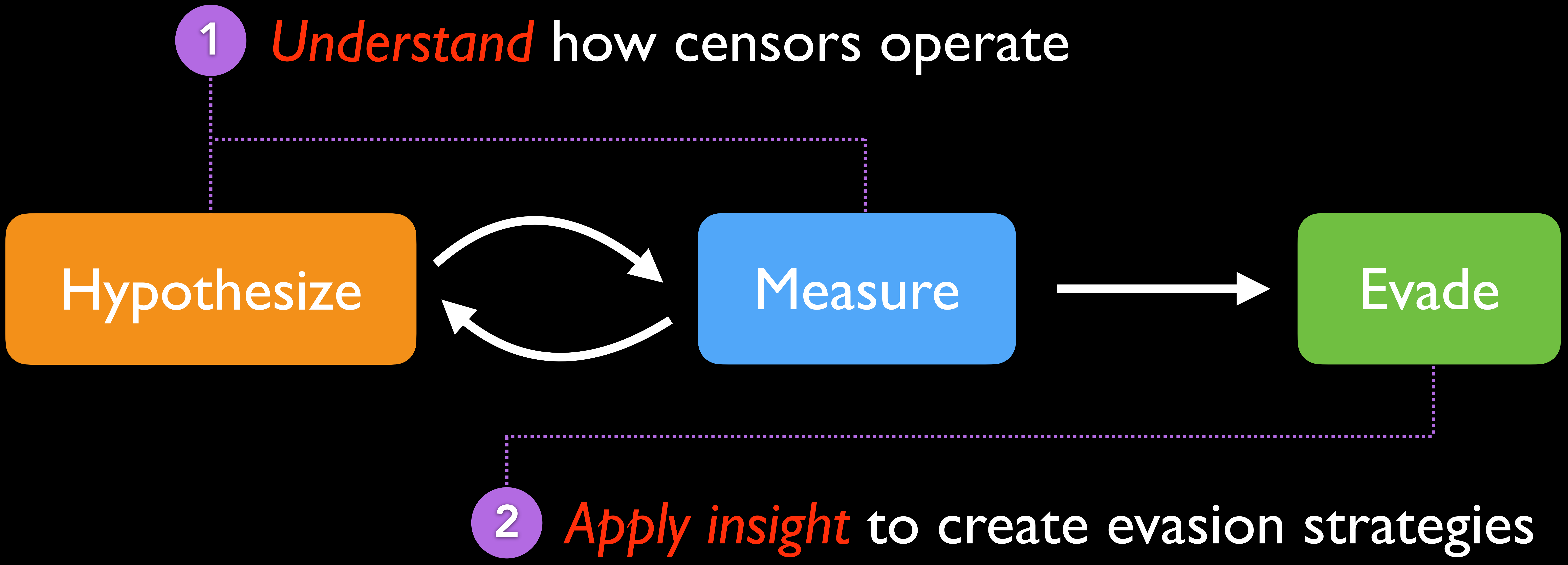
Censorship evasion research



Censorship evasion research

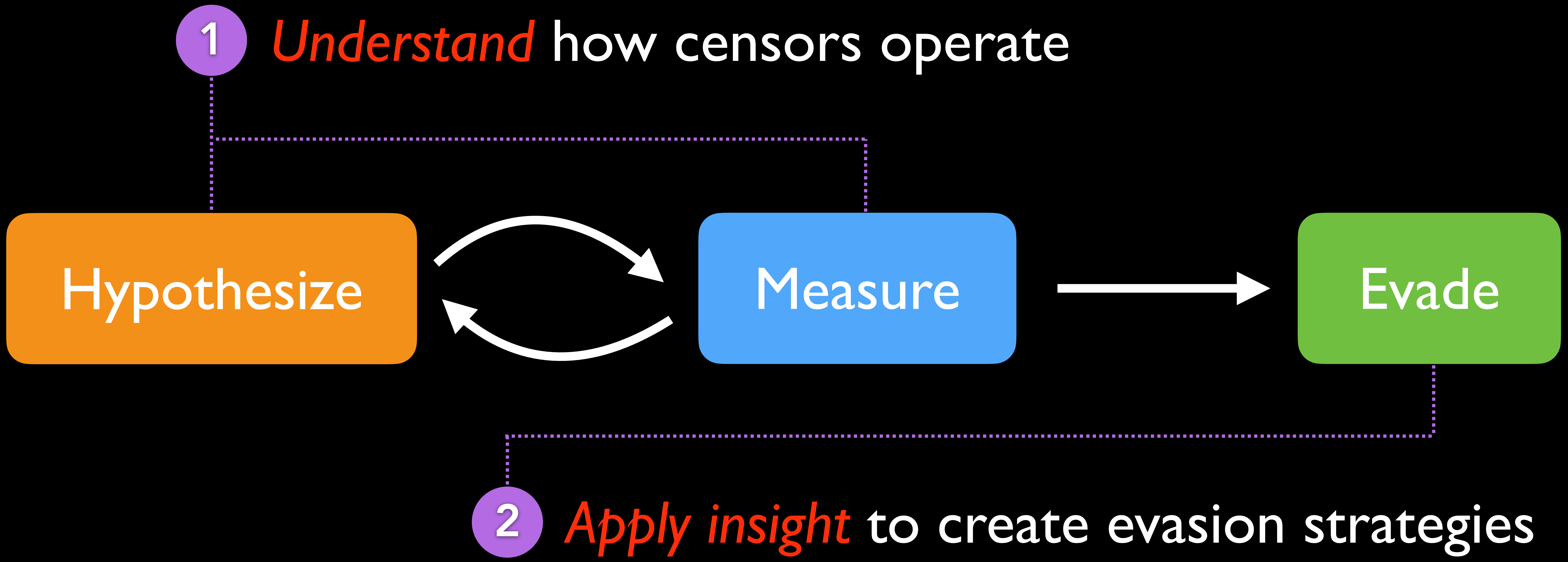

1
Understand how censors operate
Hypothesize
Measure
Evade
2
Apply insight to create evasion strategies

Censorship evasion research


1
Understand how censors operate
Hypothesize
Measure
Evade
2
Apply insight to create evasion strategies

Largely manual efforts give censors the advantage

Censorship evasion research



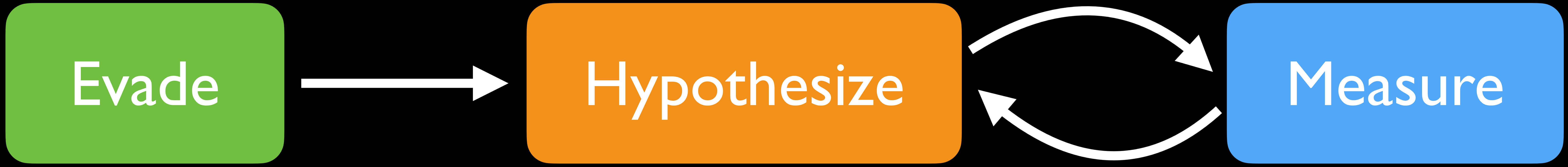
1 *Understand* how censors operate

2 *Apply insight* to create evasion strategies

Largely manual efforts give censors the advantage

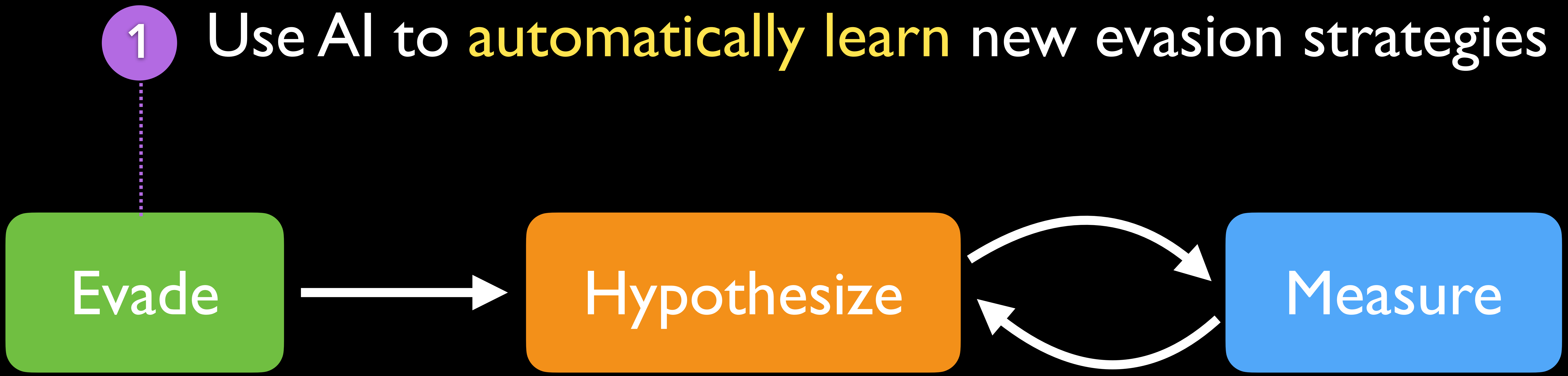
Our work gives evasion the advantage

AI-assisted censorship evasion research



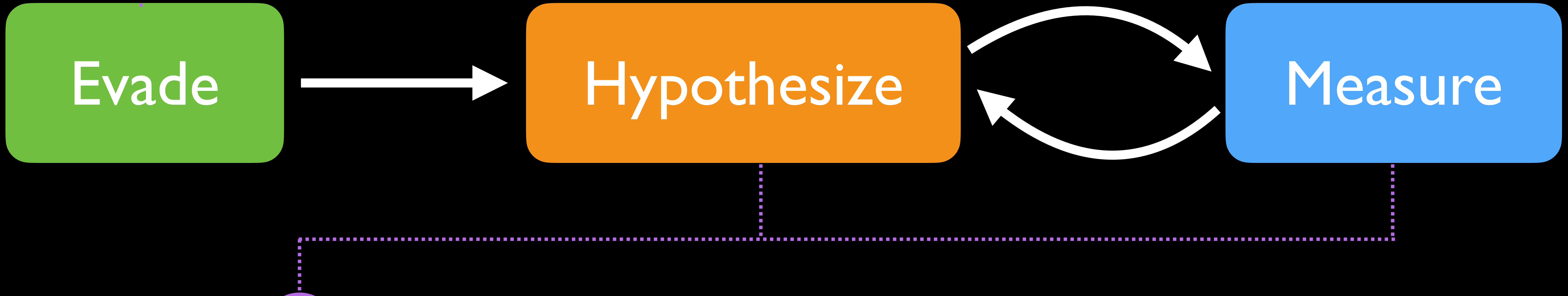
AI-assisted censorship evasion research

1 Use AI to automatically learn new evasion strategies



AI-assisted censorship evasion research

1 Use AI to automatically learn new evasion strategies

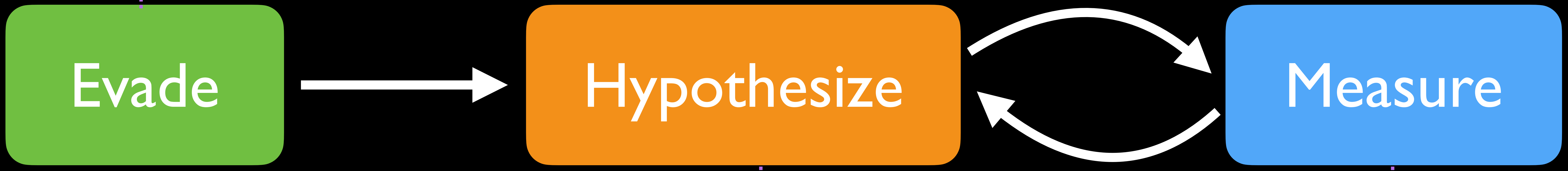


2 Use the strategies the AI finds to understand how the censor works

Geneva

Genetic Evasion

1 Use AI to automatically learn new evasion strategies



2 Use the strategies the AI finds to understand how the censor works

Geneva

Genetic Evasion

Building Blocks

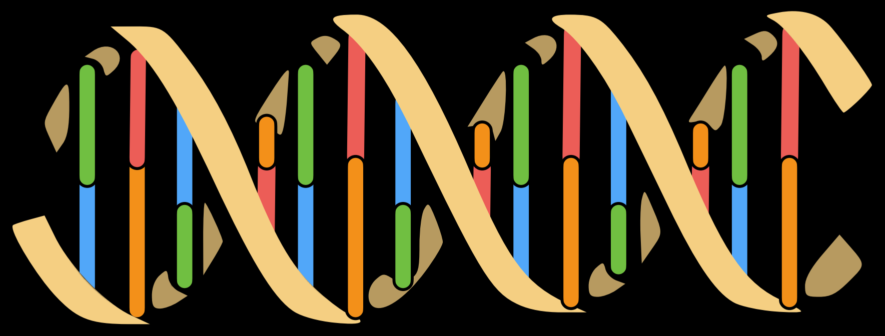
A

T

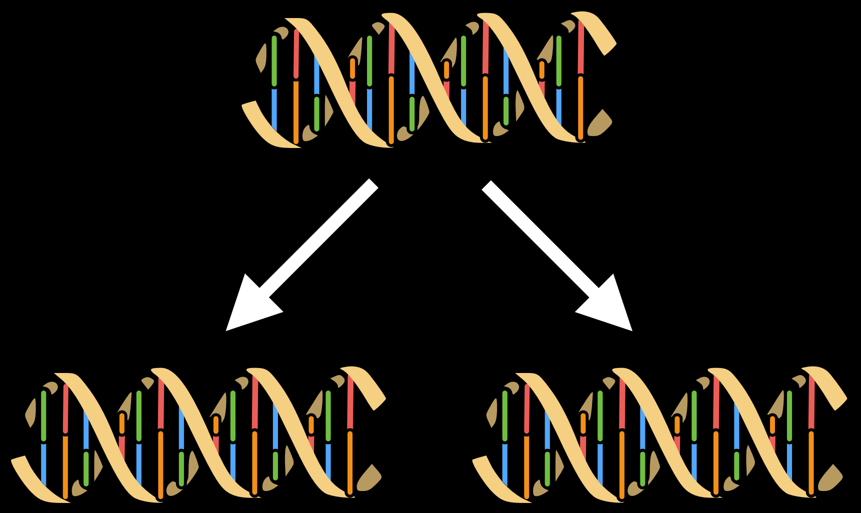
C

G

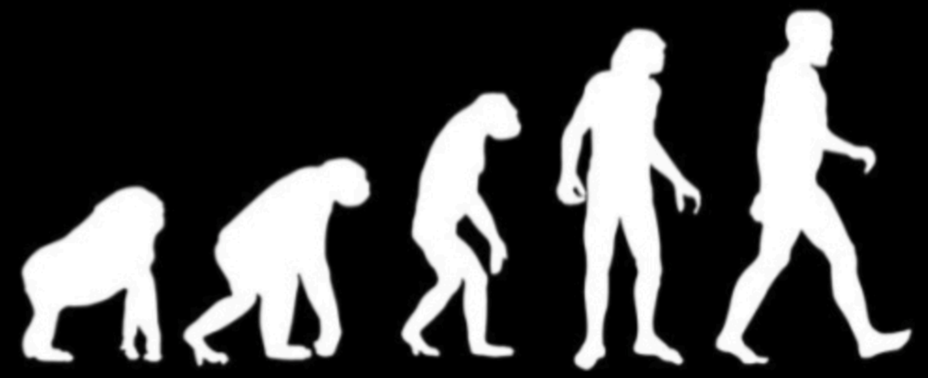
Composition



Mutation



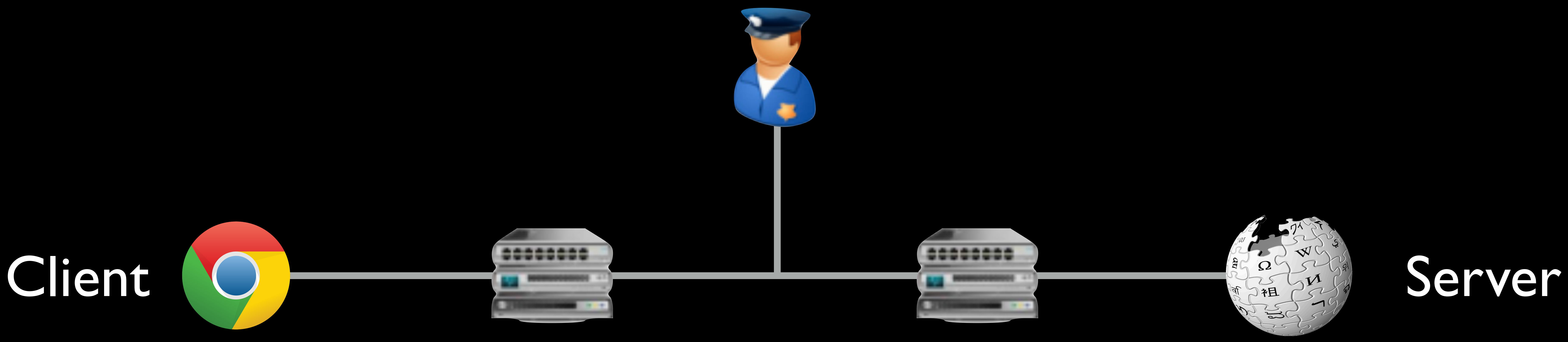
Fitness



Geneva

Genetic Evasion

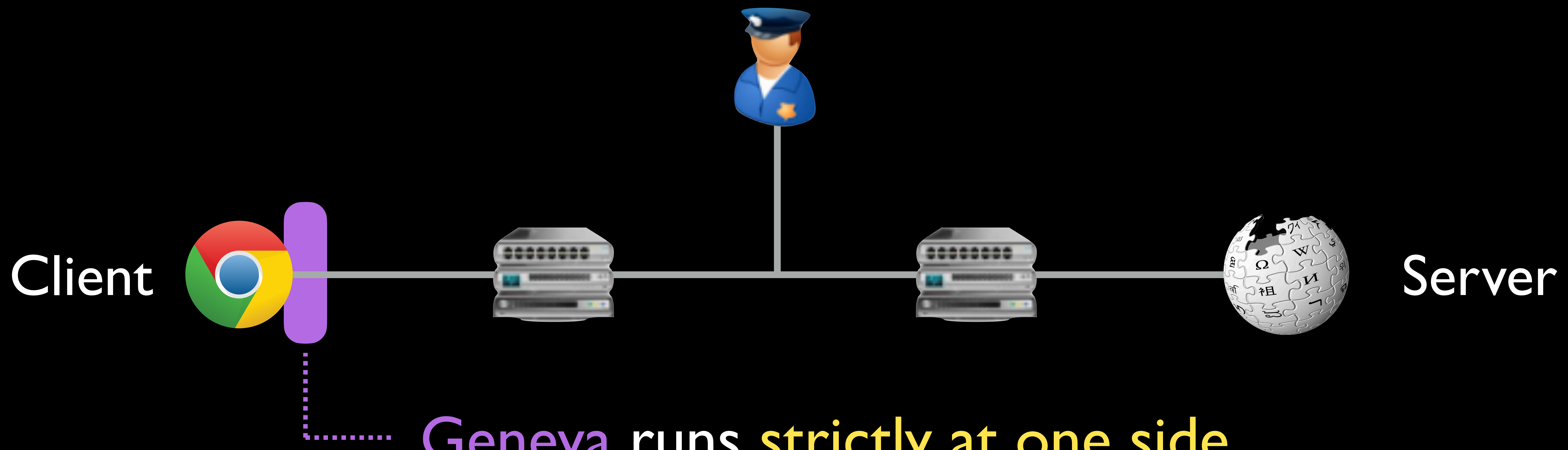
Building Blocks



Geneva

Genetic Evasion

Building Blocks



Geneva runs strictly at one side

Manipulates packets to and from the client

Geneva

Genetic Evasion

Building Blocks

Manipulates packets to and from the client

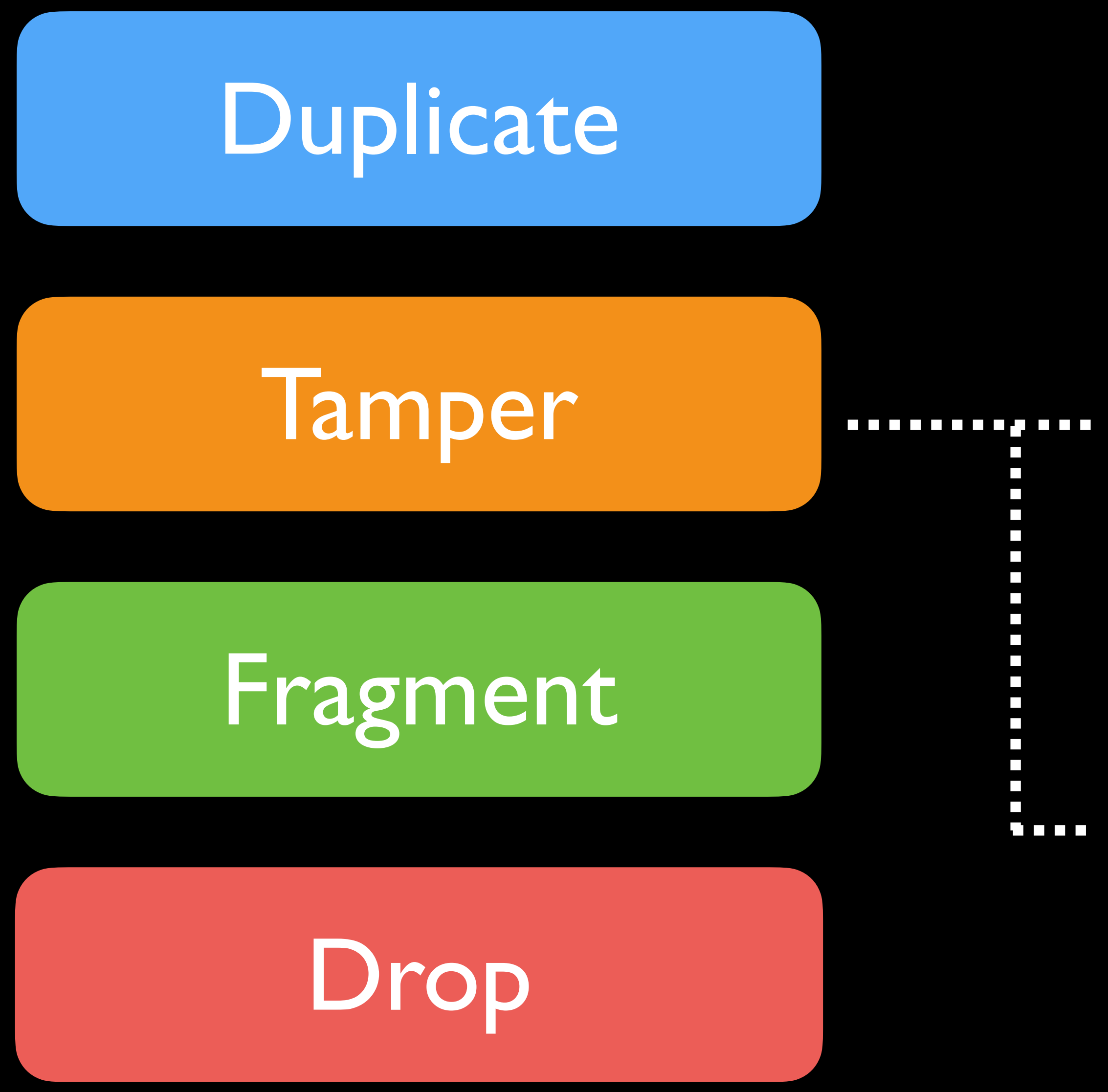
- Duplicate
- Tamper
- Fragment
- Drop

Geneva

Genetic Evasion

Building Blocks

Manipulates packets to and from the client



Duplicate

Tamper

Fragment

Drop

Alter or corrupt
any TCP/IP header field

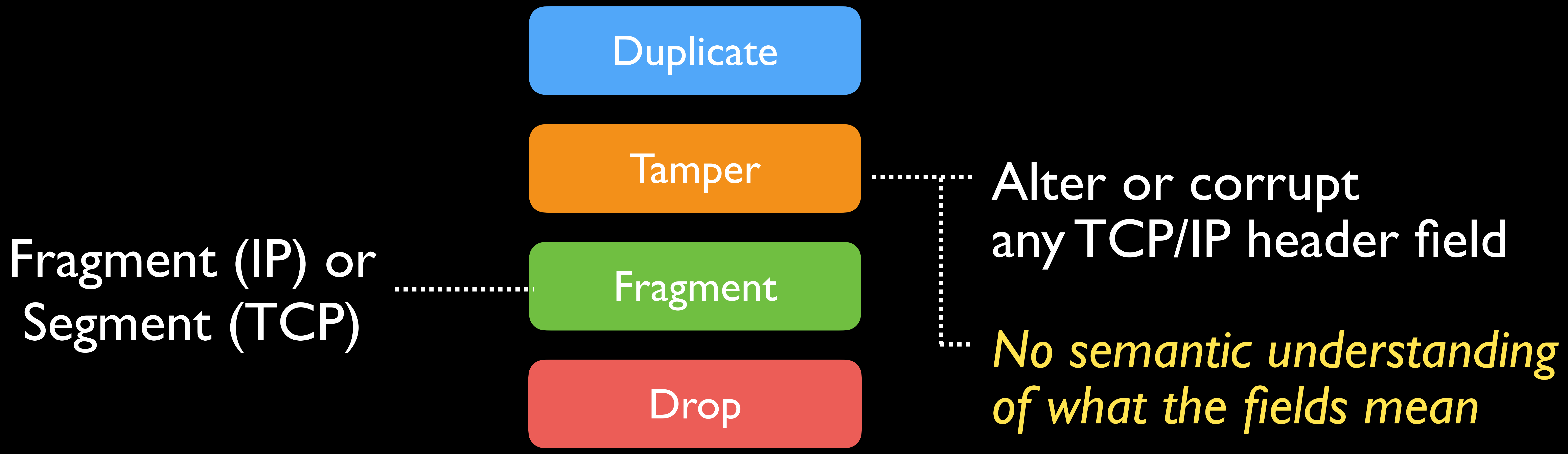
*No semantic understanding
of what the fields mean*

Geneva

Genetic Evasion

Building Blocks

Manipulates packets to and from the client



Duplicate

Tamper

Fragment

Drop

Fragment (IP) or Segment (TCP)

Alter or corrupt any TCP/IP header field

No semantic understanding of what the fields mean

Geneva

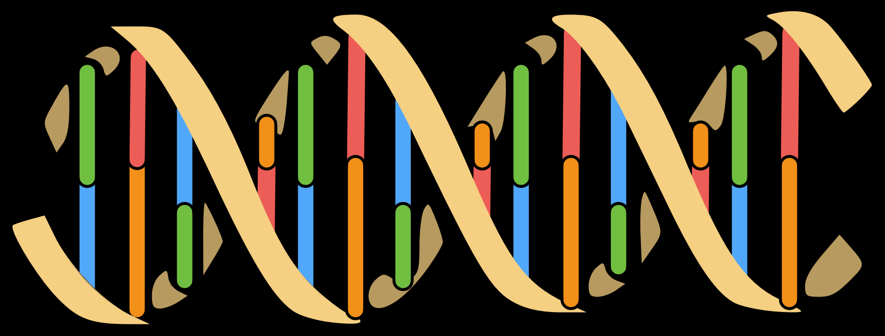
Genetic Evasion

Building Blocks

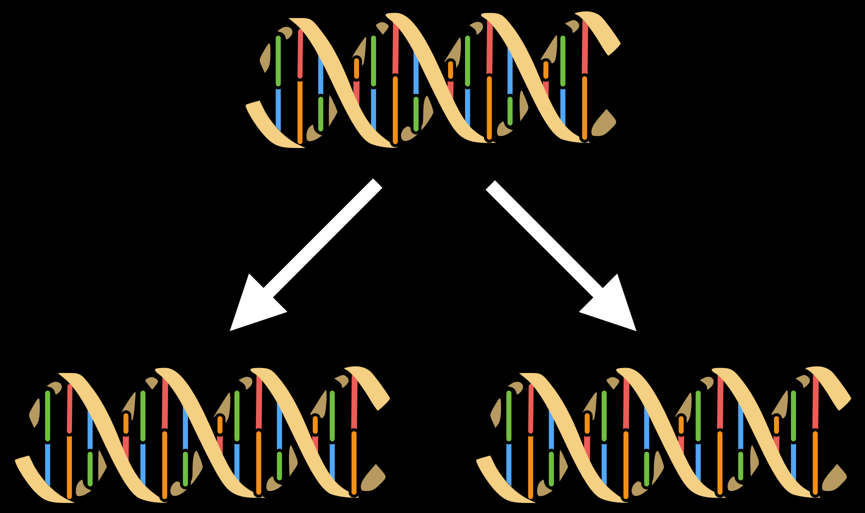
Actions manipulate individual packets

- Duplicate
- Tamper
- Fragment
- Drop

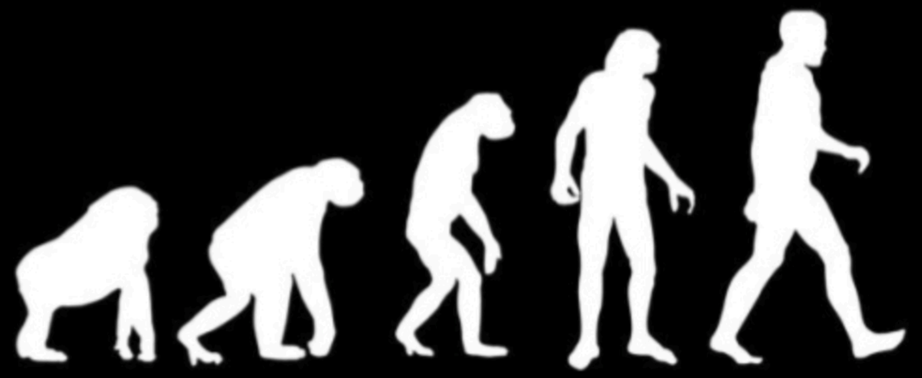
Composition



Mutation



Fitness



Geneva

Genetic Evasion

Building Blocks

Actions manipulate individual packets

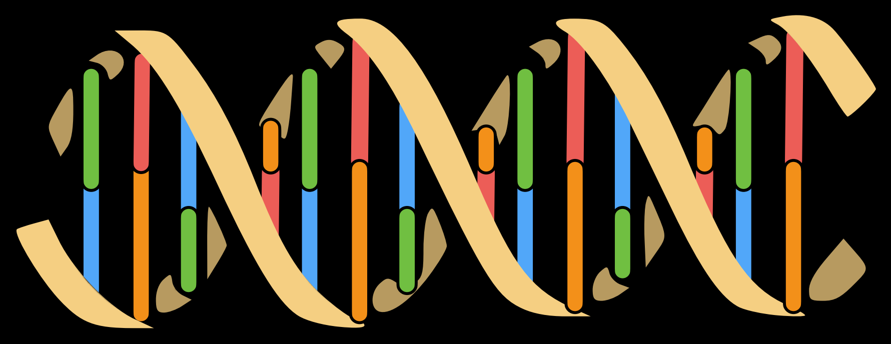
Duplicate

Tamper

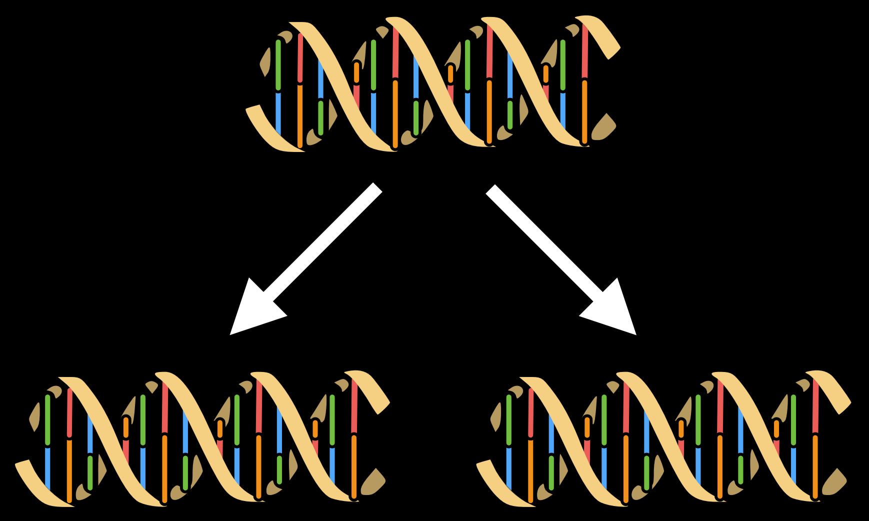
Fragment

Drop

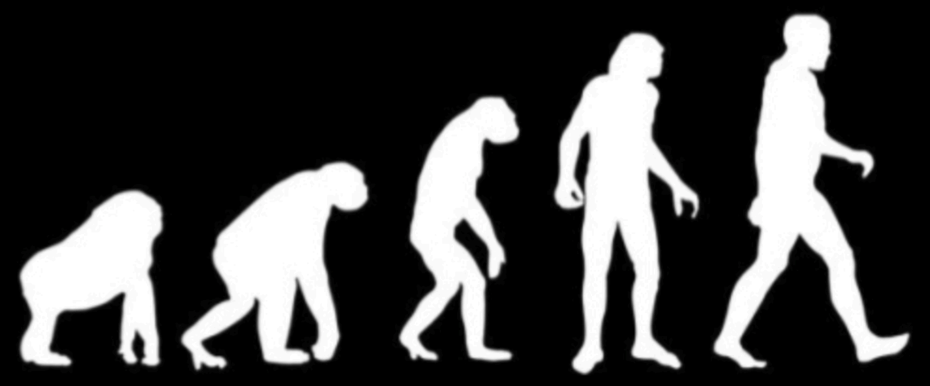
Composition



Mutation



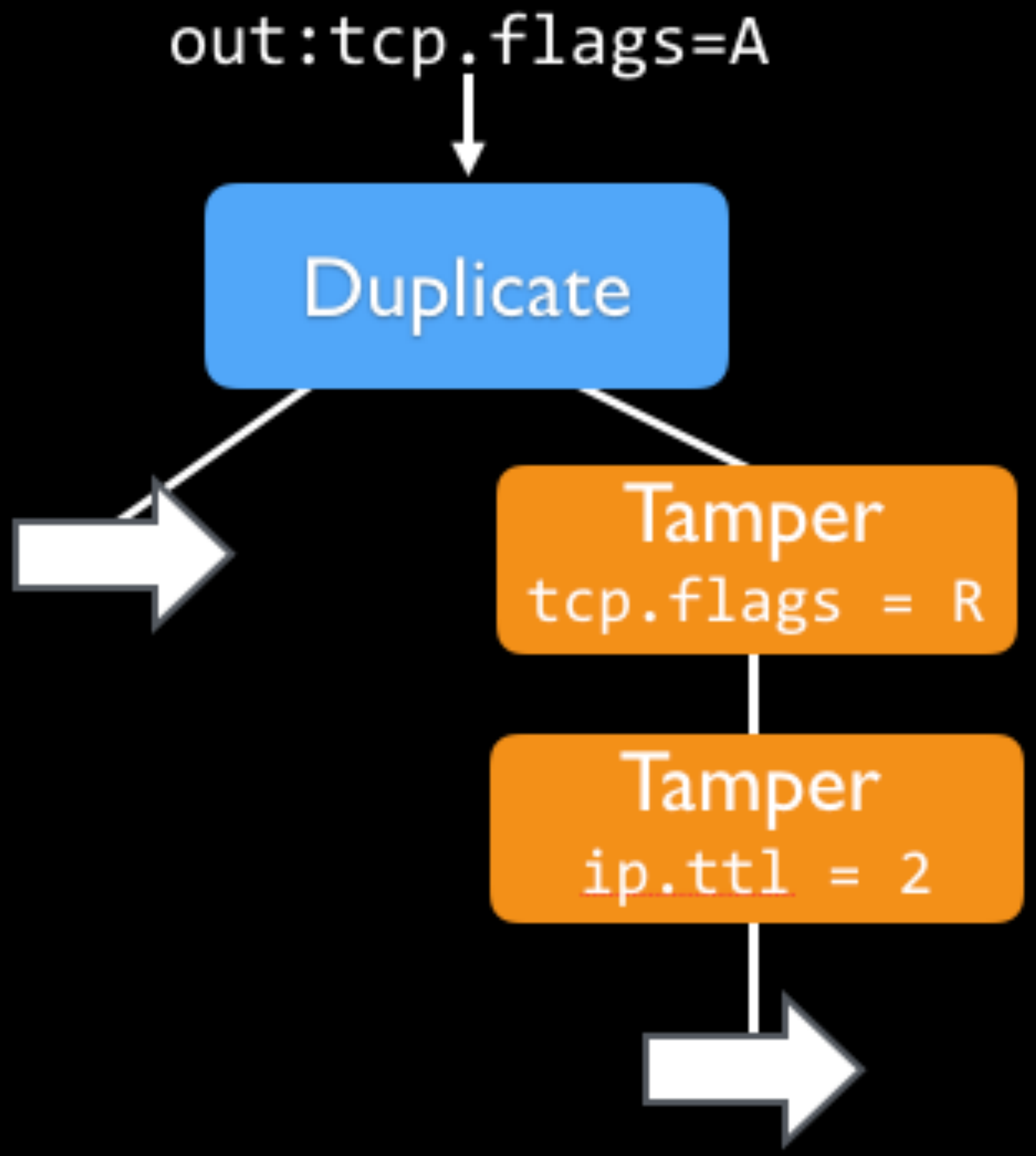
Fitness



Geneva

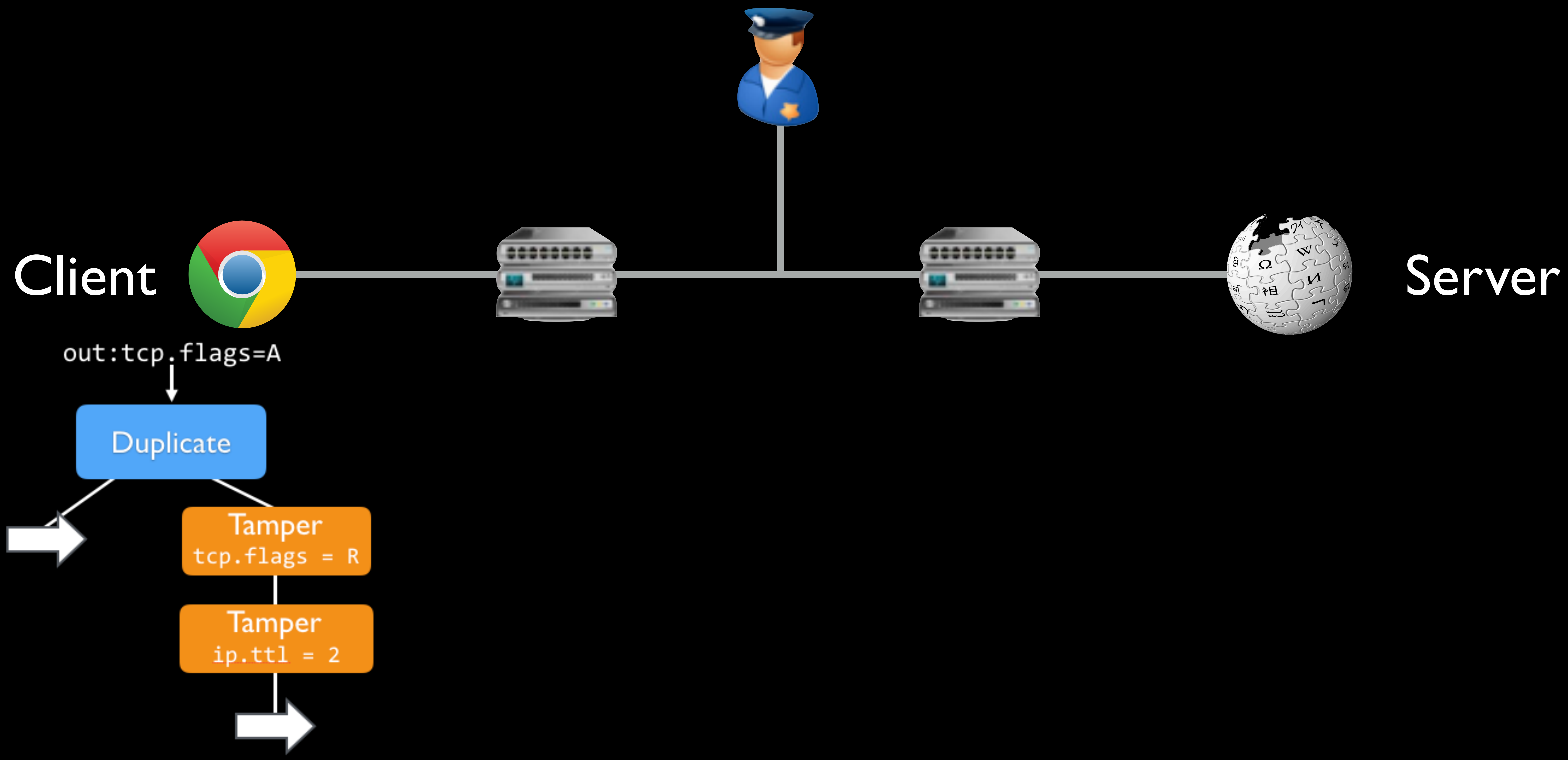
Genetic Evasion

Composition

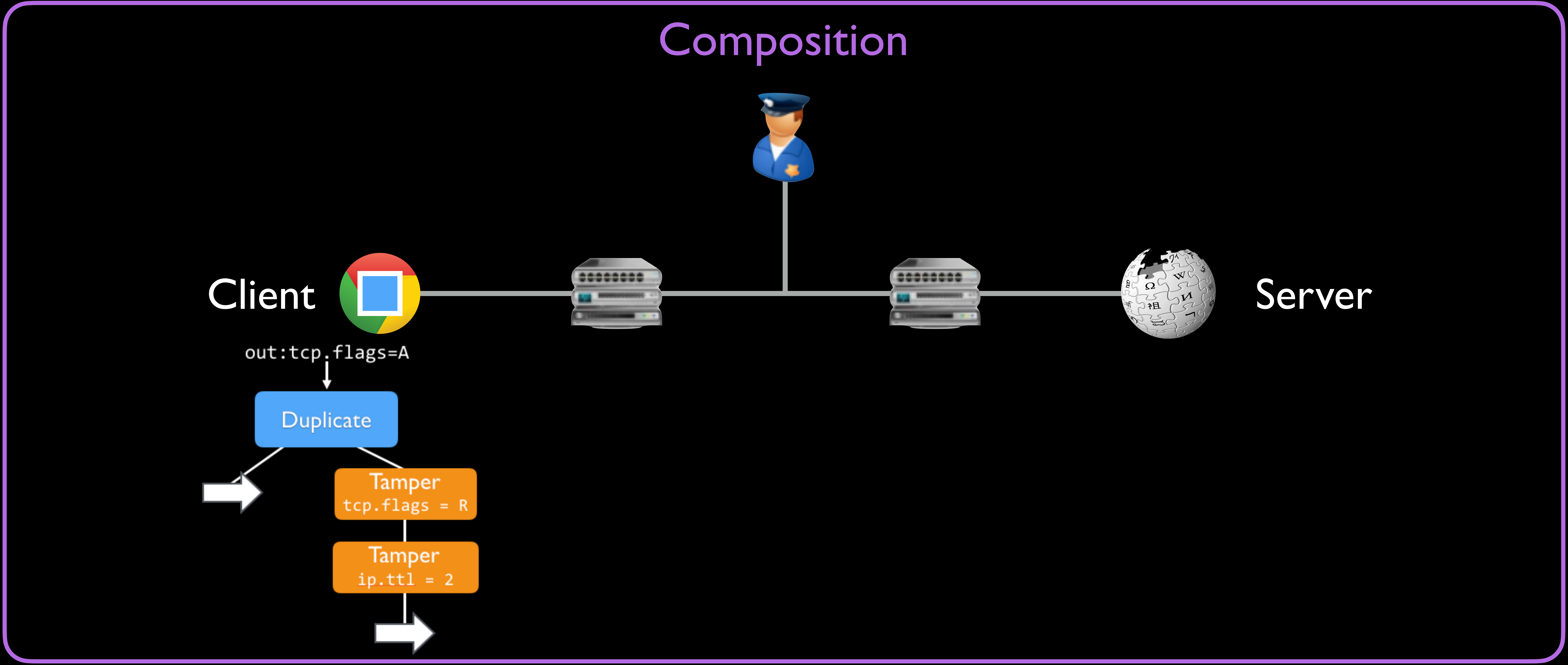


Running a Strategy

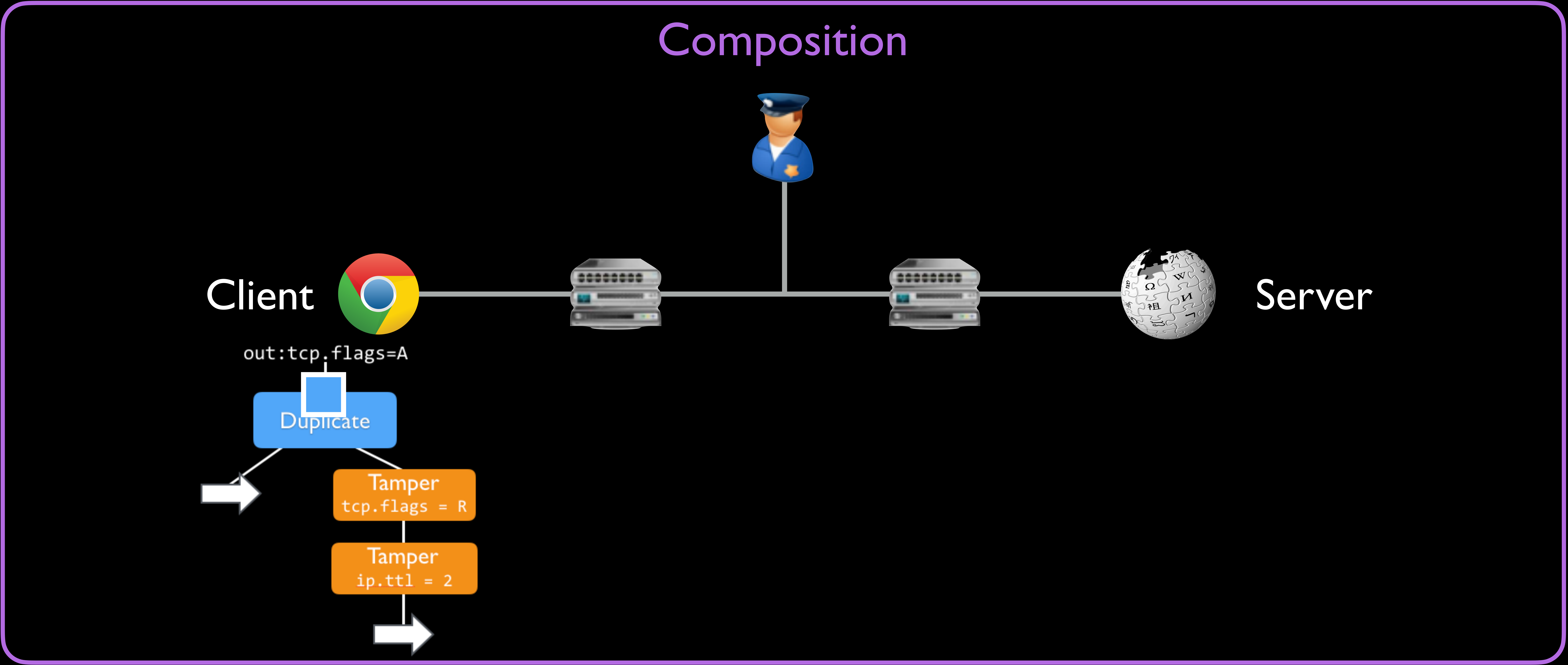
Composition



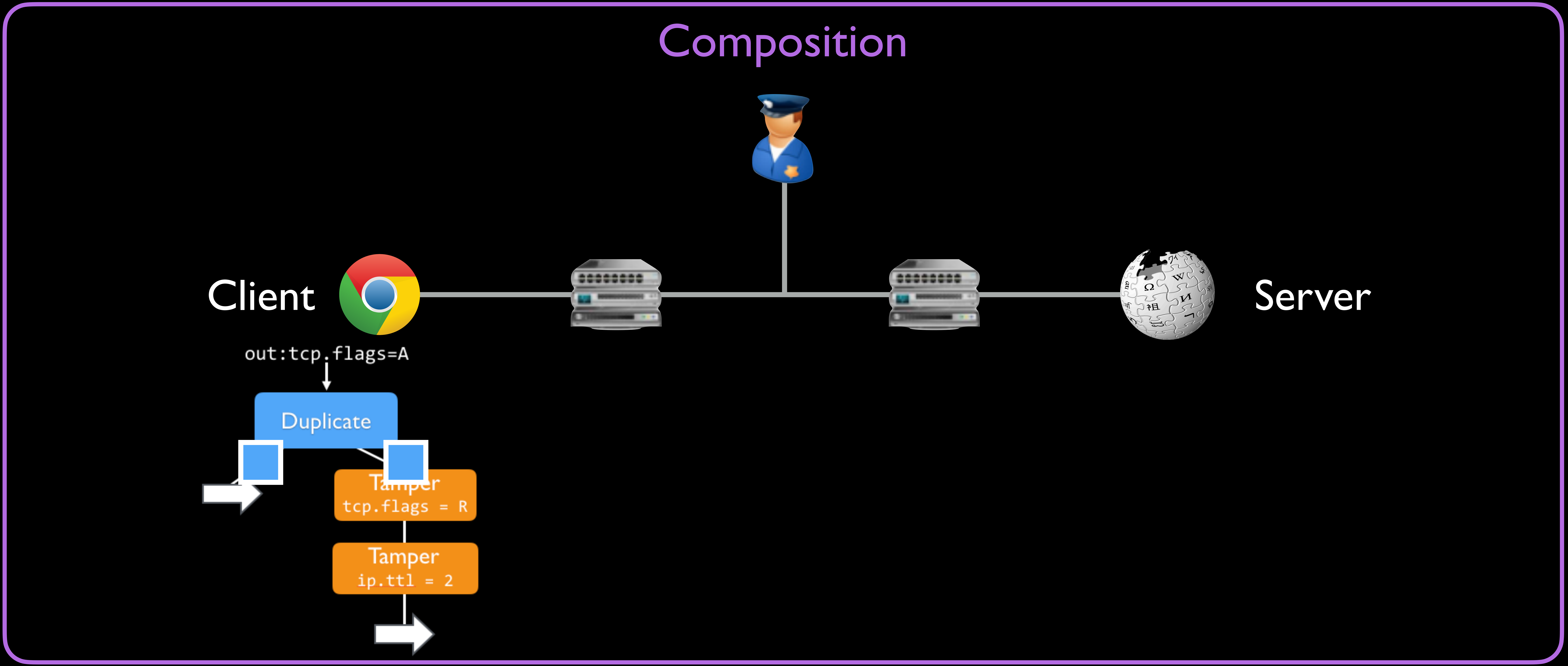
Running a Strategy



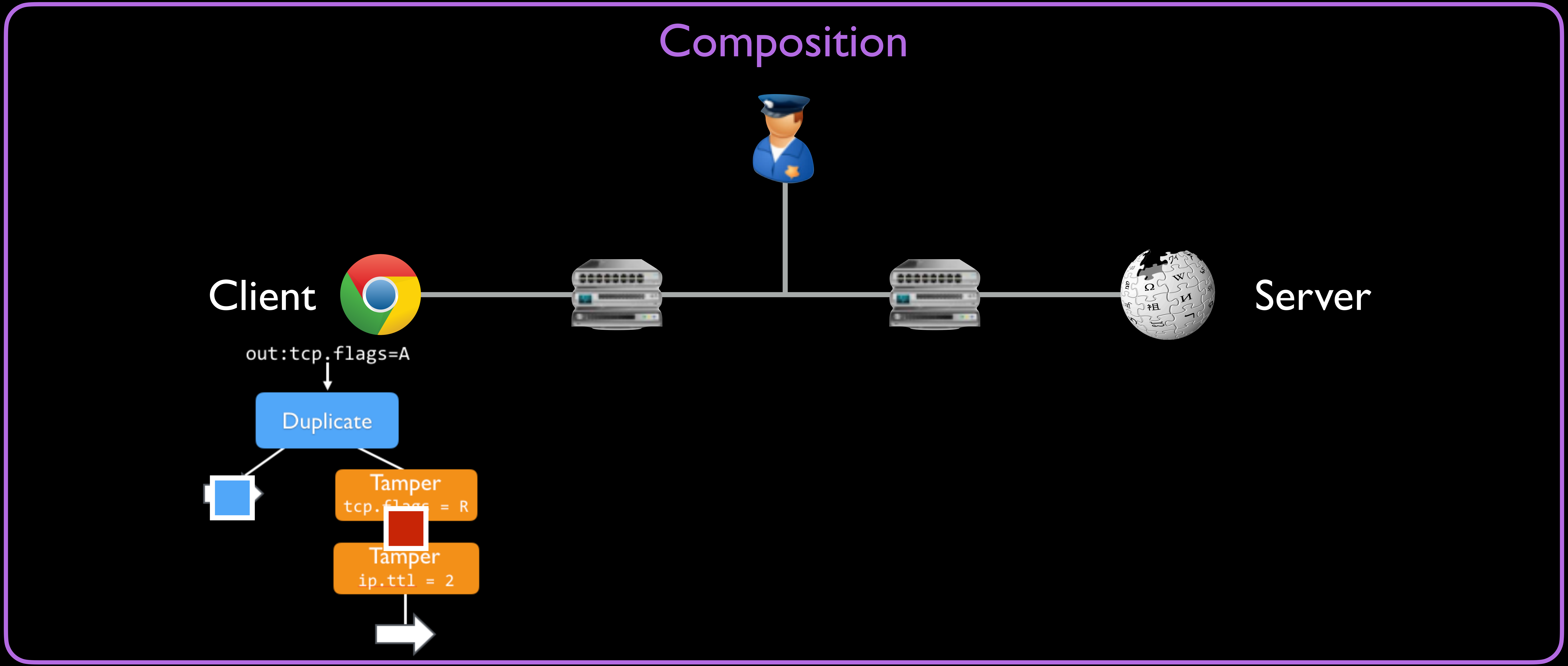
Running a Strategy



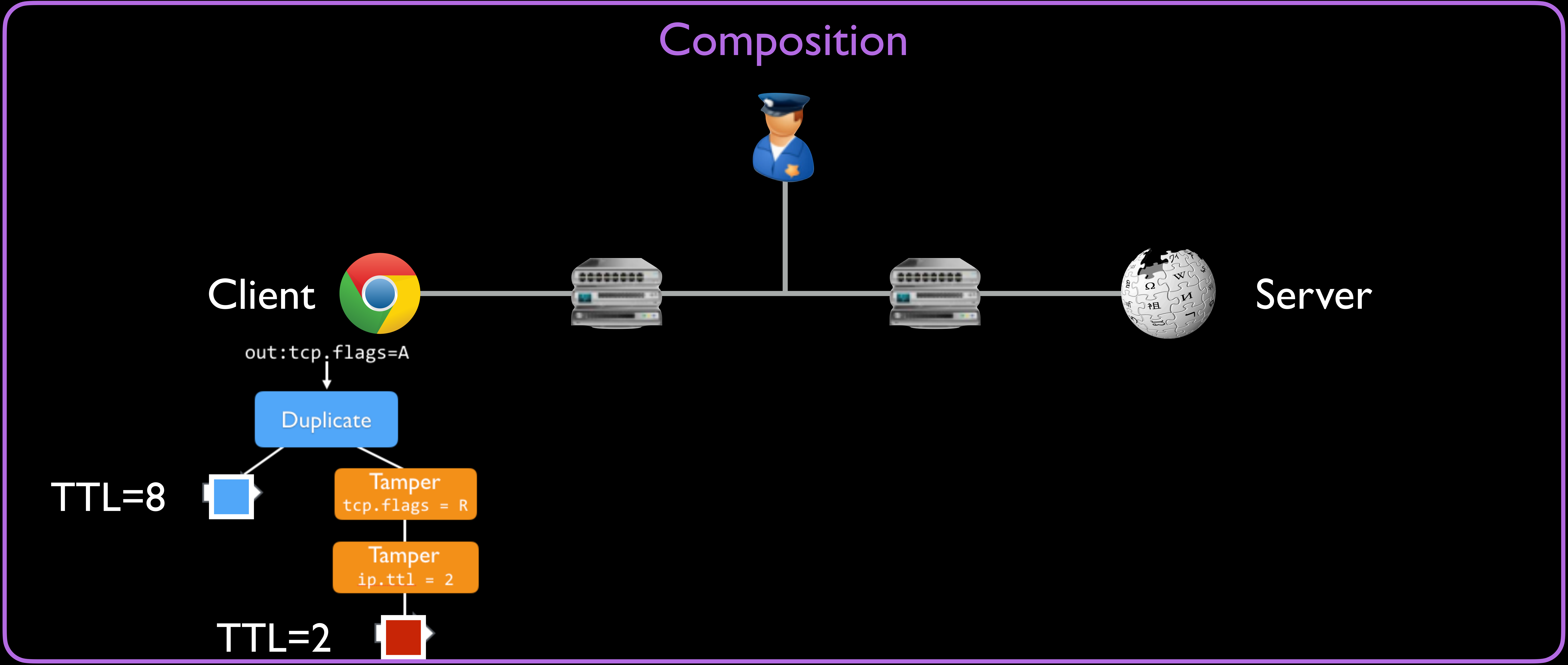
Running a Strategy



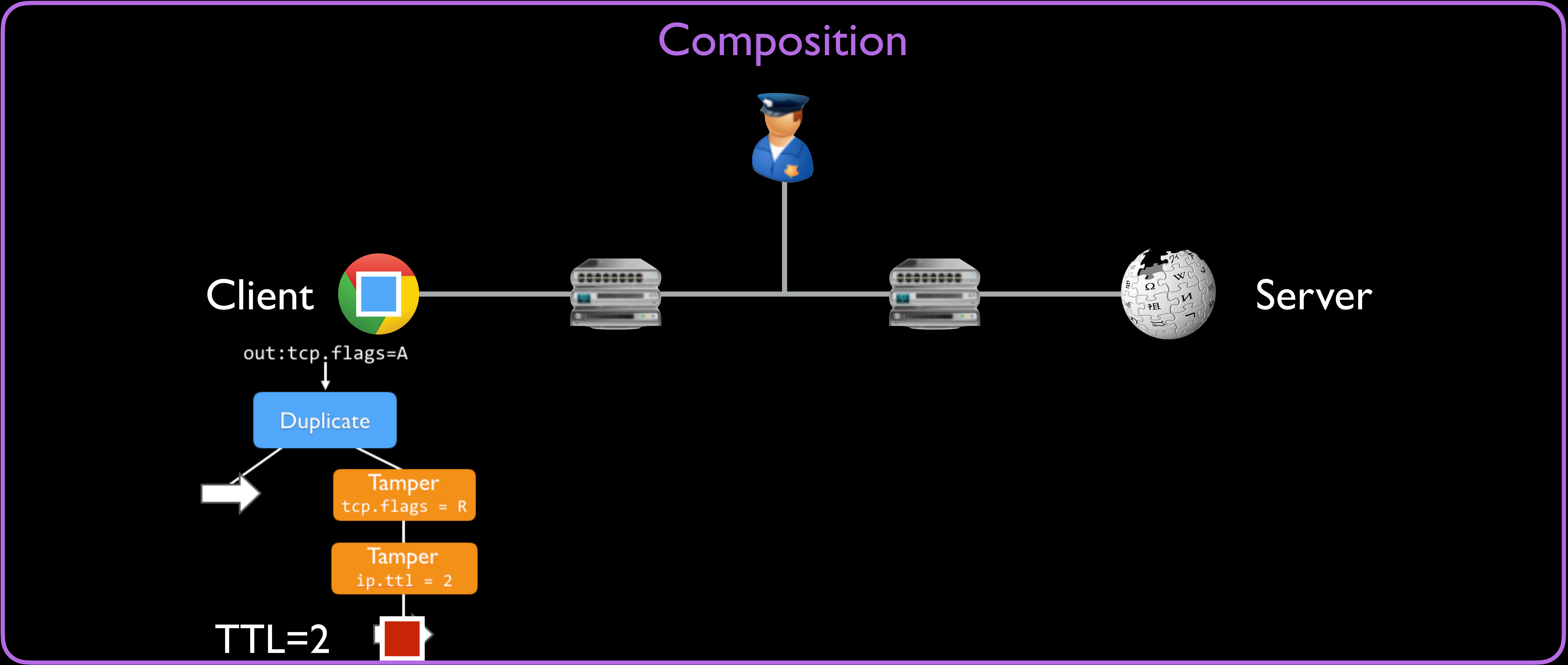
Running a Strategy



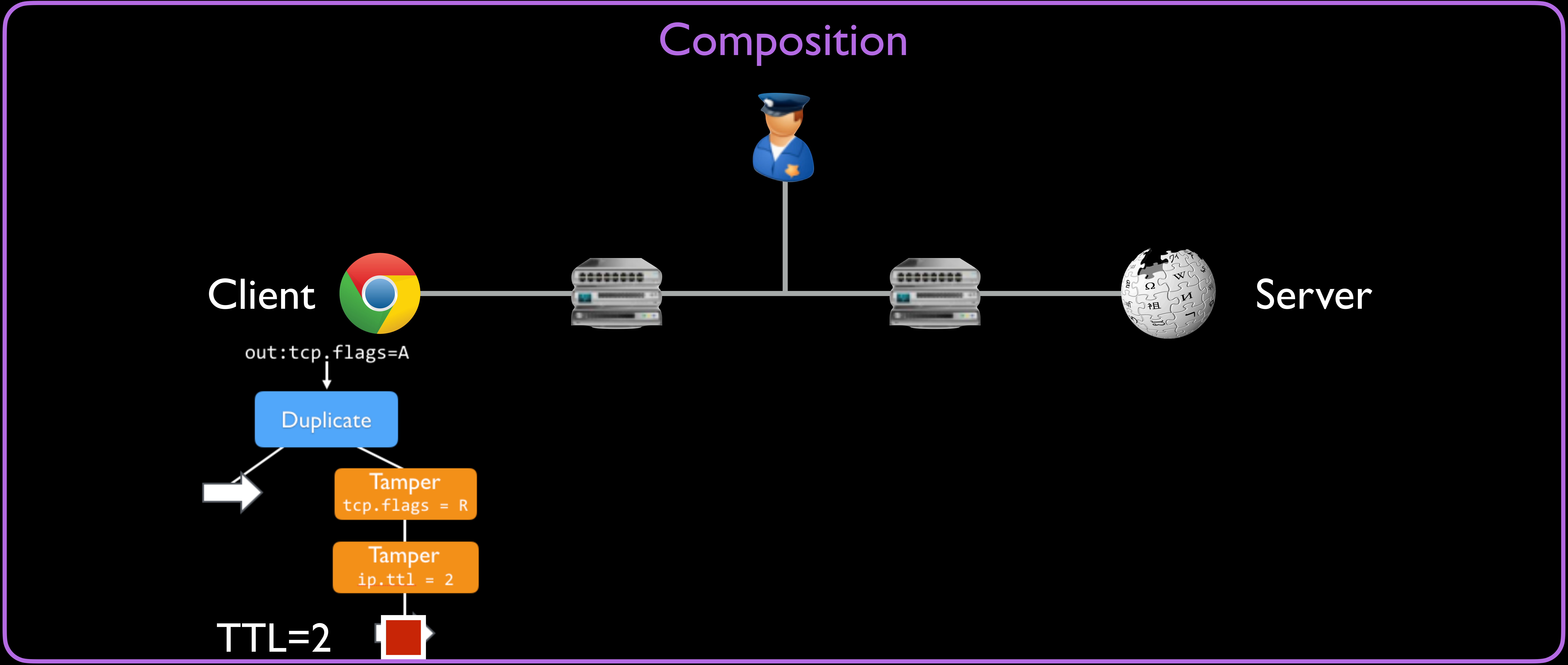
Running a Strategy



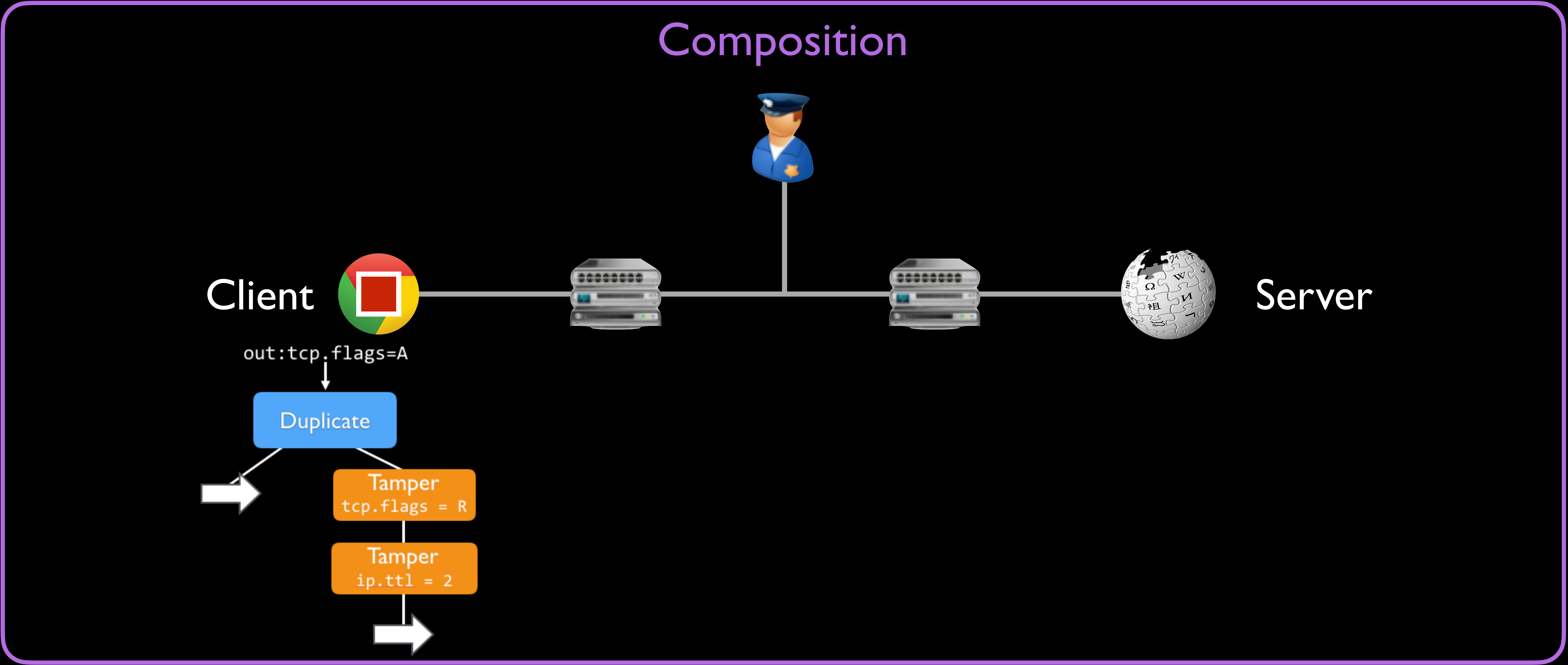
Running a Strategy



Running a Strategy



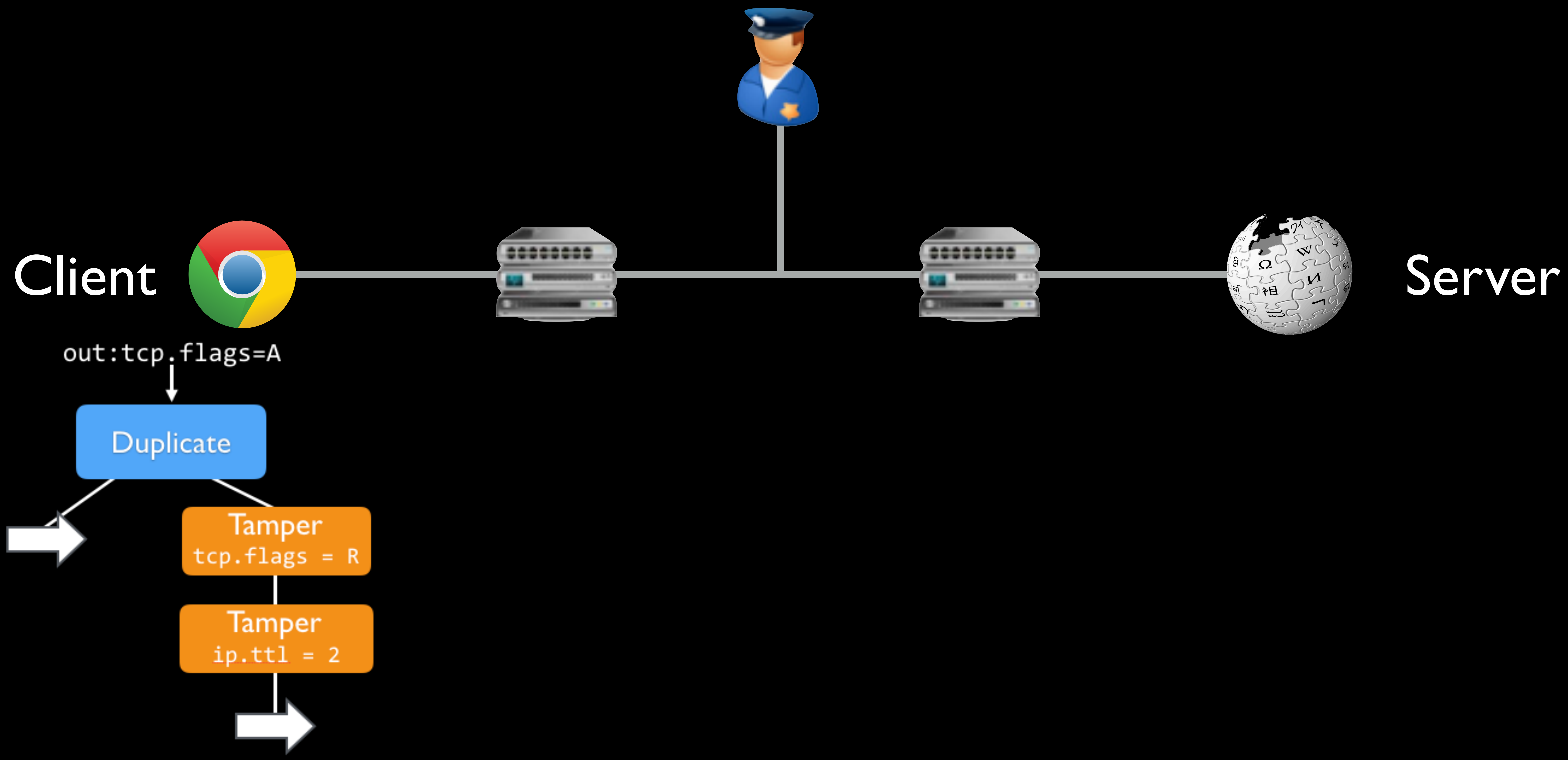
Running a Strategy



Composition

Running a Strategy

Composition



Geneva

Genetic Evasion

Building Blocks

Actions manipulate individual packets

Duplicate

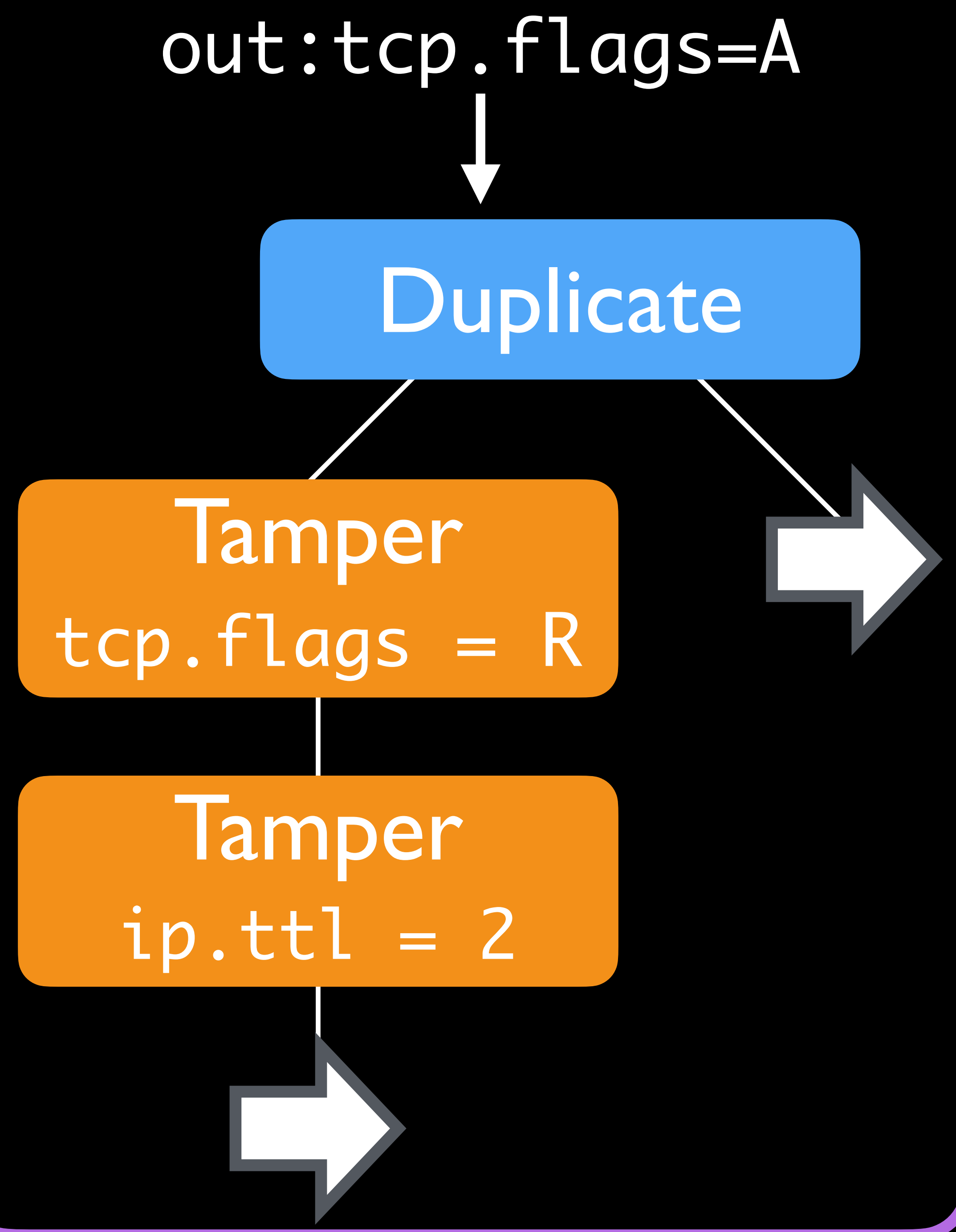
Tamper

Fragment

Drop

Composition

Actions compose to form trees



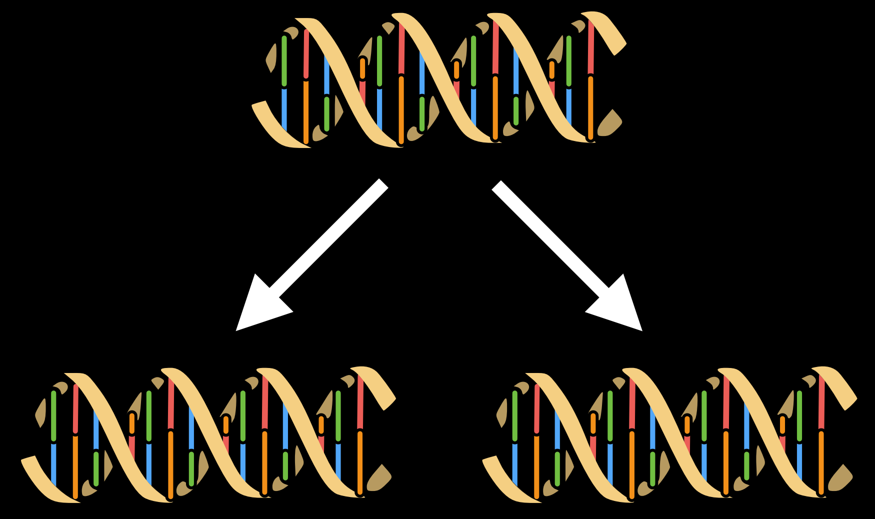
out:tcp.flags=A

Duplicate

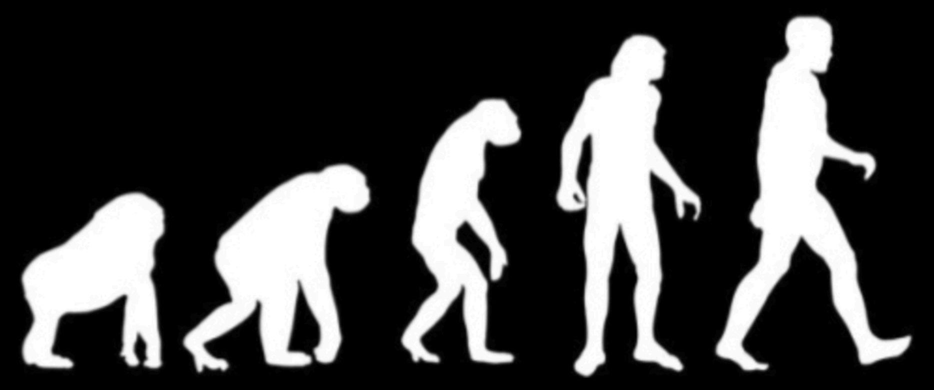
Tamper
tcp.flags = R

Tamper
ip.ttl = 2

Mutation



Fitness



Geneva

Genetic Evasion

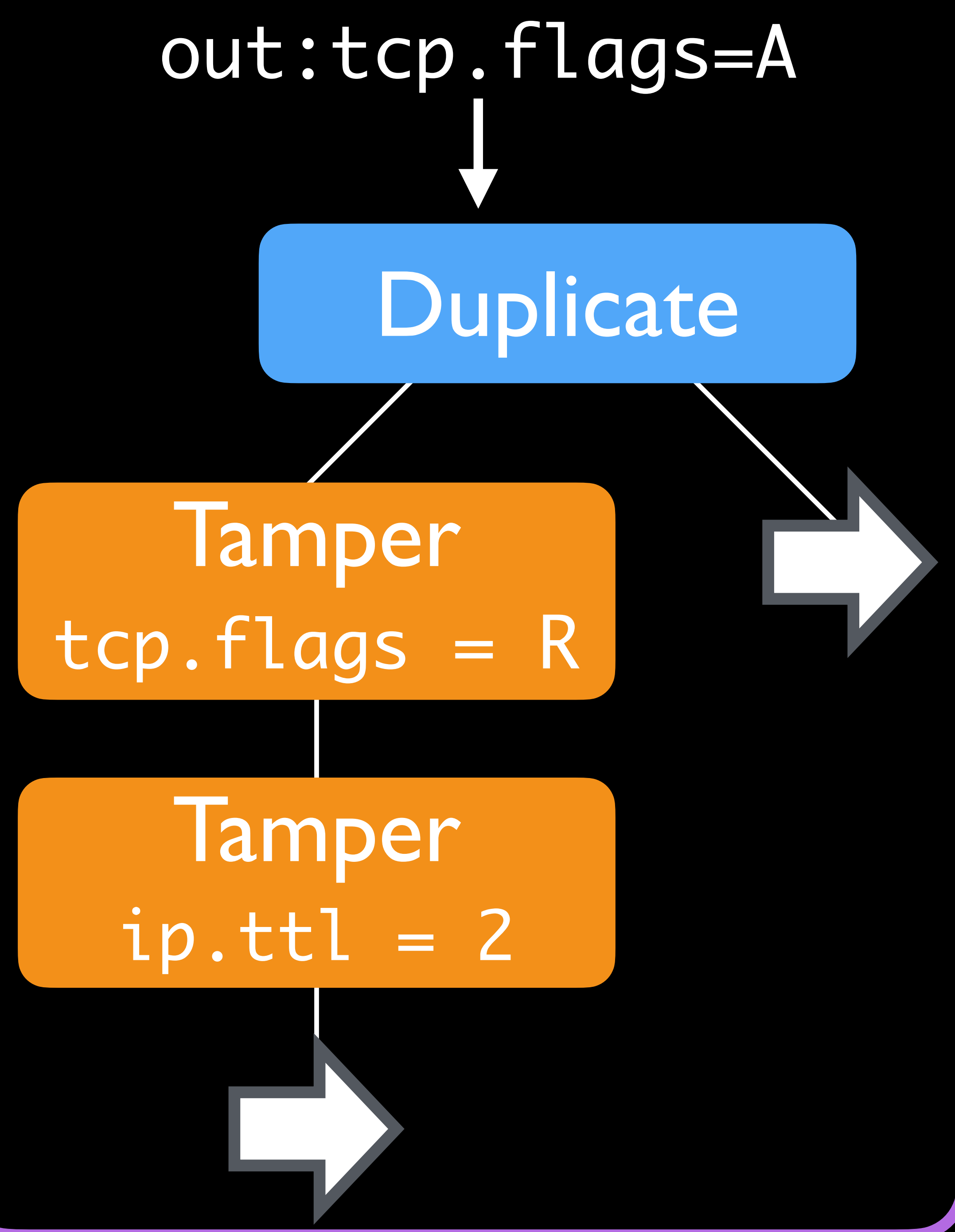
Building Blocks

Actions manipulate individual packets

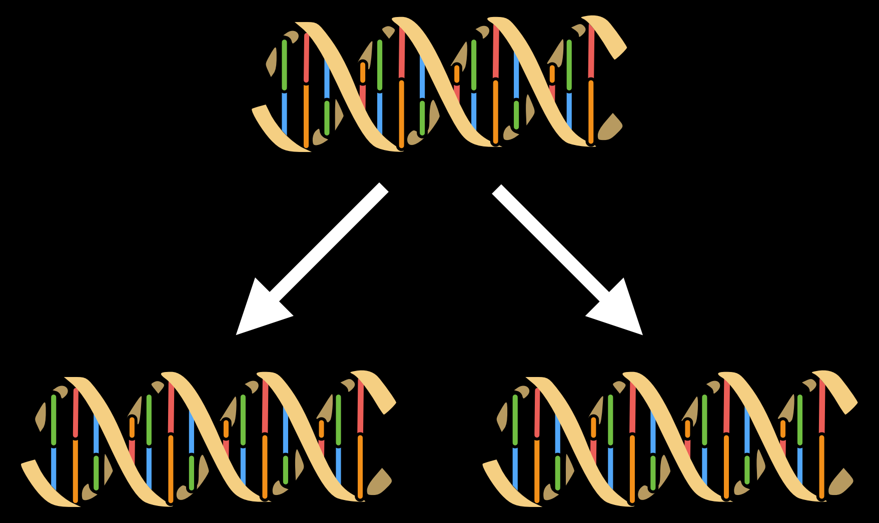
- Duplicate
- Tamper
- Fragment
- Drop

Composition

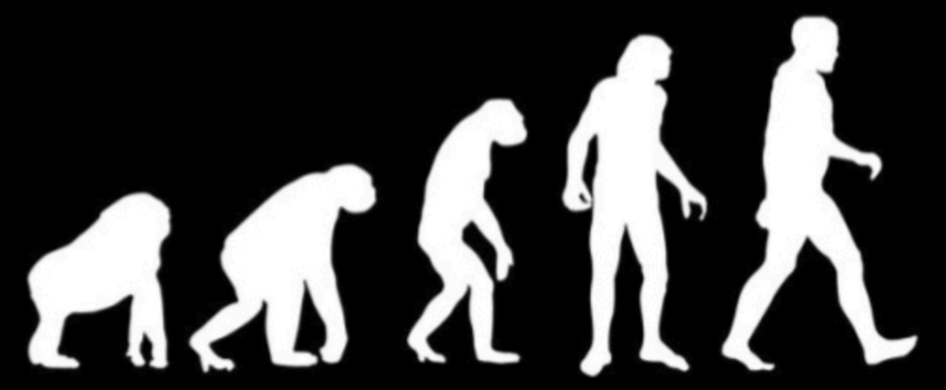
Actions compose to form trees



Mutation



Fitness



Geneva

Genetic Evasion

Building Blocks

Actions manipulate individual packets

Duplicate

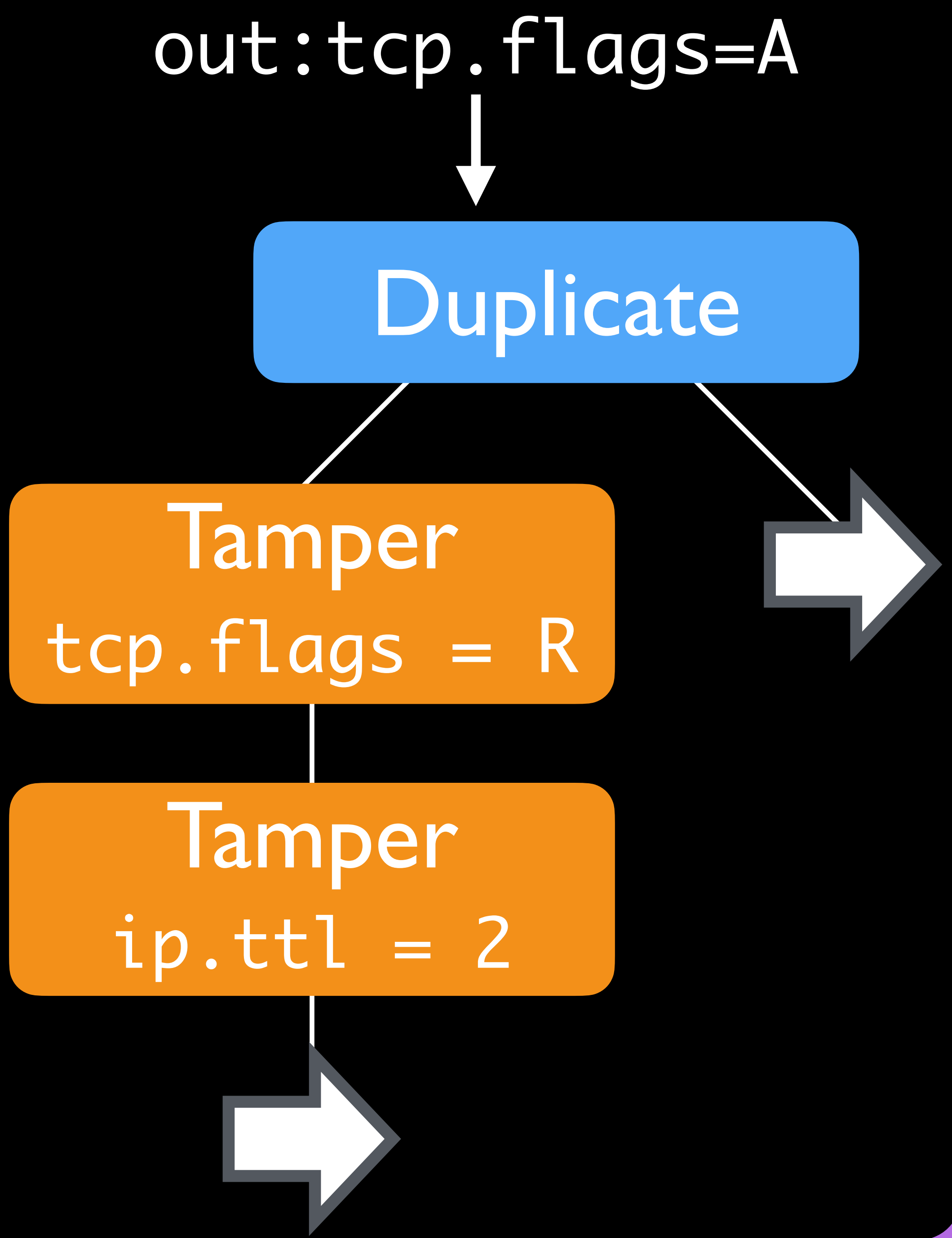
Tamper

Fragment

Drop

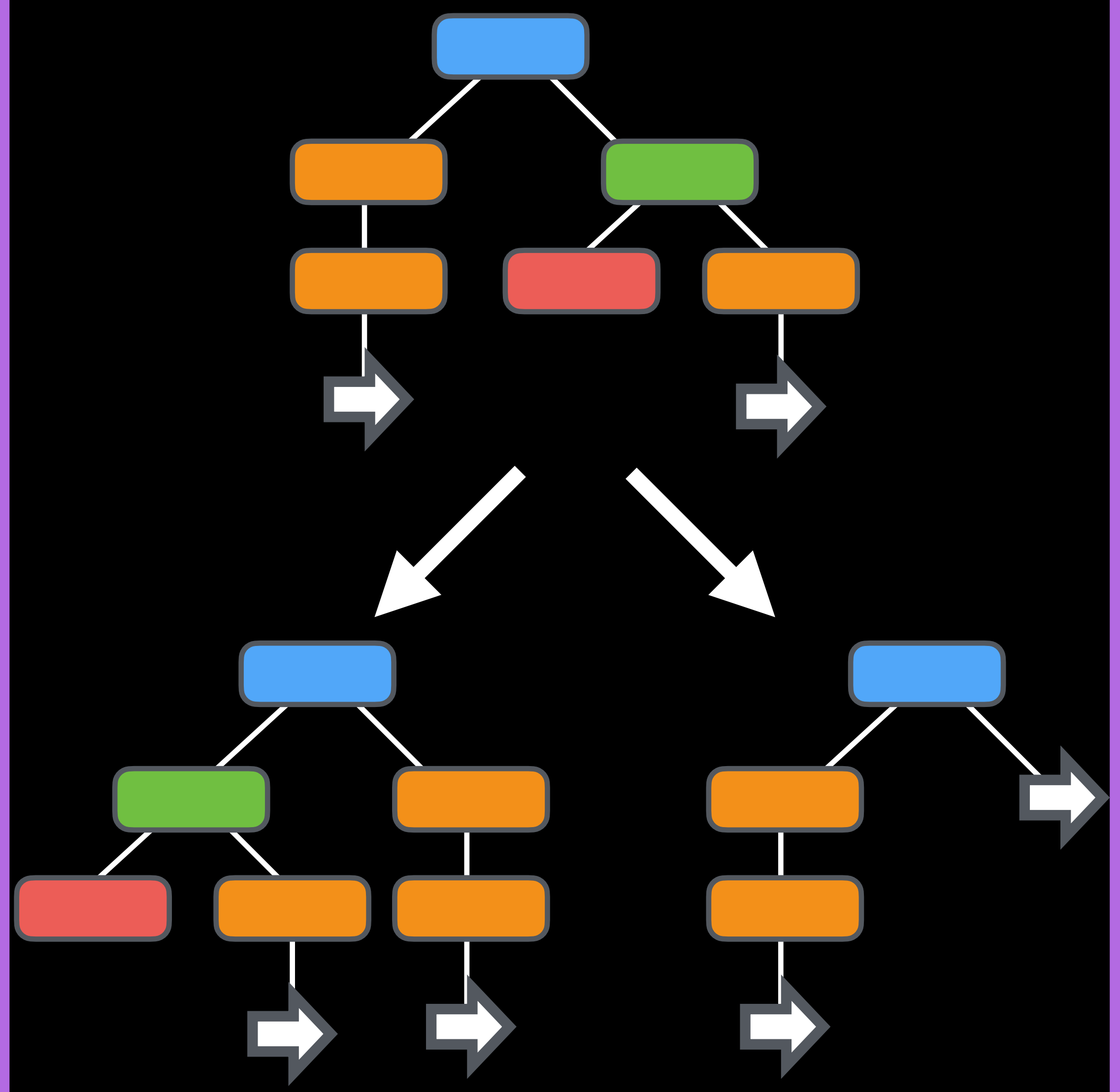
Composition

Actions compose to form trees

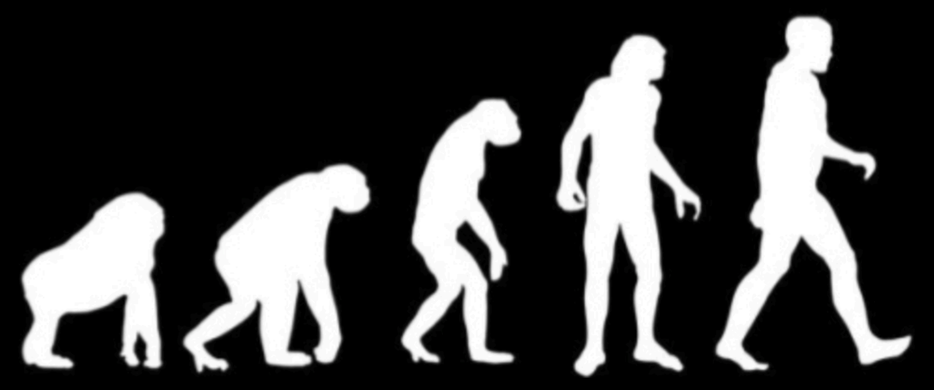


Mutation

Randomly alter types, values, and trees



Fitness



Geneva

Genetic Evasion

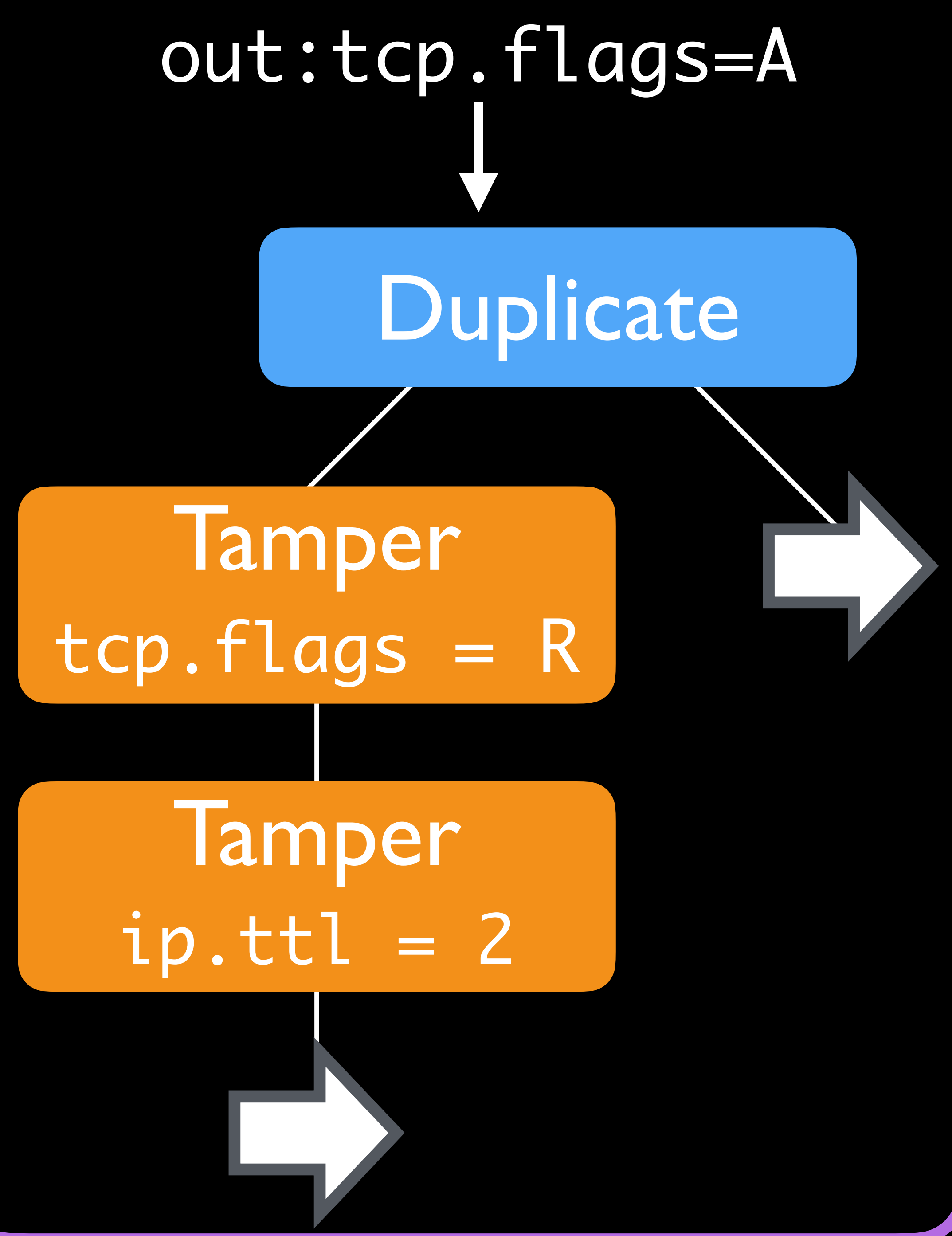
Building Blocks

Actions manipulate individual packets

- Duplicate
- Tamper
- Fragment
- Drop

Composition

Actions compose to form trees



`out:tcp.flags=A`

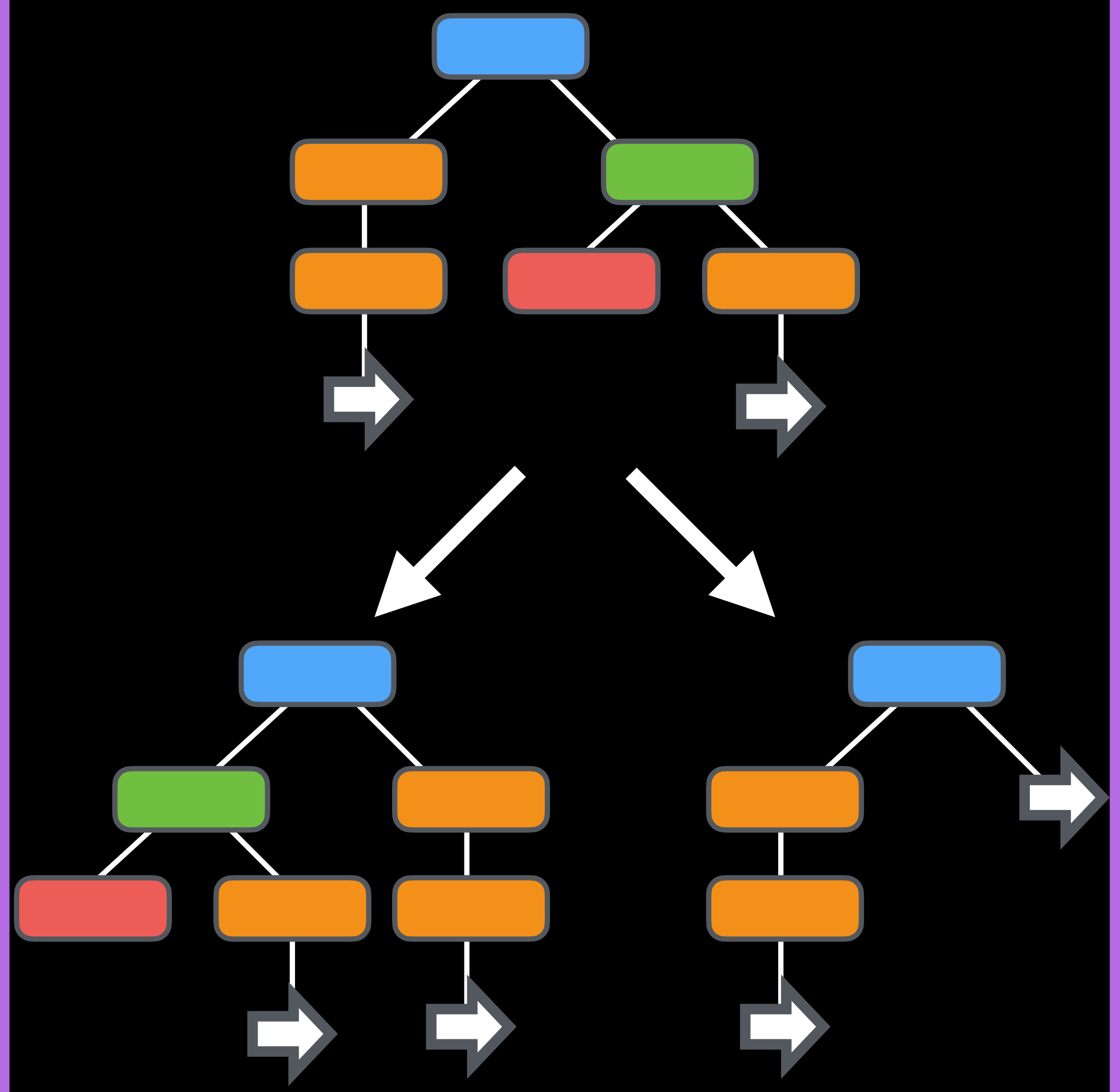
Duplicate

Tamper
`tcp.flags = R`

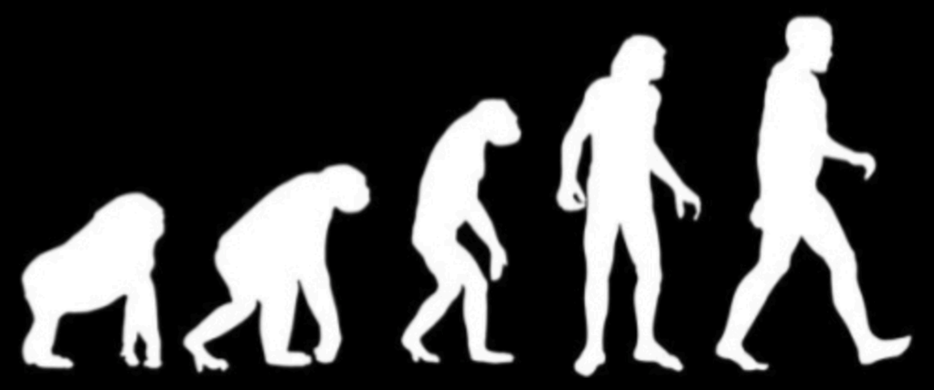
Tamper
`ip.ttl = 2`

Mutation

Randomly alter types, values, and trees



Fitness

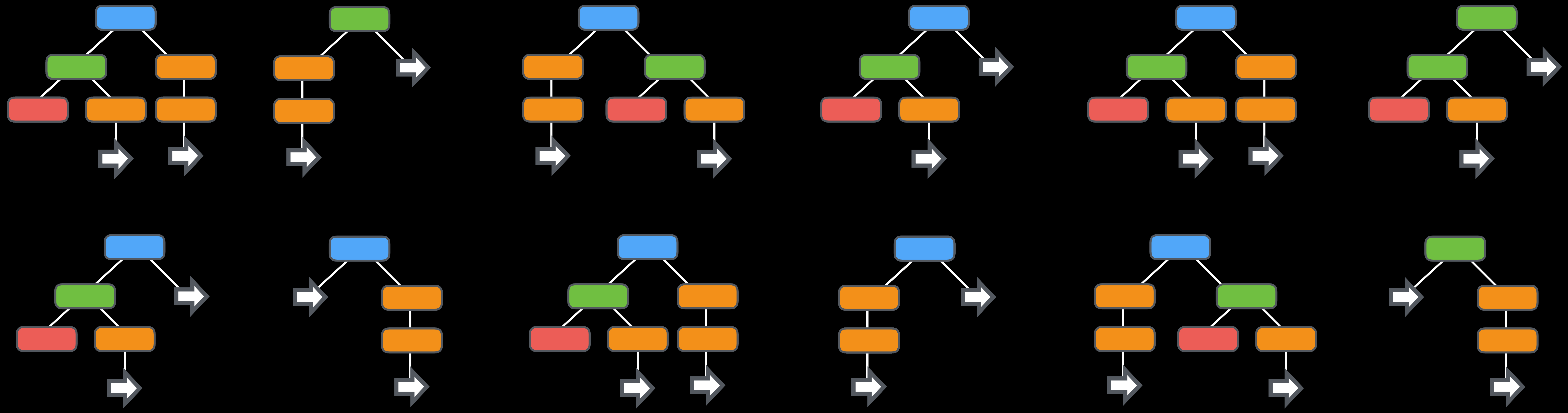


Geneva

Genetic Evasion

Fitness

Which individuals should survive to the next generation?

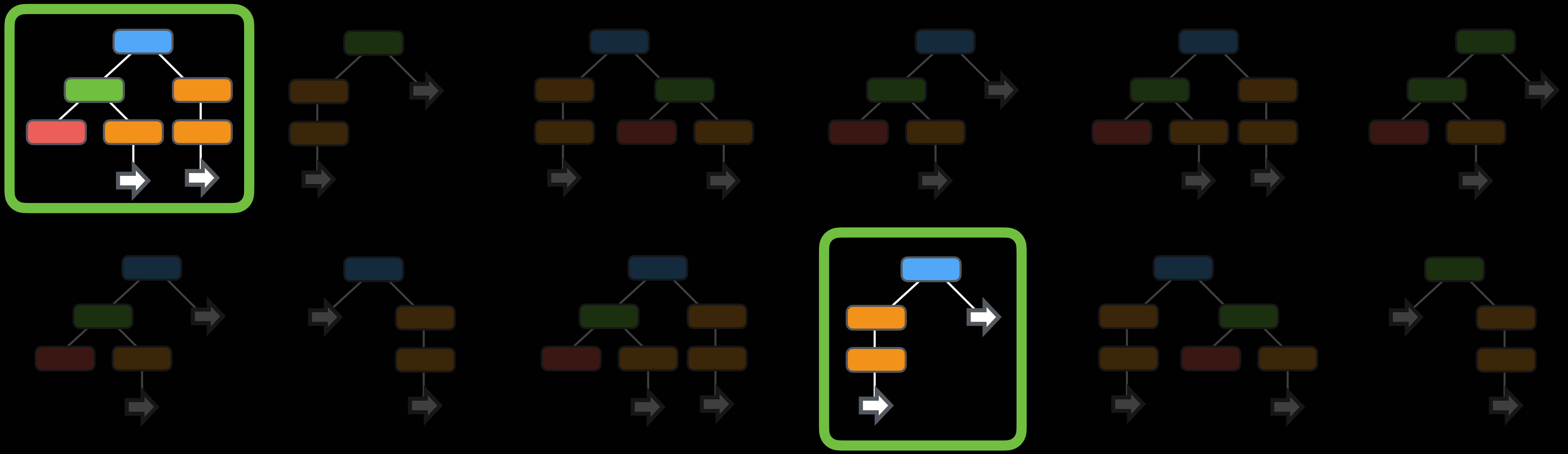


Geneva

Genetic Evasion

Fitness

Which individuals should survive to the next generation?

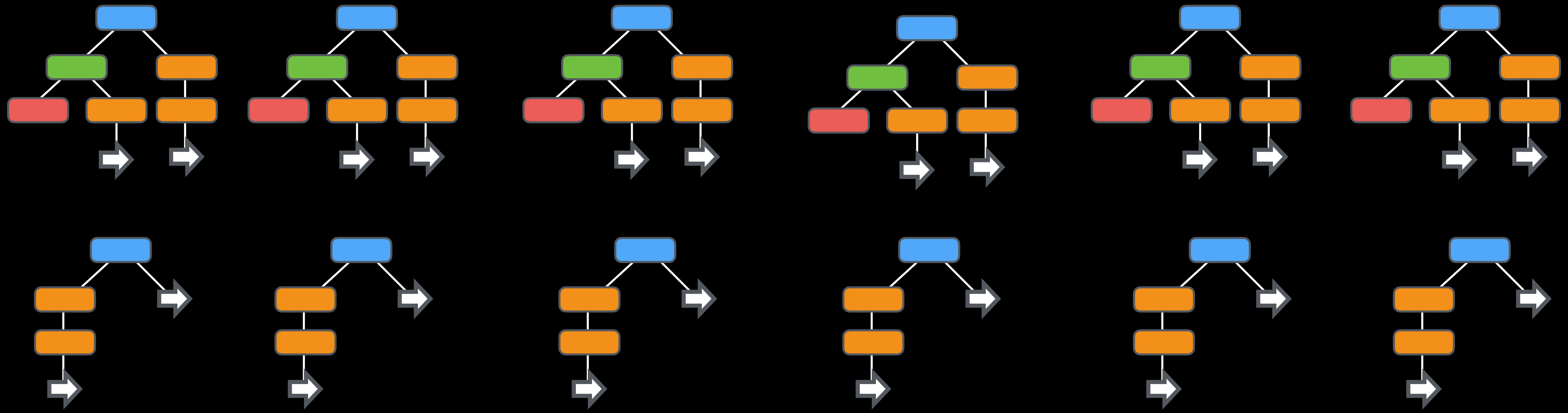


Geneva

Genetic Evasion

Fitness

Which individuals should survive to the next generation?



Geneva

Genetic Evasion

Fitness

Which individuals should survive to the next generation?

- − Not triggering on any packets
- − Breaking the TCP connection
- + Successfully obtaining forbidden content
- + Conciseness

Geneva

Genetic Evasion

Building Blocks

Actions manipulate individual packets

Duplicate

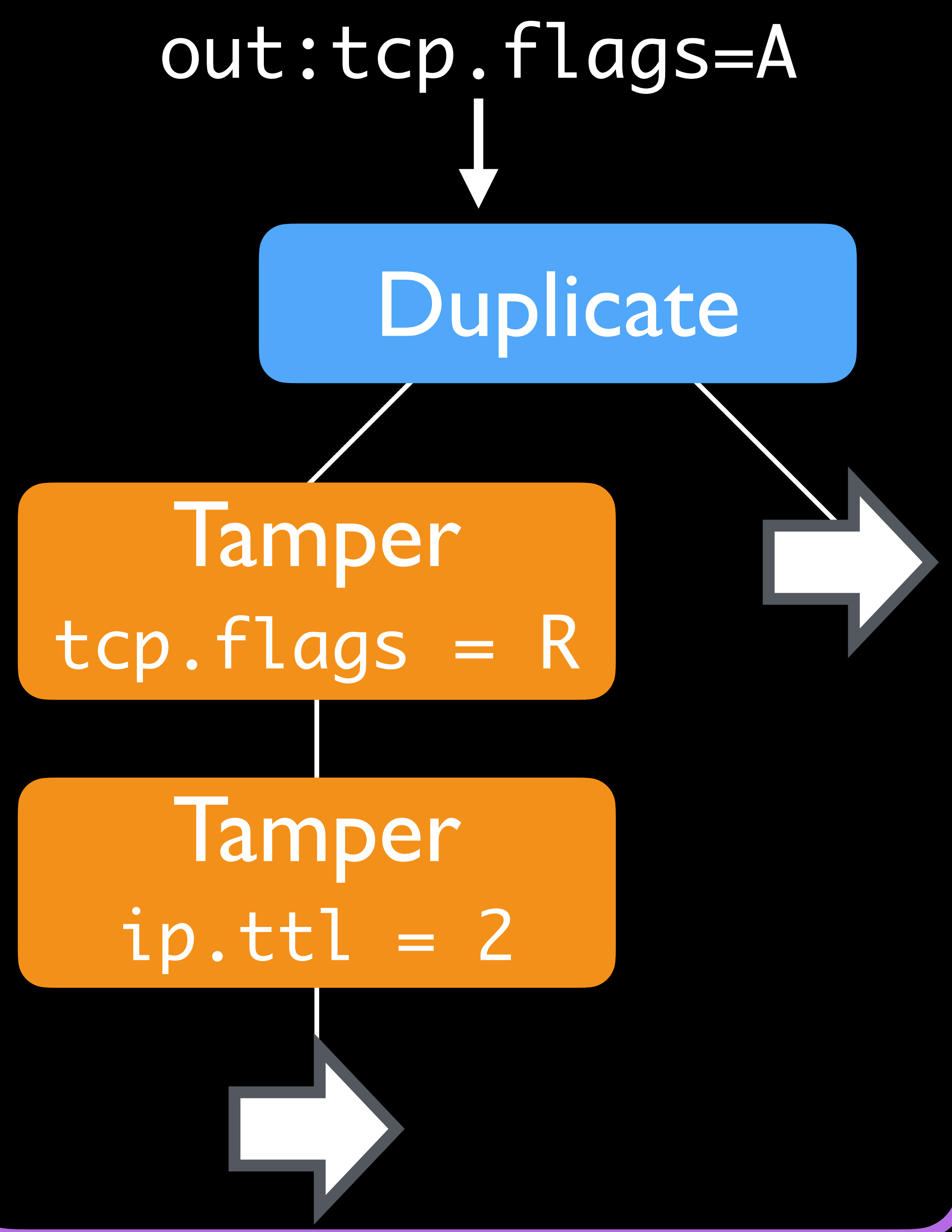
Tamper

Fragment

Drop

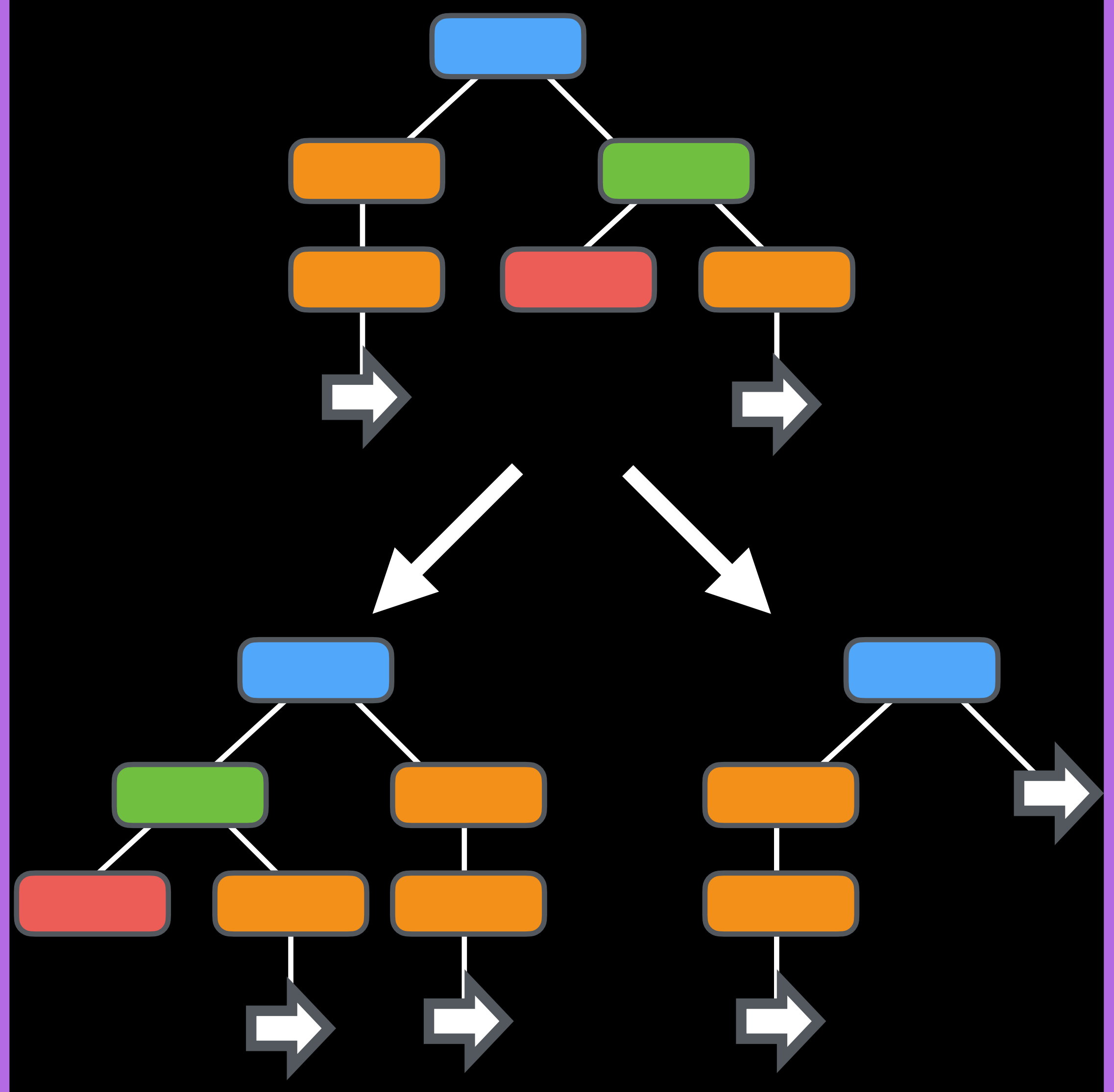
Composition

Actions compose to form trees



Mutation

Randomly alter types, values, and trees



Fitness

Goal: Fewest actions needed to succeed

🟥 No trigger

🟥 Break TCP

✚ Successful

✚ Concise

Geneva’s results – Real censor experiments

		HTTP	HTTPS	DNS	FTP	SMTP
Injects TCP RSTs	China	👮	👮	👮	👮	👮
Injects & blackholes	Iran	👮	👮	👮*		
Injects & blackholes	Kazakhstan	👮	👮			
Injects a block page	India	👮				

Geneva's results – Real censor experiments

Diversity of censors

		HTTP	HTTPS	DNS	FTP	SMTP
Injects TCP RSTs	China	👮	👮	👮	👮	👮
Injects & blackholes	Iran	👮	👮	👮*		
Injects & blackholes	Kazakhstan	👮	👮			
Injects a block page	India	👮				

Geneva’s results – Real censor experiments

Diversity of censors		Diversity of protocols				
		HTTP	HTTPS	DNS	FTP	SMTP
Injects TCP RSTs	🇨🇳 China	👮	👮	👮	👮	👮
Injects & blackholes	🇮🇷 Iran	👮	👮	👮*		
Injects & blackholes	🇰🇿 Kazakhstan	👮	👮			
Injects a block page	🇮🇳 India	👮				

Geneva's results – Real censor experiments

China

India

Iran

Kazakhstan

Geneva's results – Real censor experiments

6 Species

13 Sub-species

36 Variants

China

India

Iran

Kazakhstan

Geneva's results – Real censor experiments

6 Species	The underlying bug
13 Sub-species	How Geneva exploits it
36 Variants	Functionally distinct

China

India

Iran

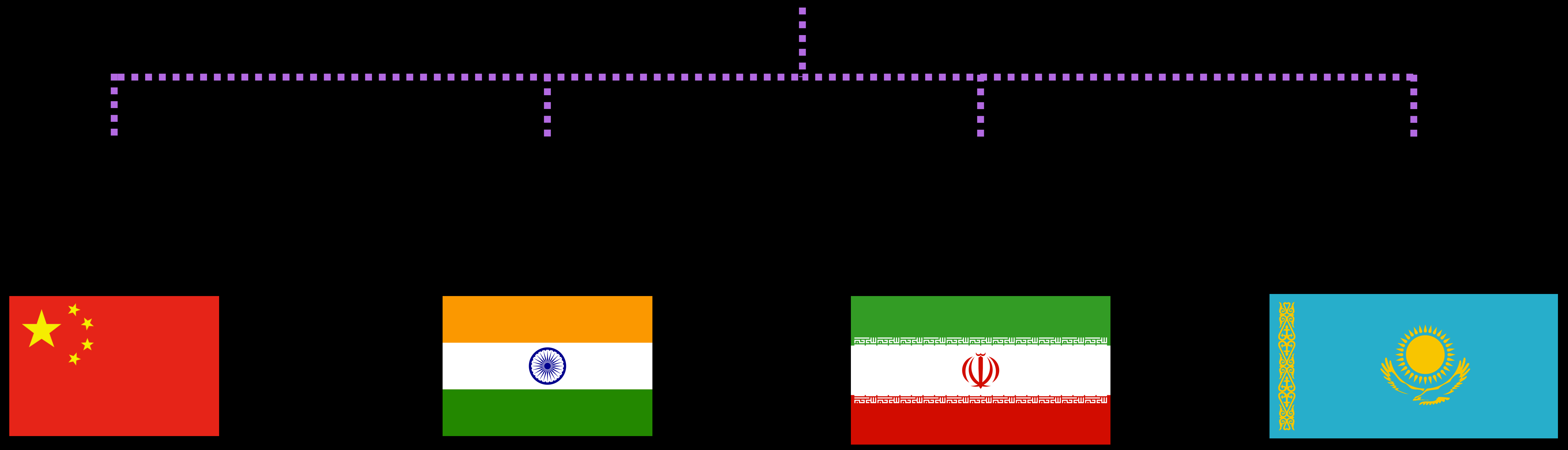
Kazakhstan

Geneva's results – Real censor experiments

6 Species .... The underlying bug

13 Sub-species .... How Geneva exploits it

36 Variants .... Functionally distinct



China

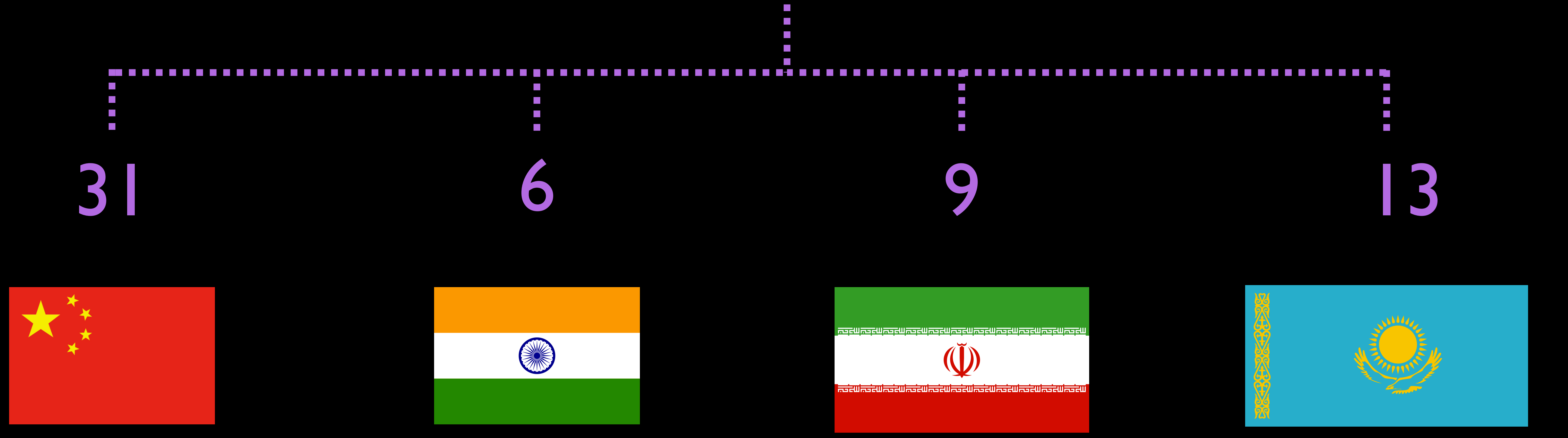
India

Iran

Kazakhstan

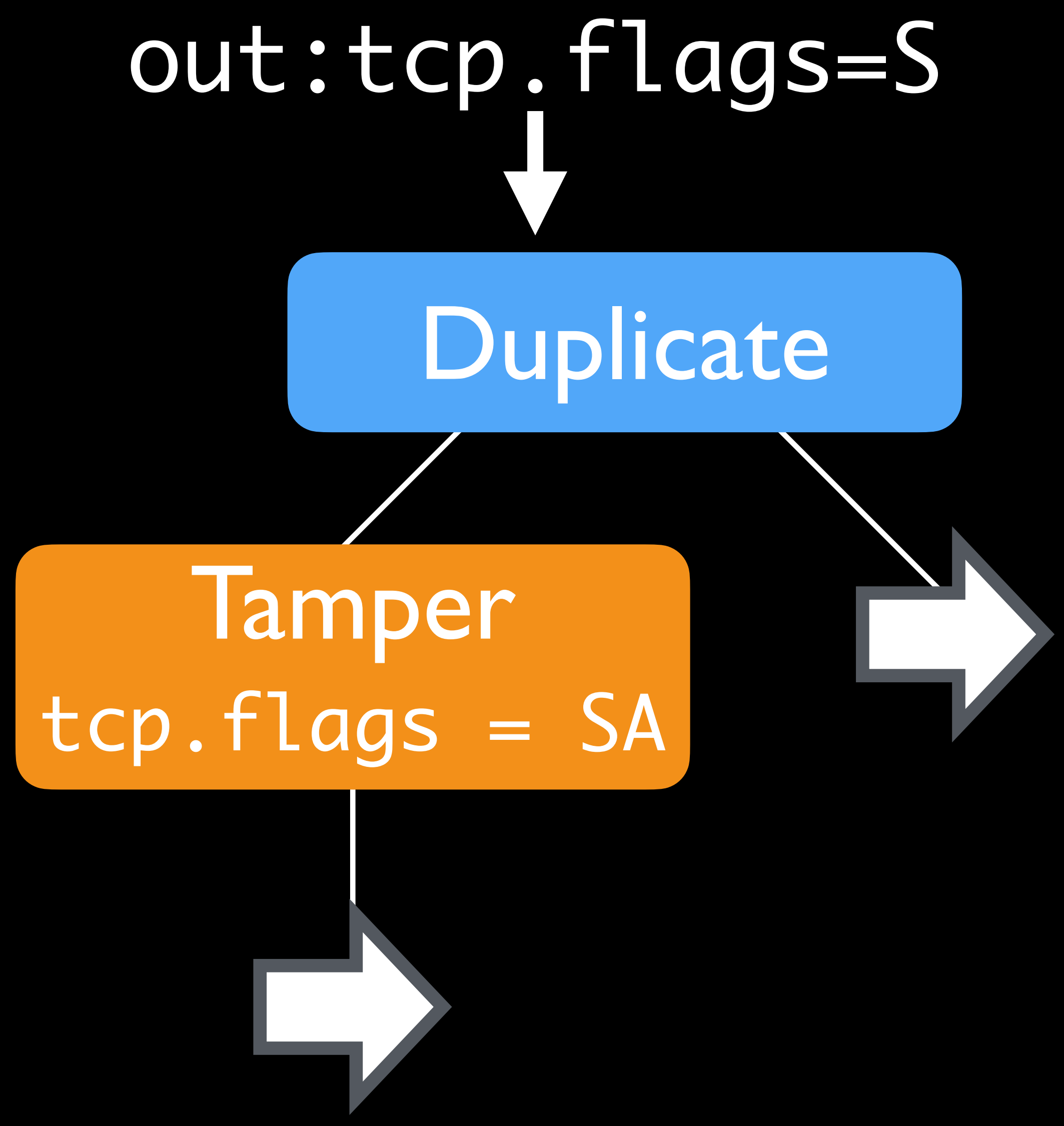
Geneva’s results – Real censor experiments

6	Species	The underlying bug
13	Sub-species	How Geneva exploits it
36	Variants	Functionally distinct



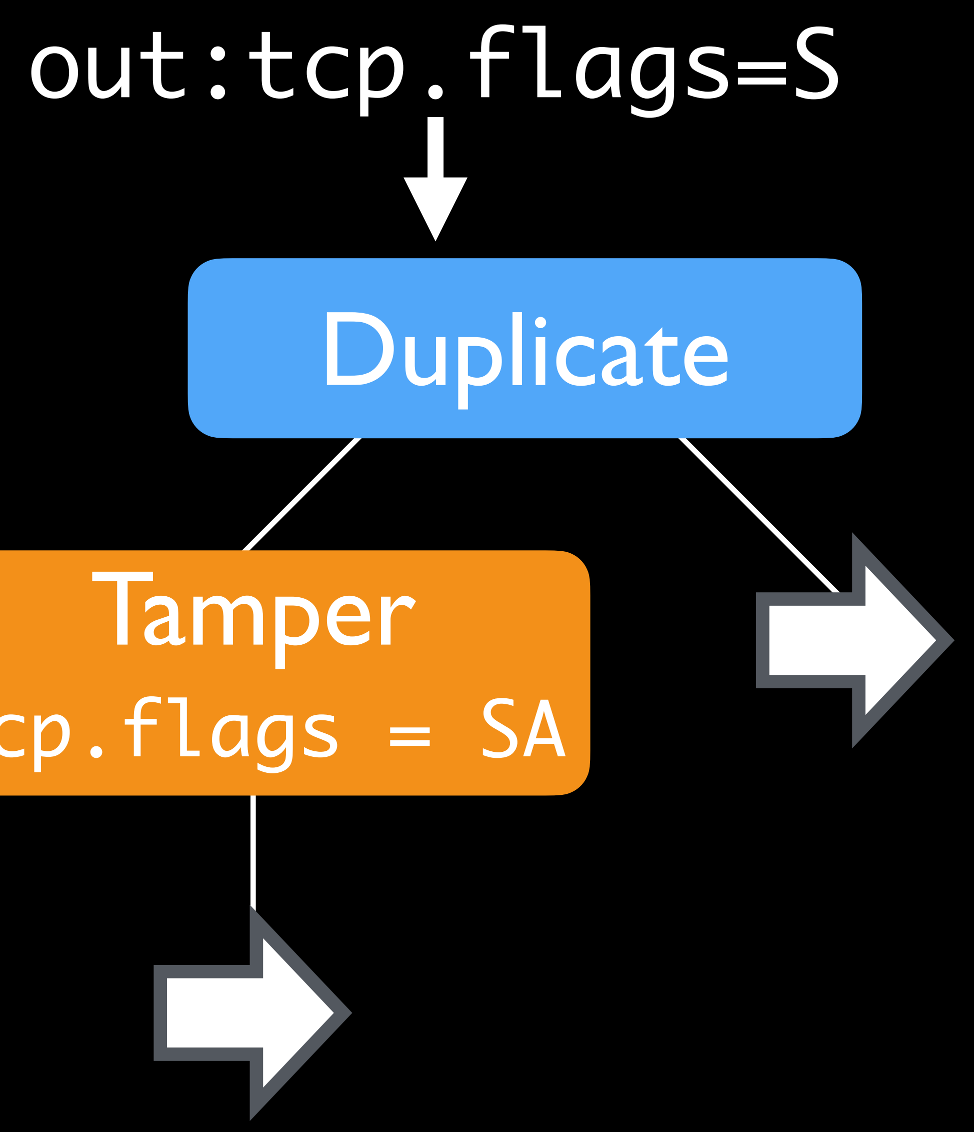
31	6	9	13
China	India	Iran	Kazakhstan

Turnaround species



Trick the censor into thinking the client is the server

Turnaround species



out:tcp.flags=S

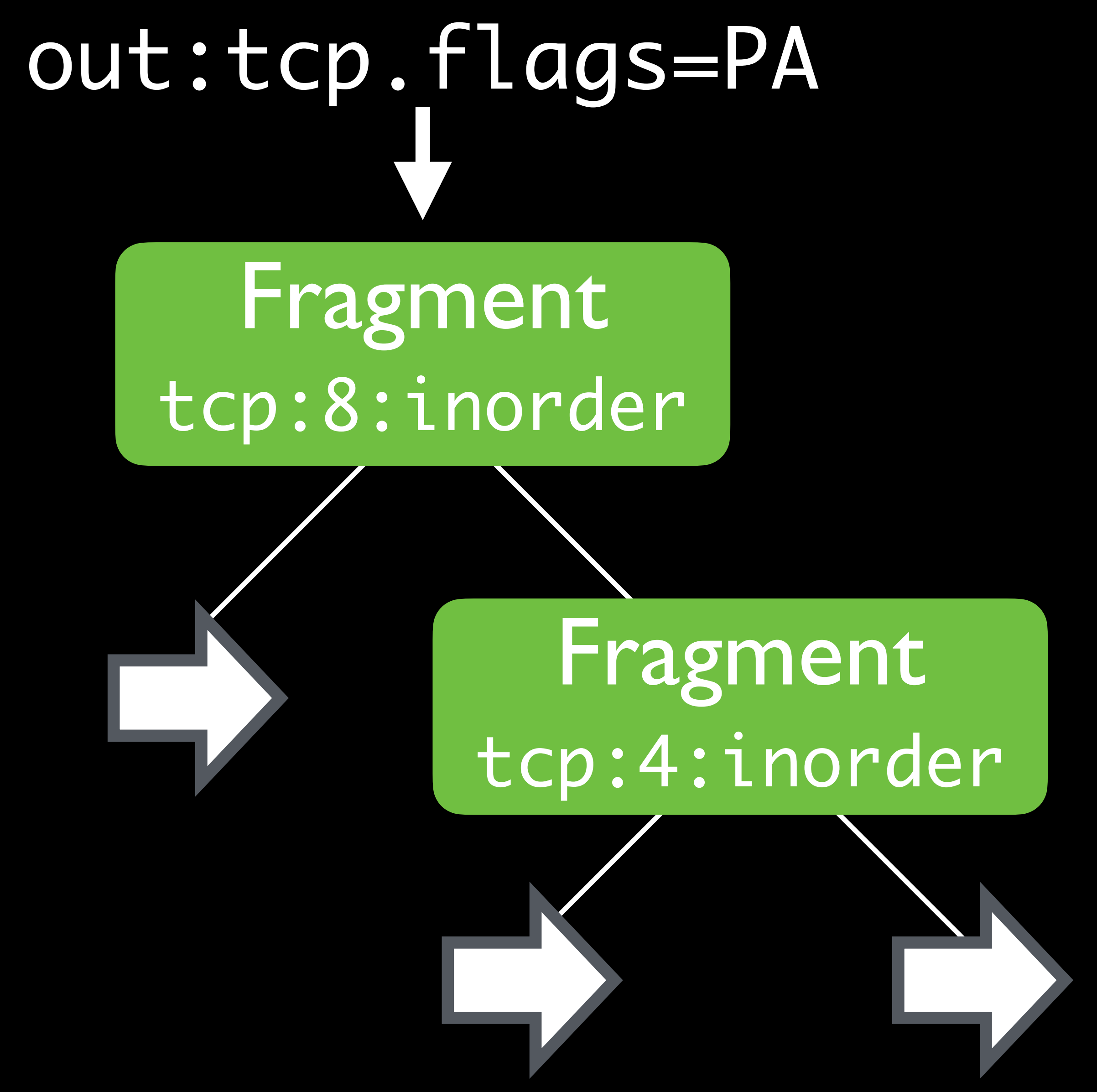
Duplicate

Tamper

tcp.flags = SA

Trick the censor into thinking the client is the server

Segmentation species



out:tcp.flags=PA

Fragment

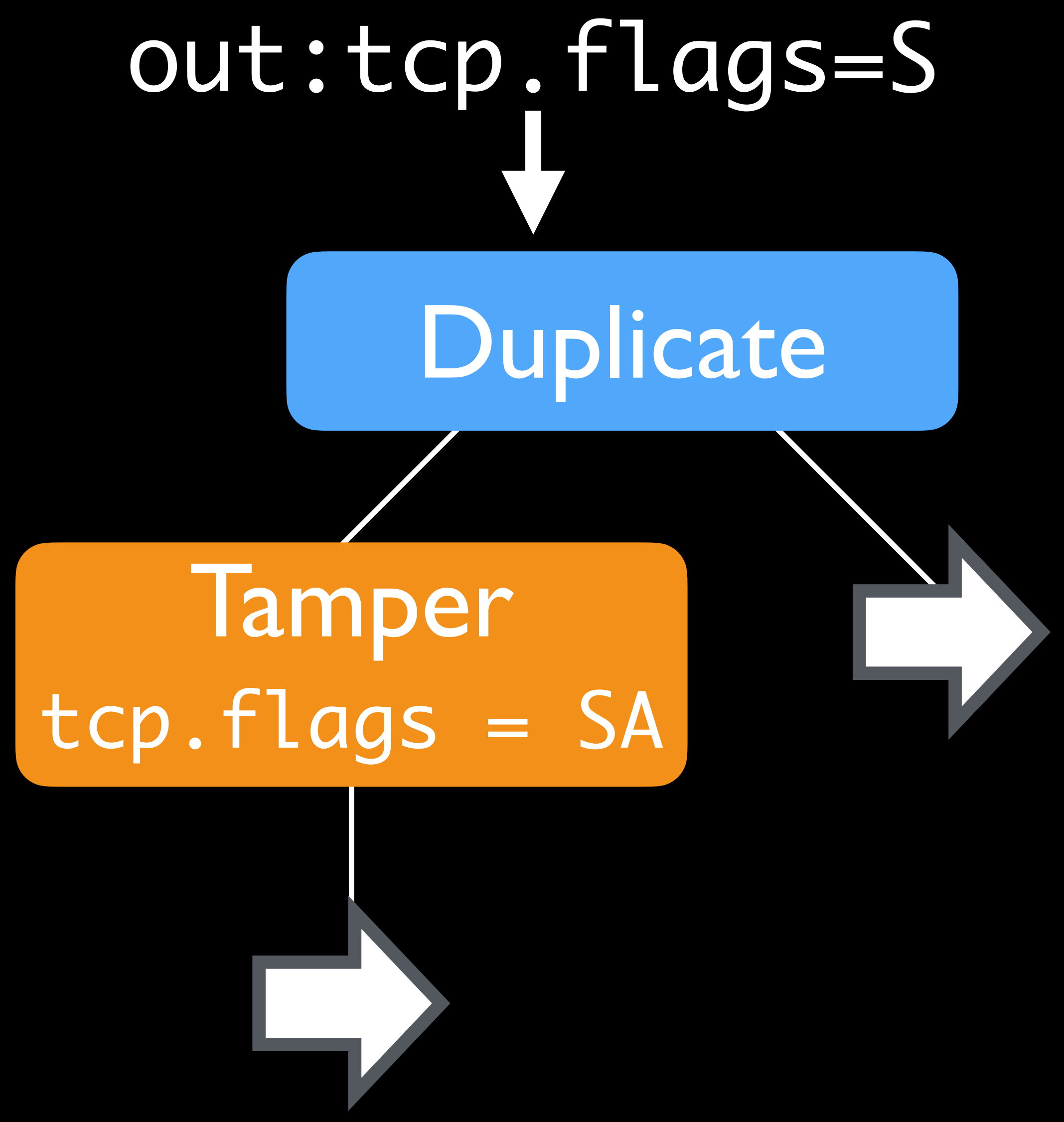
tcp:8:inorder

Fragment

tcp:4:inorder

Segment the request

Turnaround species



out:tcp.flags=S

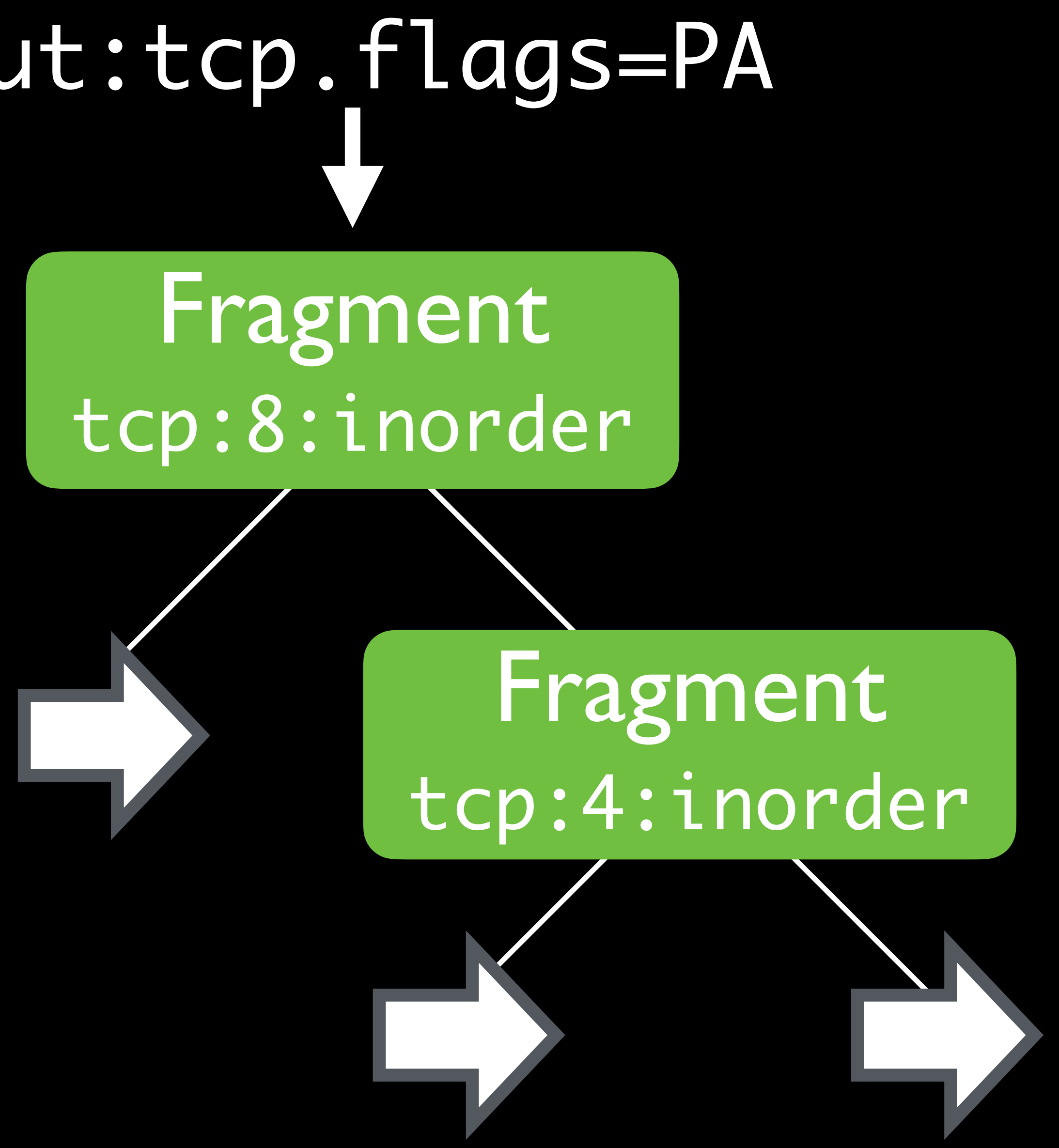
Duplicate

Tamper

tcp.flags = SA

Trick the censor into thinking the client is the server

Segmentation species



out:tcp.flags=PA

Fragment

tcp:8:inorder

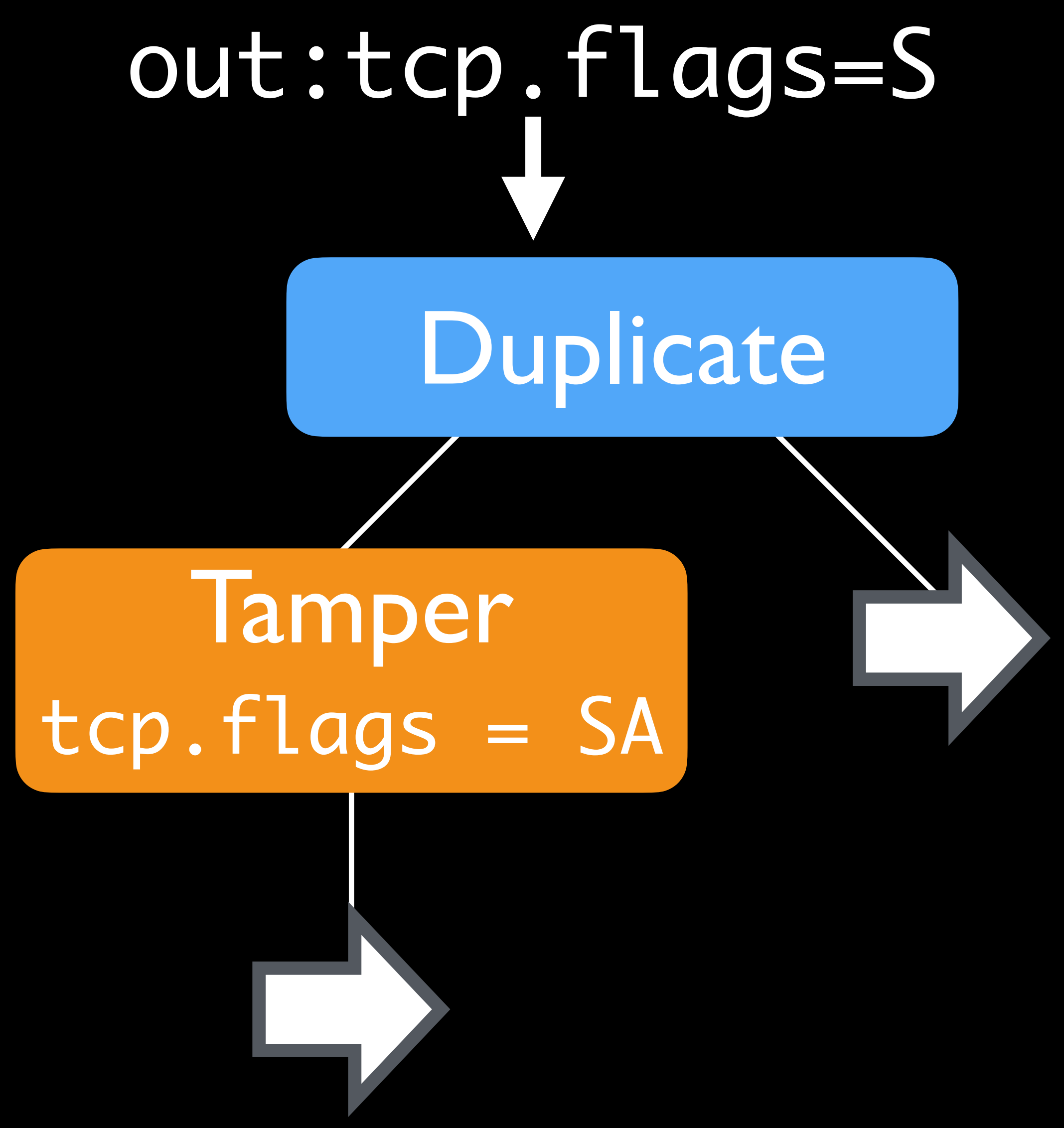
Fragment

tcp:4:inorder

```
GET /?search=ultrasurf
```

Segment the request

Turnaround species



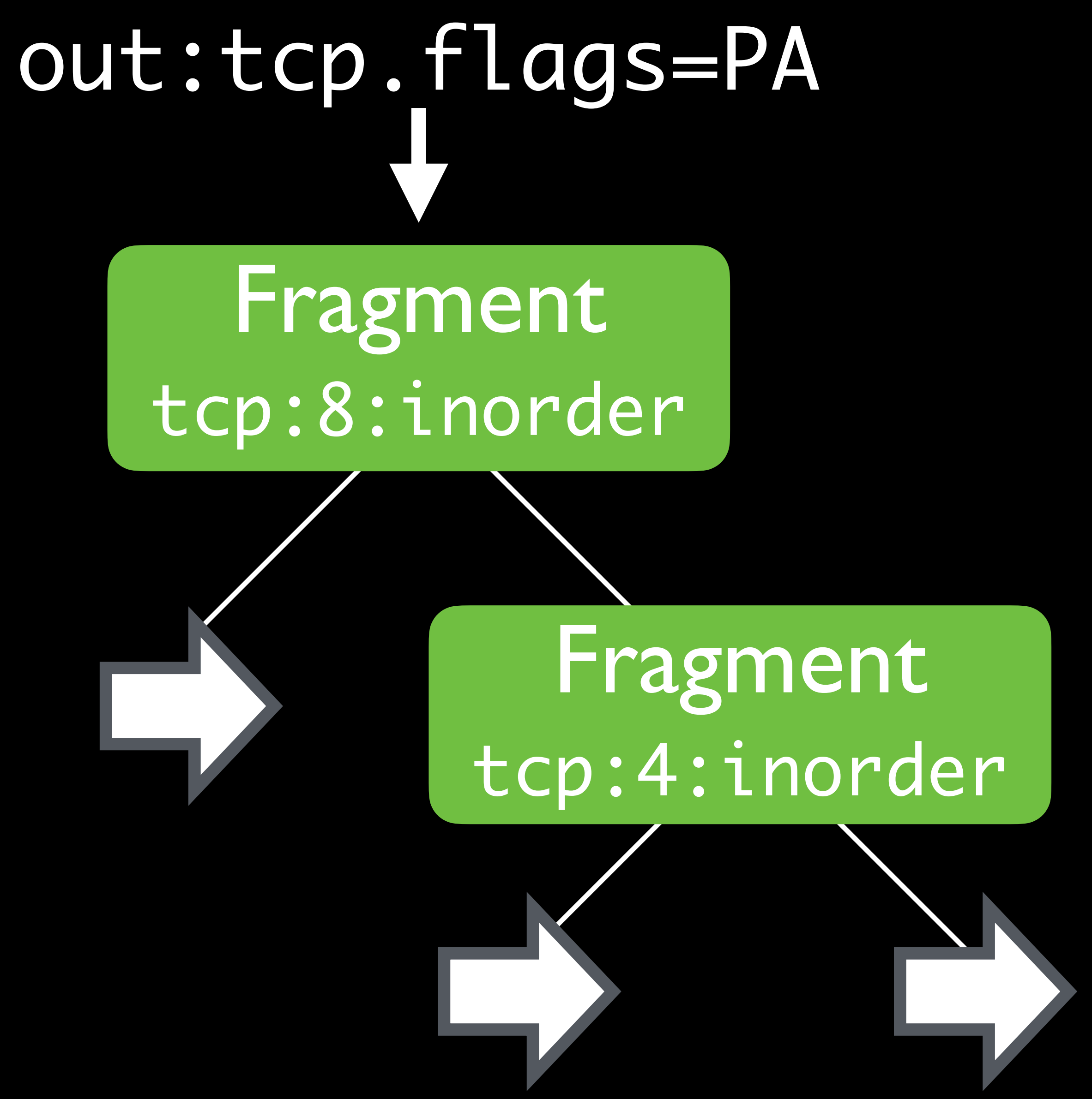
out:tcp.flags=S

Duplicate

Tamper
tcp.flags = SA

Trick the censor into thinking the client is the server

Segmentation species



out:tcp.flags=PA

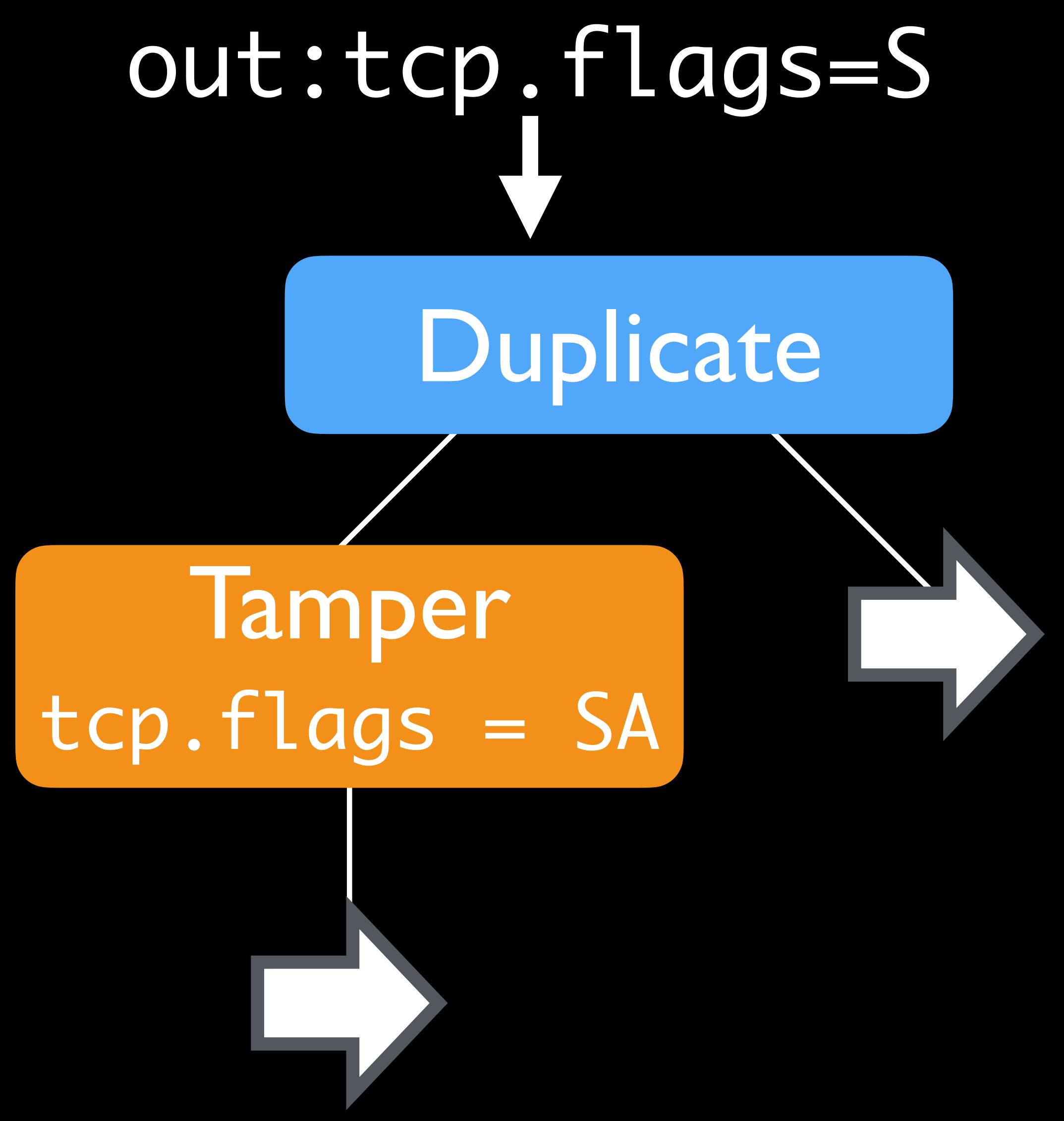
Fragment
tcp:8:inorder

Fragment
tcp:4:inorder

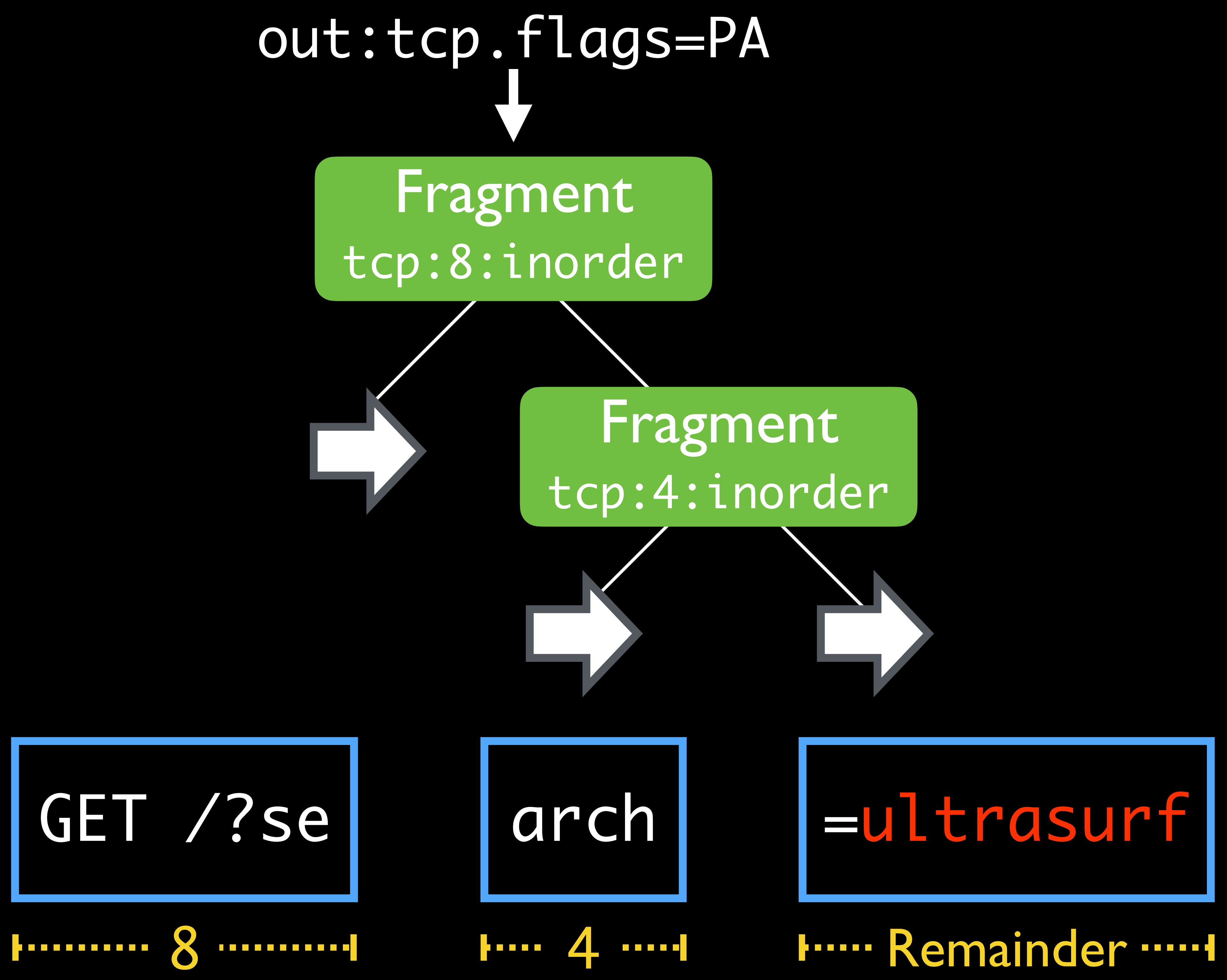
```
GET /?search=ultrasurf
```

Segment the request

Turnaround species



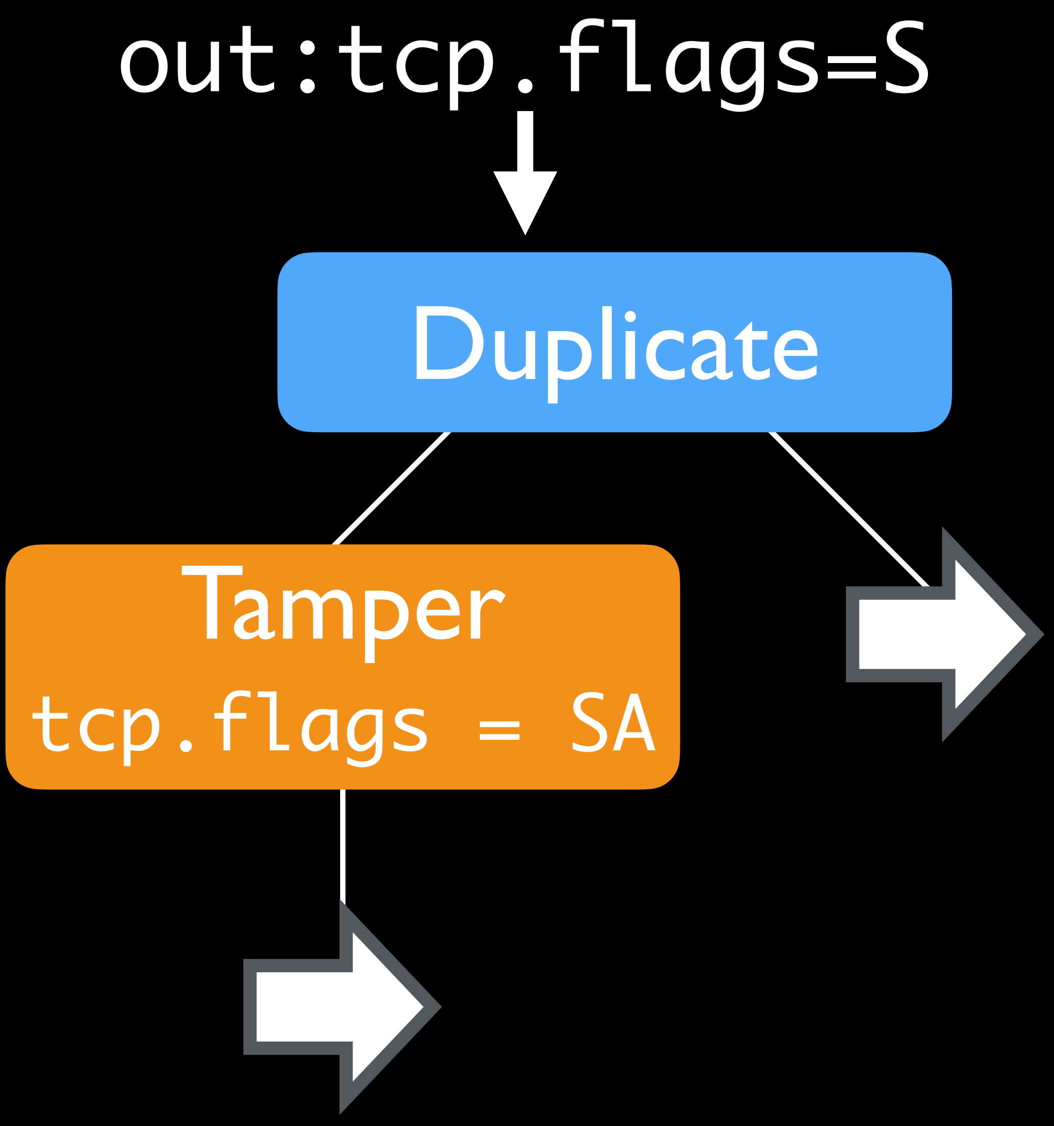
Segmentation species



Trick the censor into thinking the client is the server

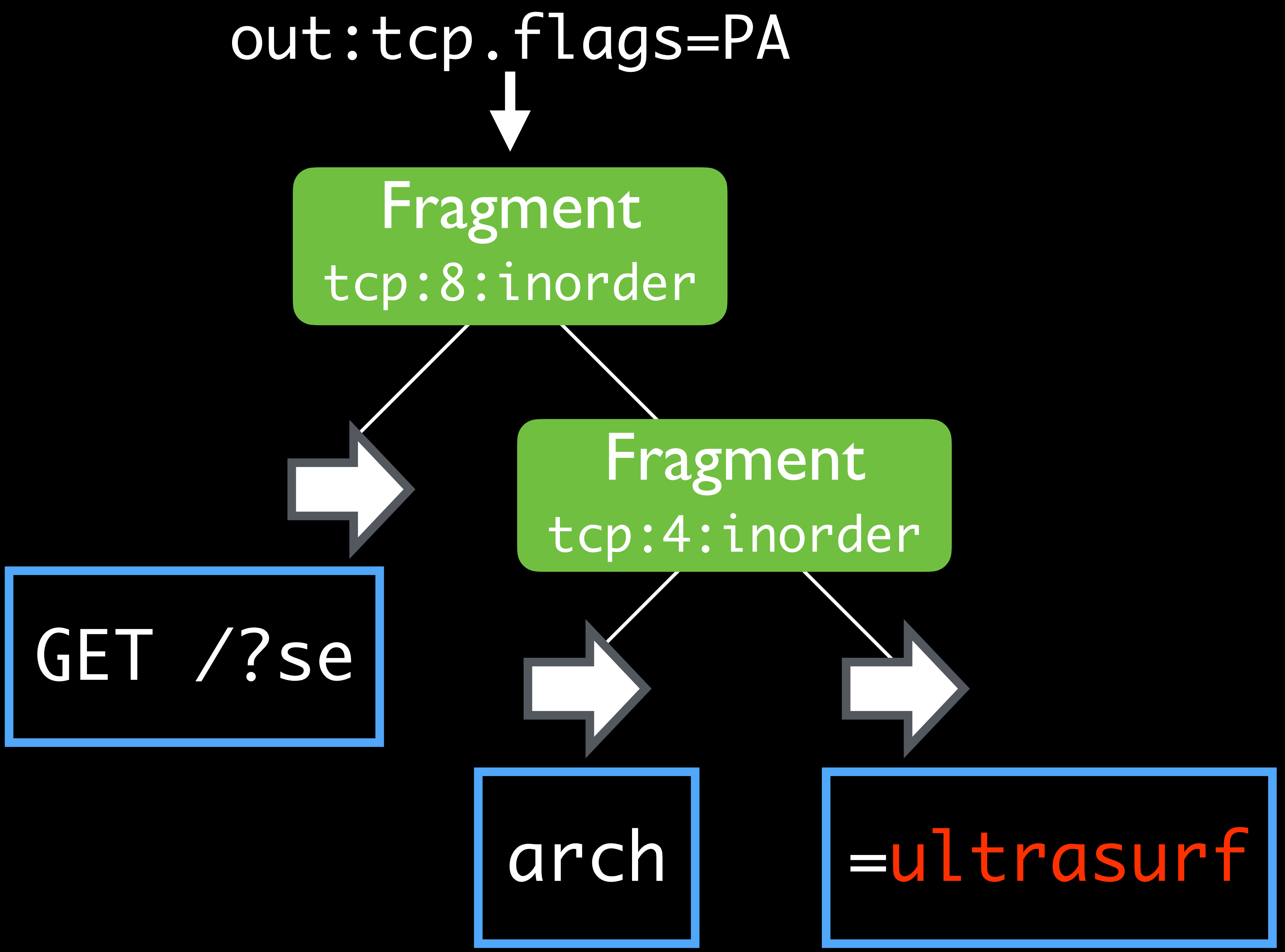
Segment the request, but *not the keyword*

Turnaround species



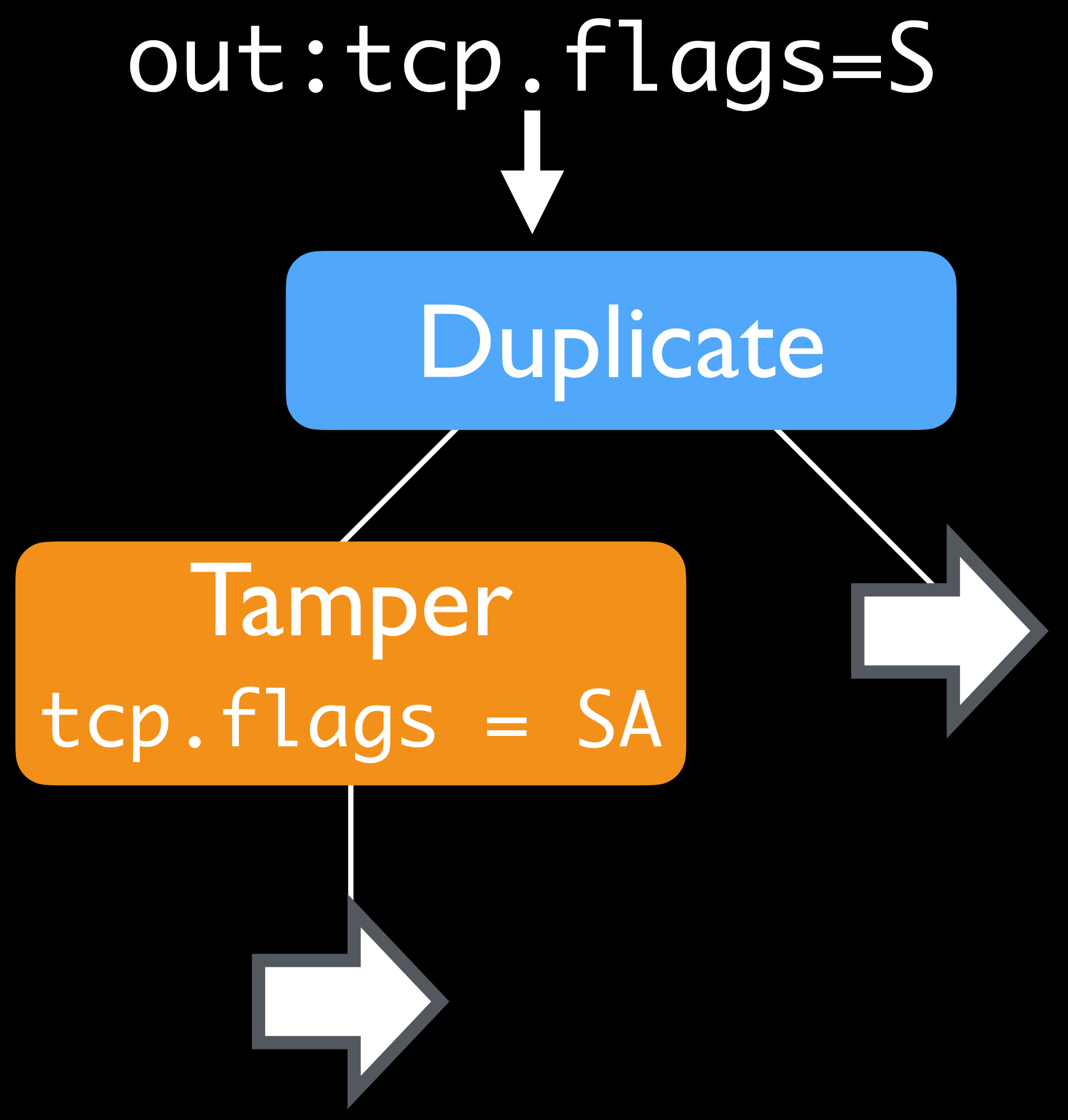
Trick the censor into thinking the client is the server

Segmentation species



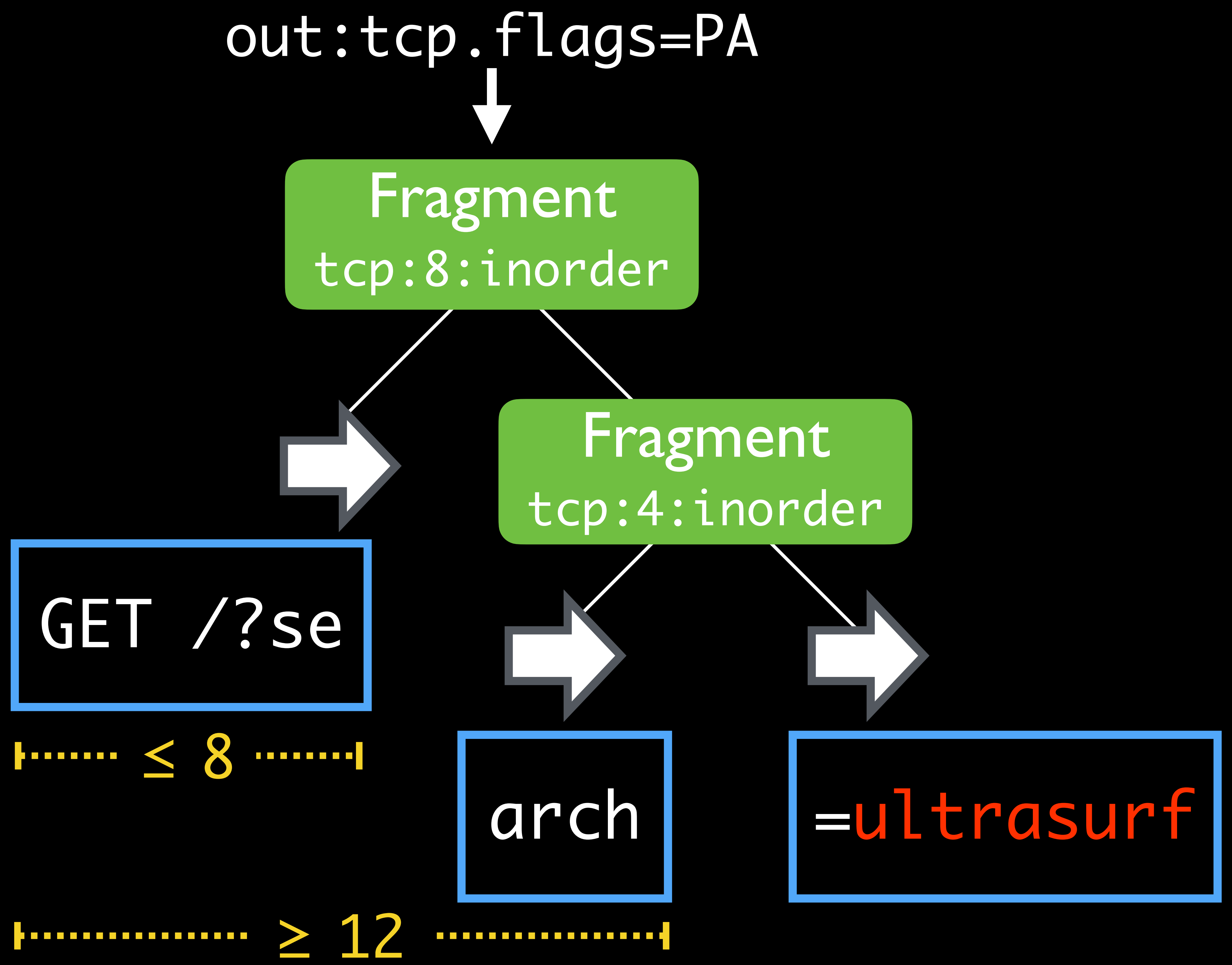
Segment the request, but *not the keyword*

🇰🇿 Turnaround species

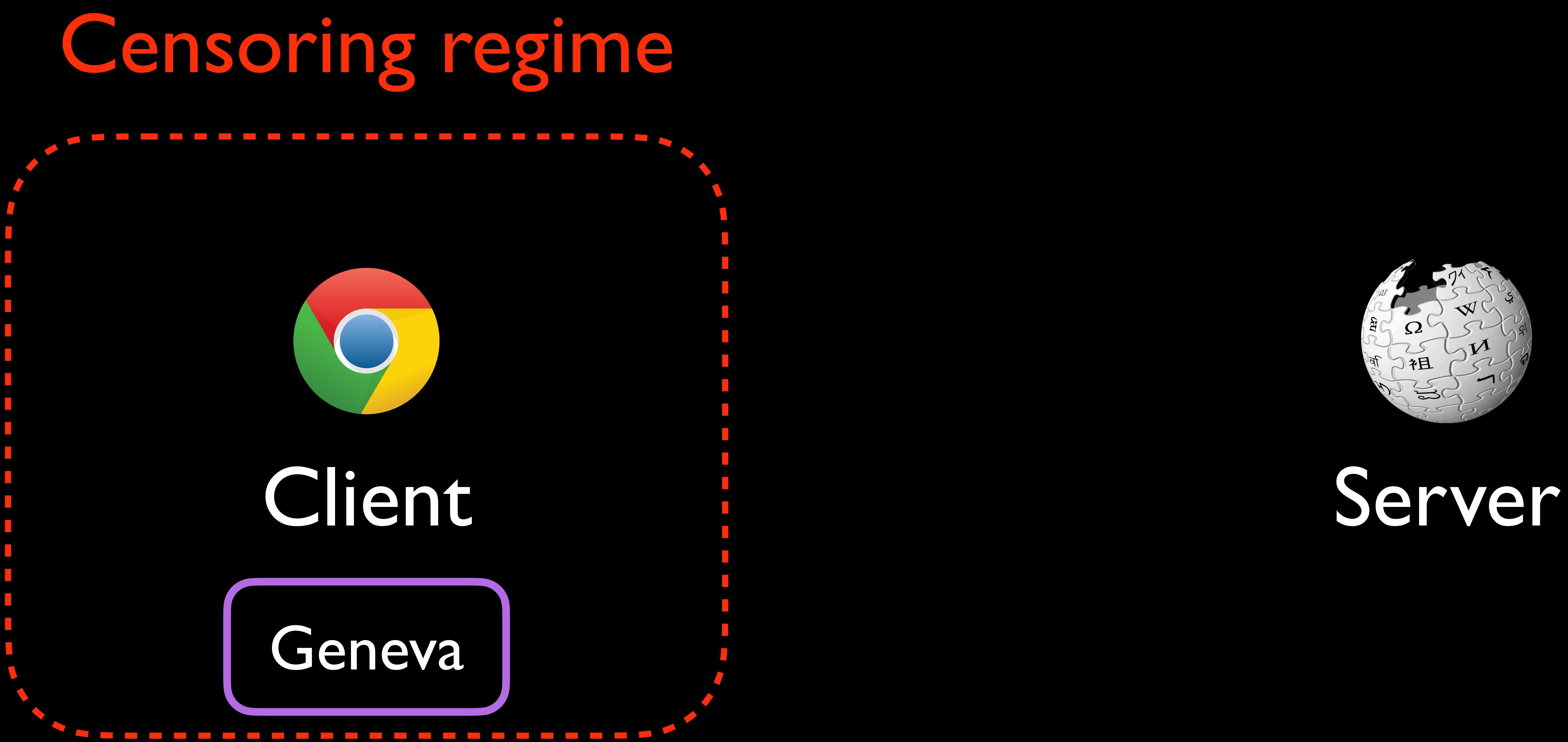


Trick the censor into thinking the client is the server

🇨🇳 Segmentation species



Segment the request, but *not the keyword*



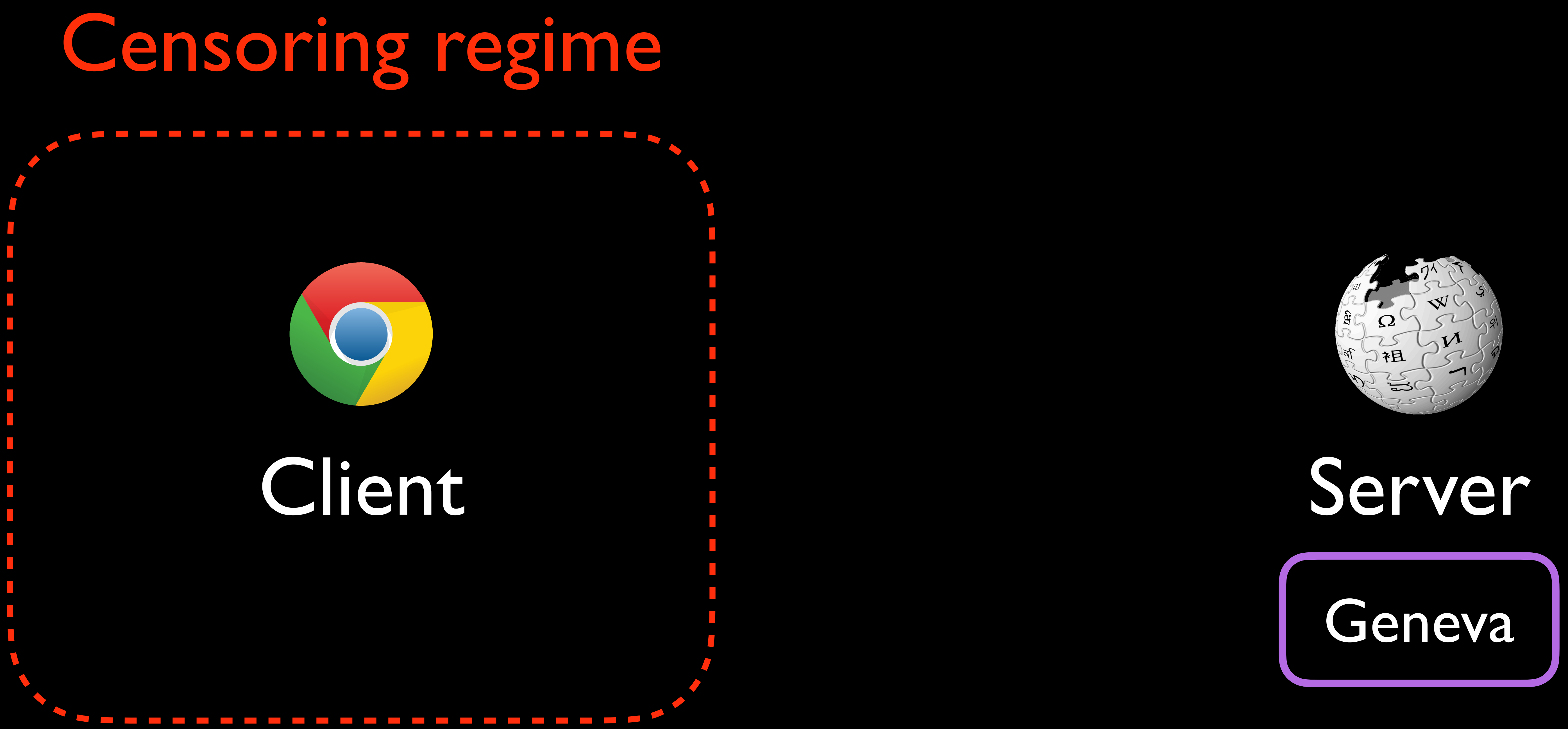
Censoring regime

Client

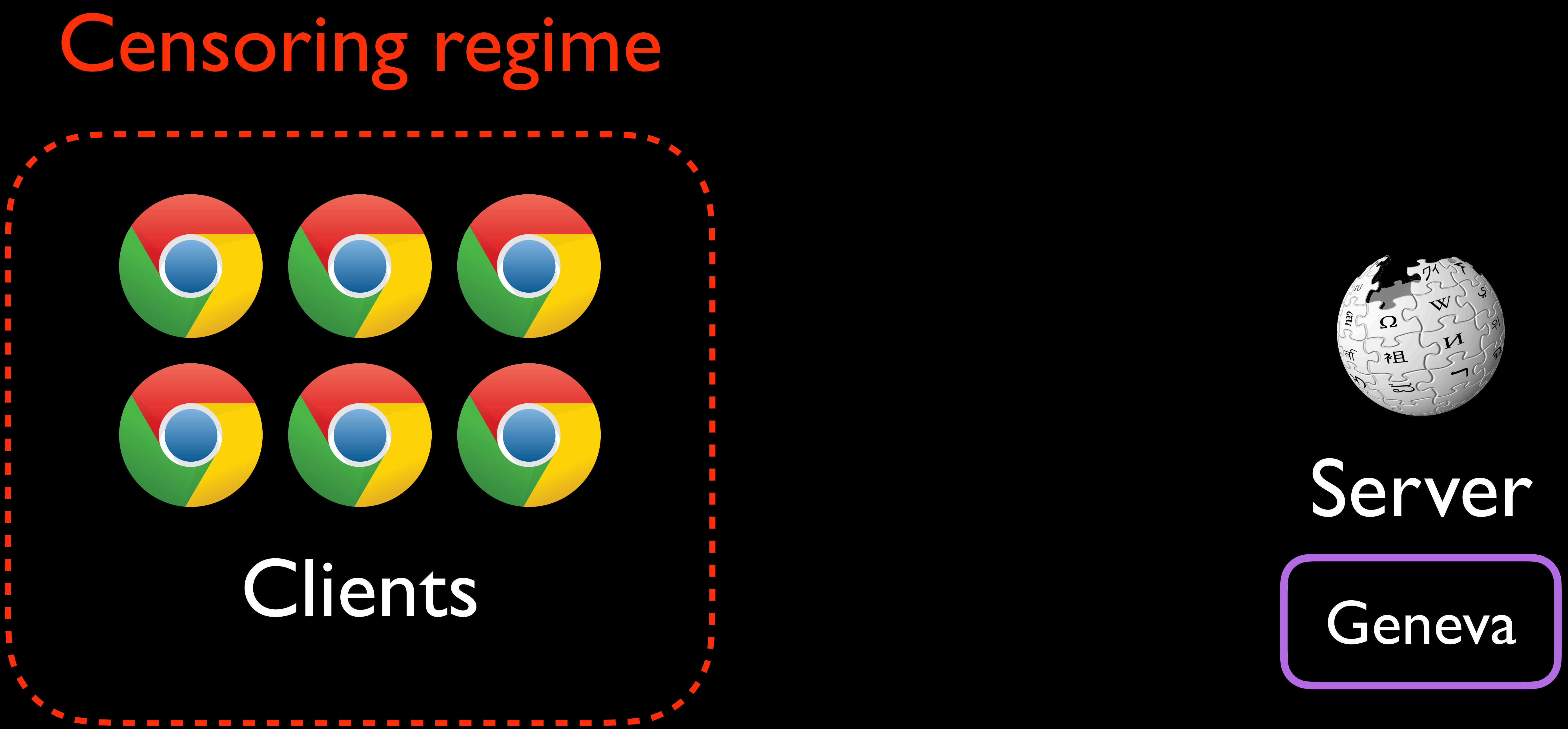
Geneva

Server

Server-side evasion

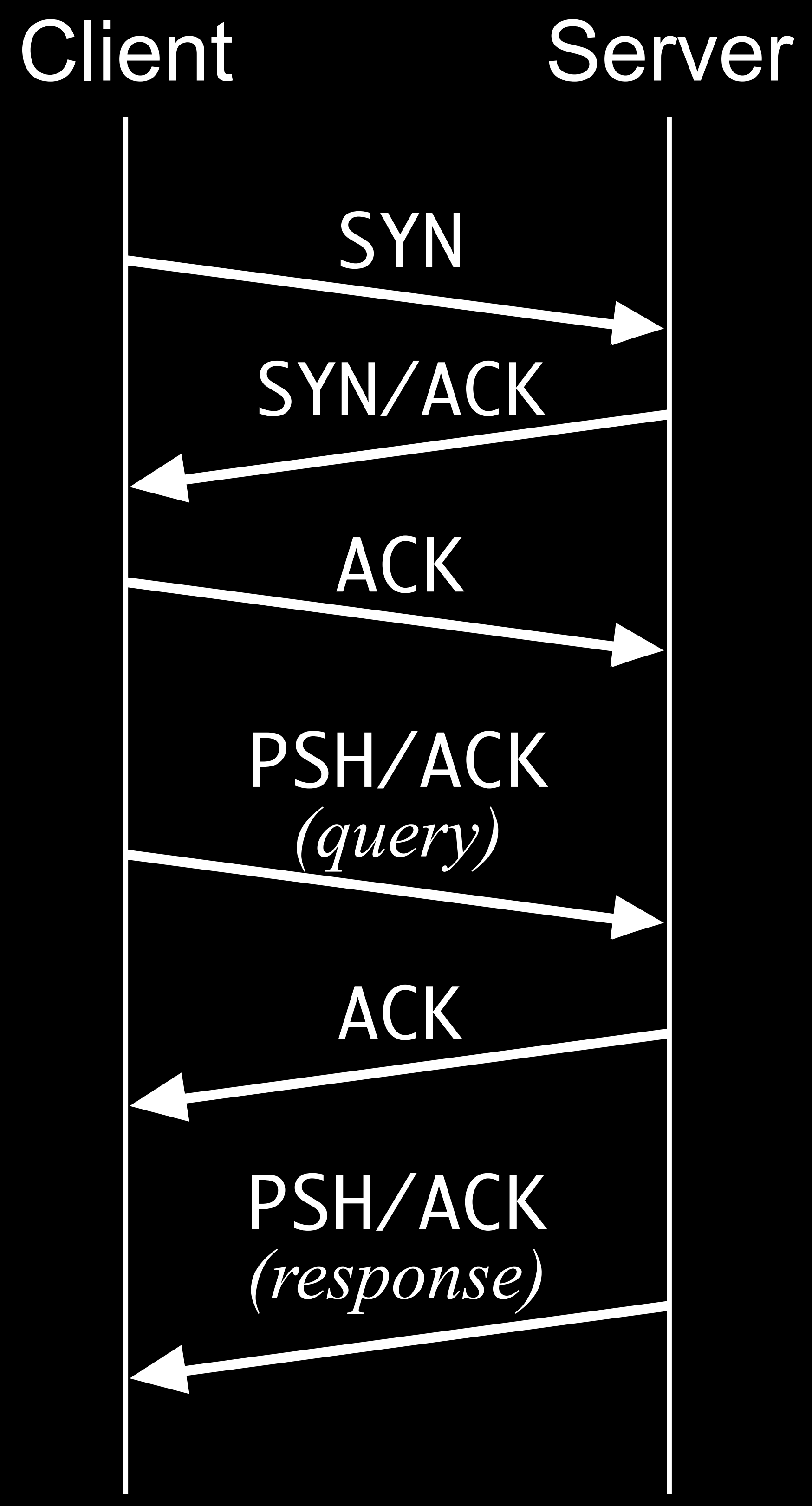


Server-side evasion

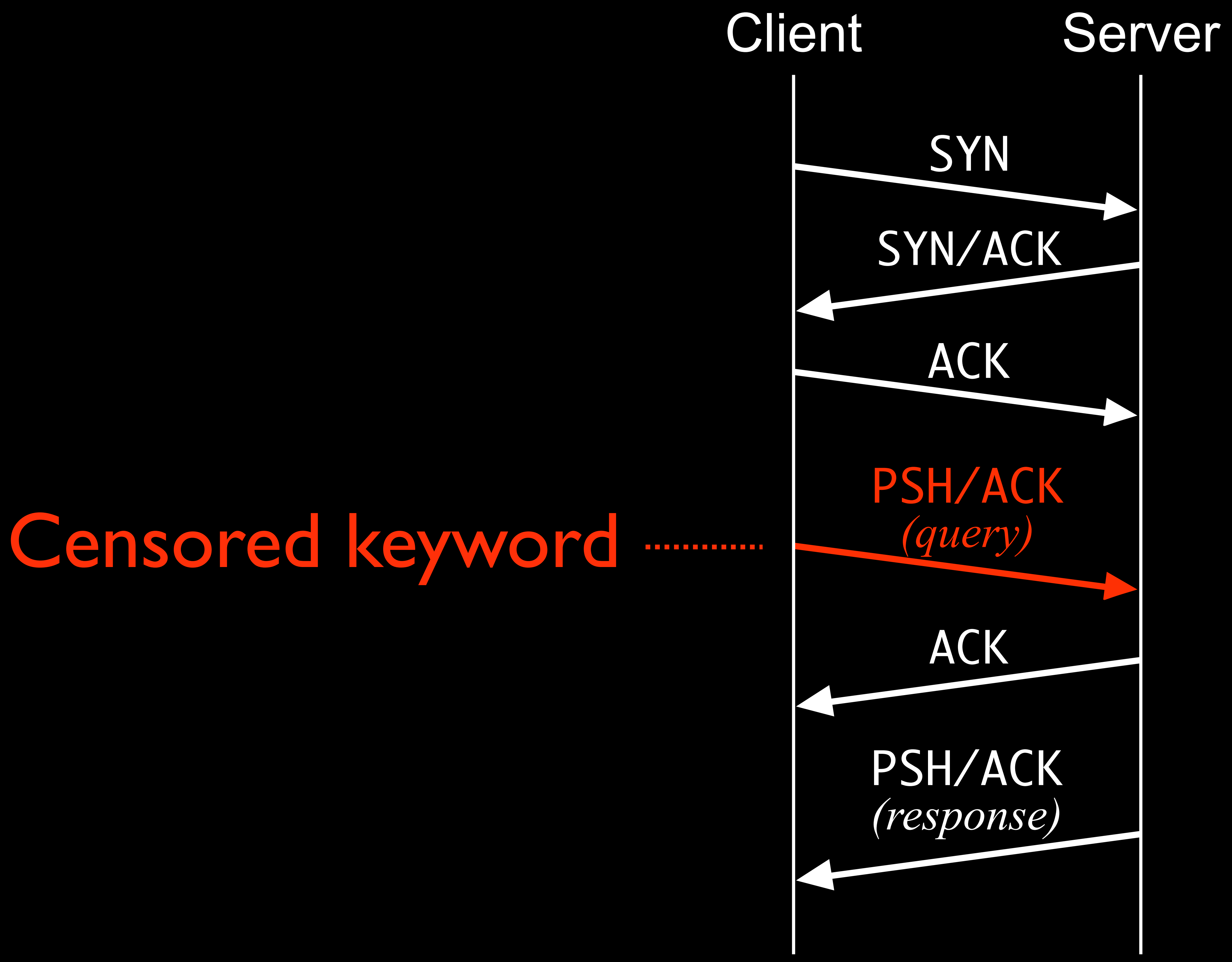


Potentially broadens reachability
without *any* client-side deployment

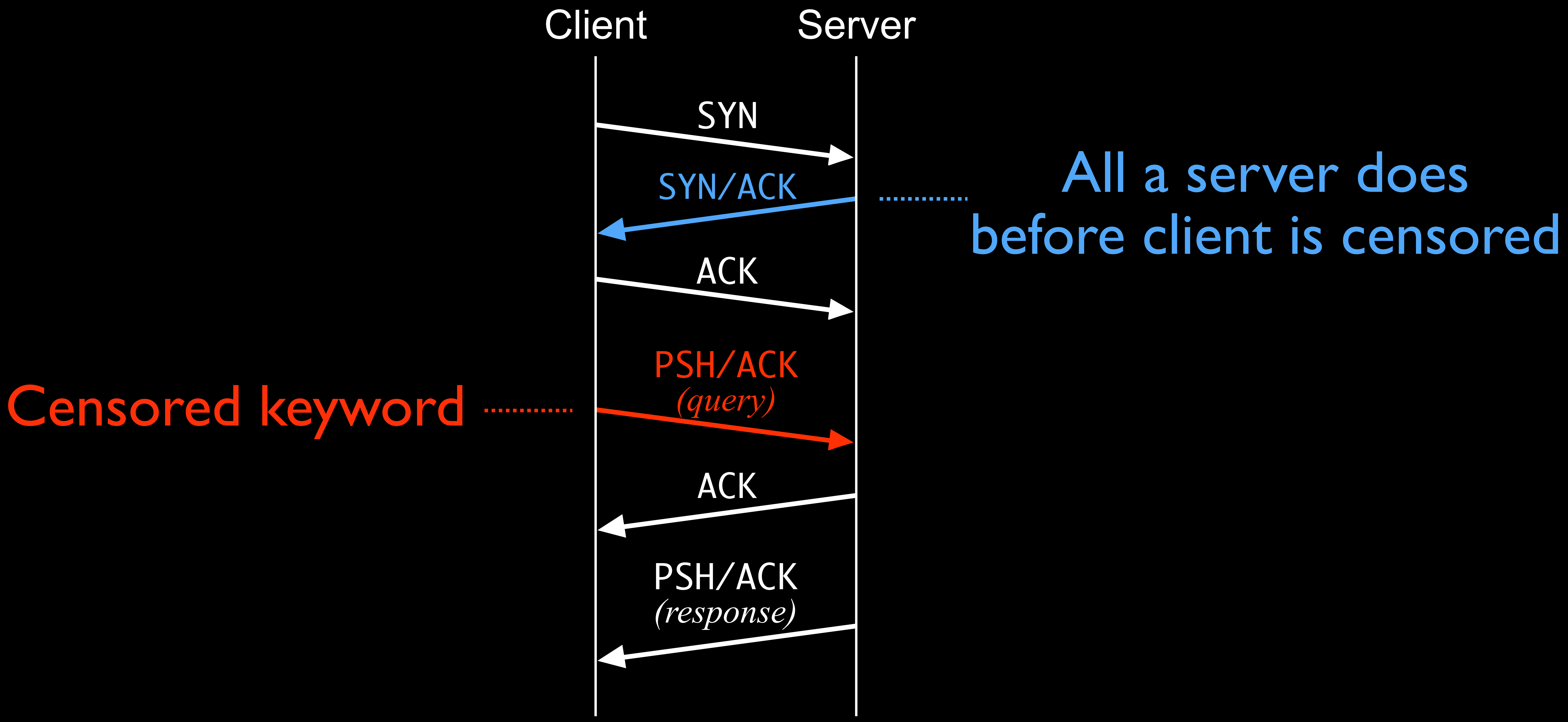
Server-side evasion “shouldn’t” work



Server-side evasion “shouldn’t” work



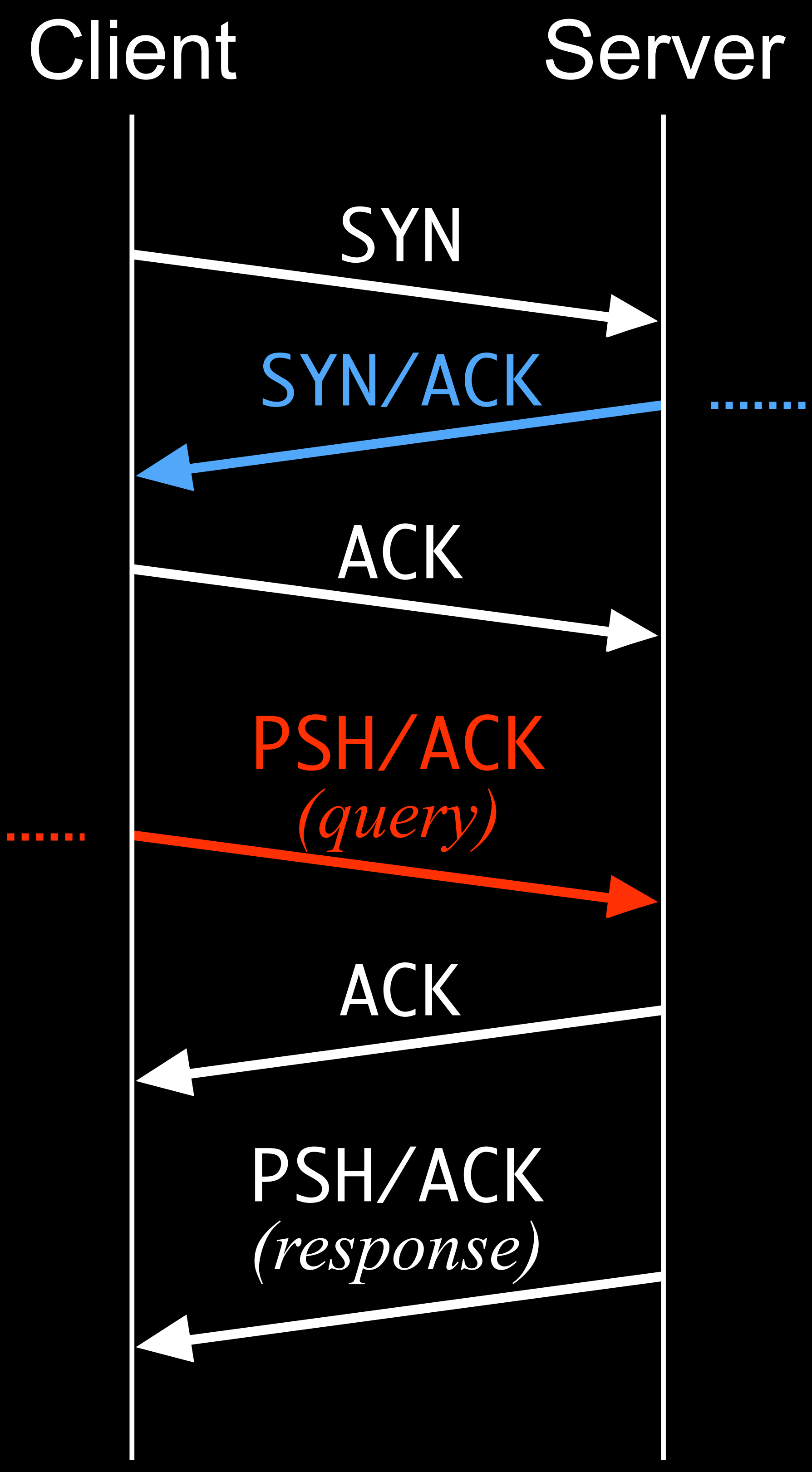
Server-side evasion “shouldn’t” work



All a server does before client is censored

Censored keyword

Server-side evasion “shouldn’t” work



All a server does before client is censored

Censored keyword

Fortunately, the AI doesn’t know it “shouldn’t” work

Server-side evasion “shouldn’t” work

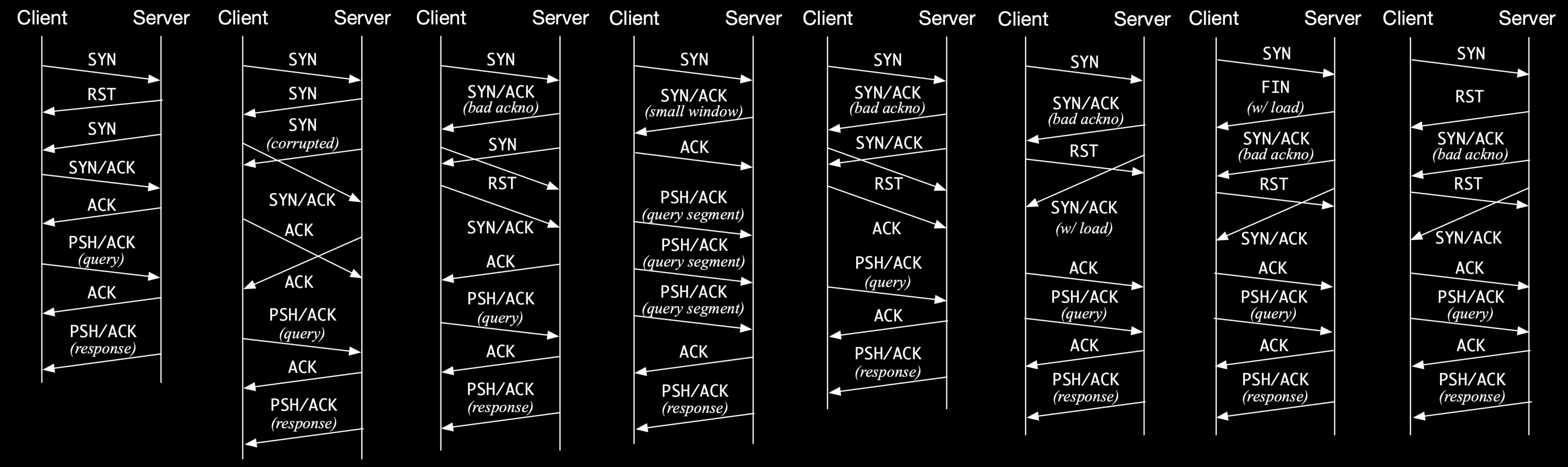
Server-side results

Server-side evasion “shouldn’t” work

Server-side results

China

8 strategies

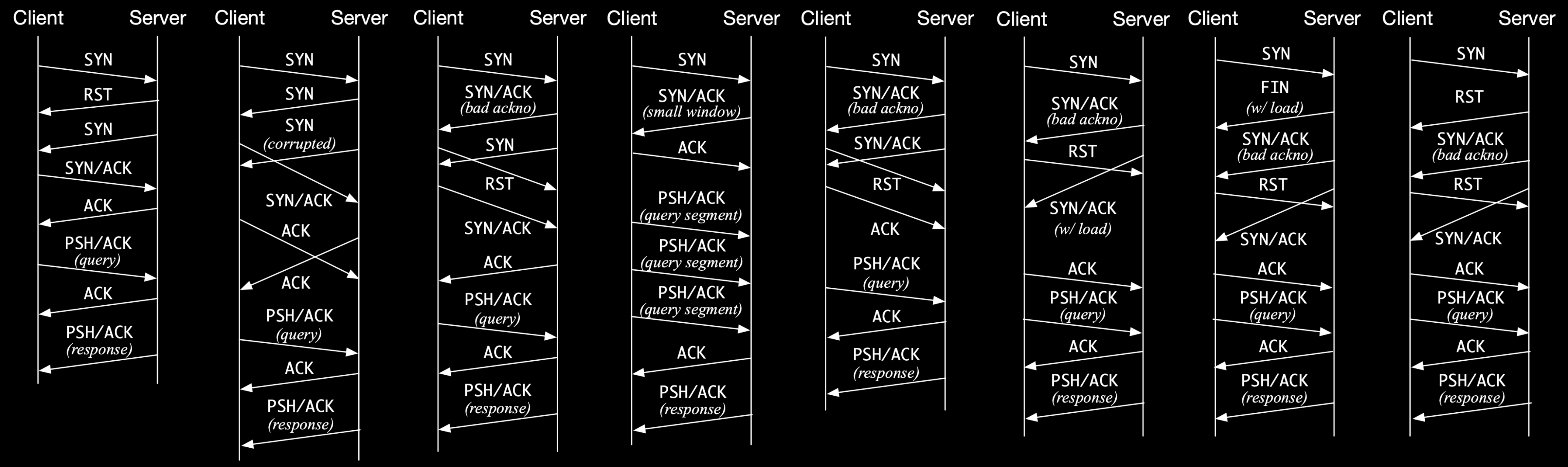


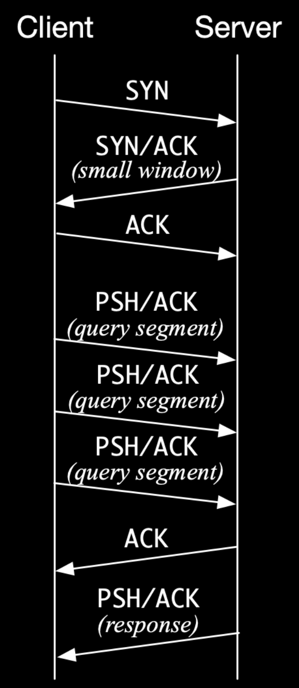
Server-side evasion “shouldn’t” work

Server-side results

China

8 strategies





Iran/India

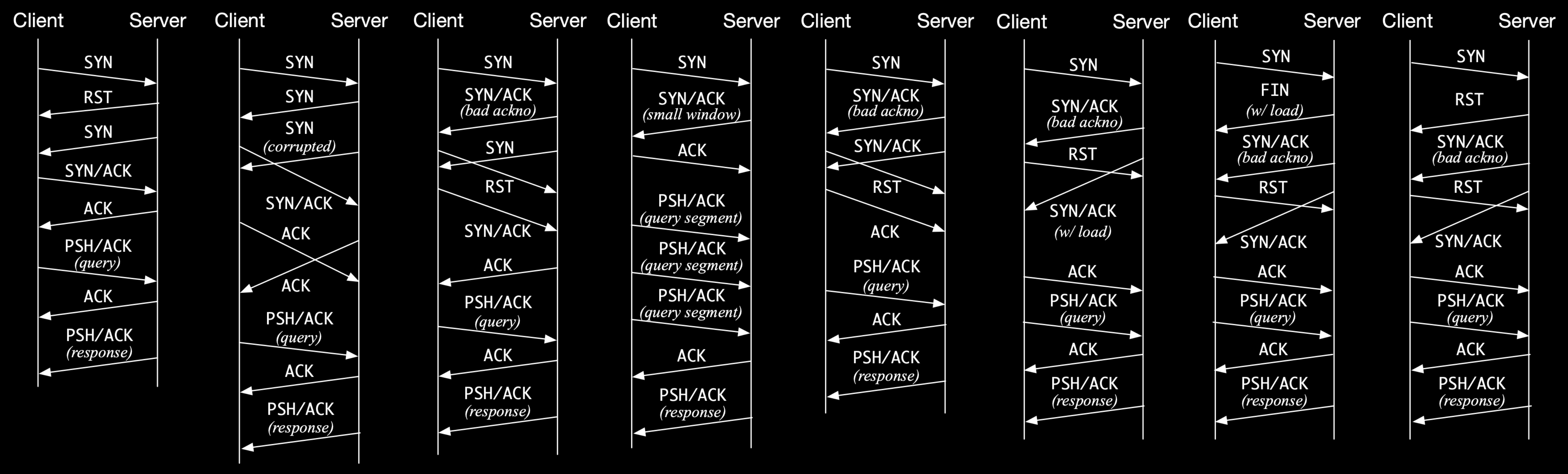
1 strategy

Server-side evasion "shouldn't" work

Server-side results

China

8 strategies



Client Server
SYN
RST
SYN
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN
SYN *(corrupted)*
SYN/ACK
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(bad ackno)*
SYN
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

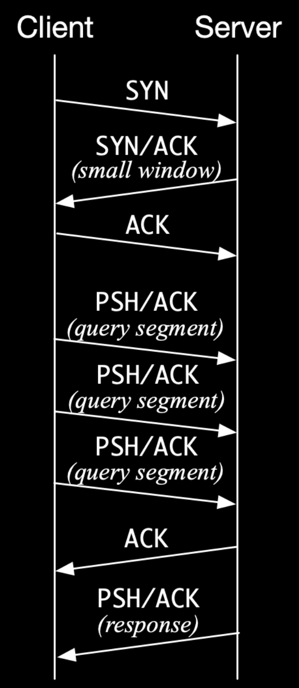
Client Server
SYN
SYN/ACK *(small window)*
ACK
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(bad ackno)*
SYN/ACK
RST
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(bad ackno)*
RST
SYN/ACK *(w/ load)*
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
FIN *(w/ load)*
SYN/ACK *(bad ackno)*
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
RST
SYN/ACK *(bad ackno)*
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*



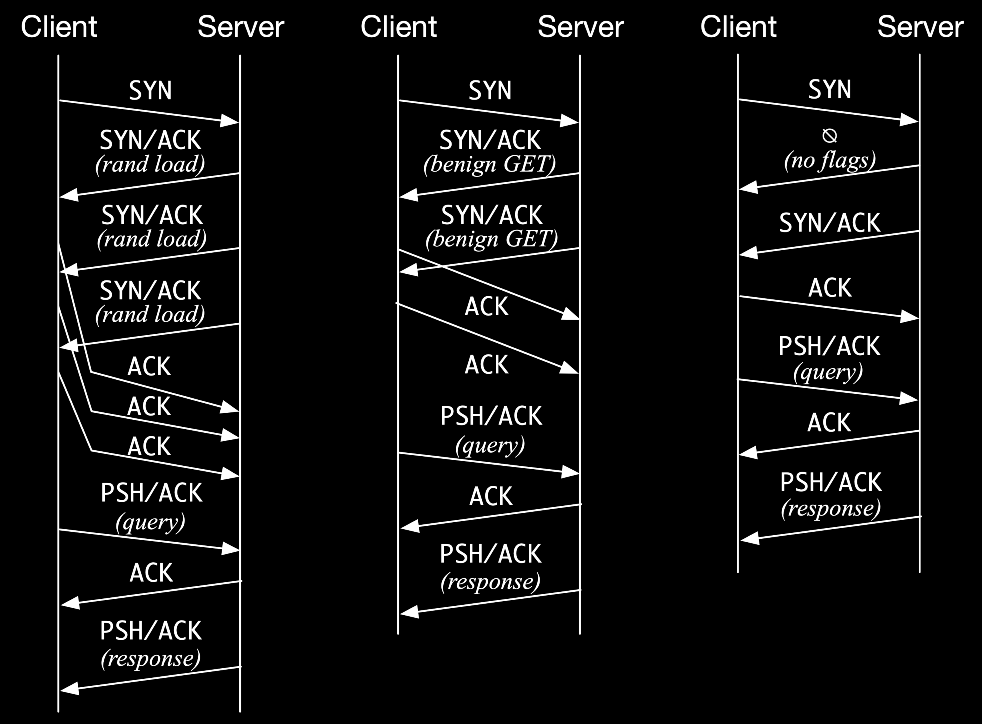
Client Server
SYN
SYN/ACK *(small window)*
ACK
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
ACK
PSH/ACK *(response)*

Iran/India

1 strategy

Kazakhstan

3 strategies



Client Server
SYN
SYN/ACK *(rand load)*
SYN/ACK *(rand load)*
SYN/ACK *(rand load)*
ACK
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(benign GET)*
SYN/ACK *(benign GET)*
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

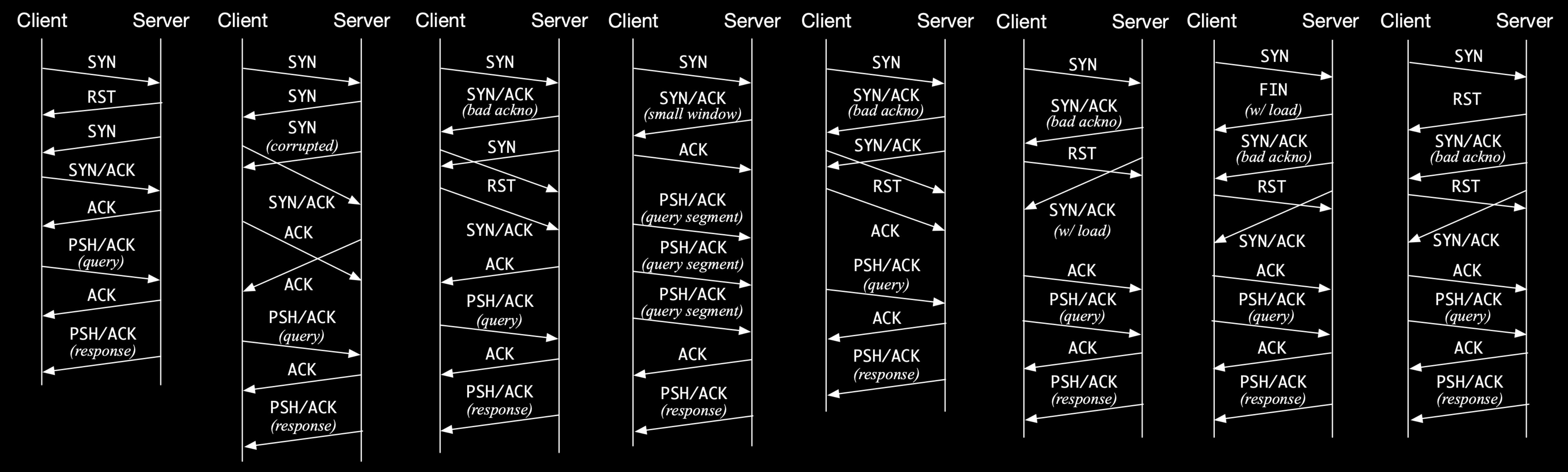
Client Server
SYN
⦸ *(no flags)*
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Server-side evasion “shouldn’t” work

Server-side results

China

8 strategies



Client Server
SYN
RST
SYN
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN
SYN *(corrupted)*
SYN/ACK
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(bad ackno)*
SYN
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(small window)*
ACK
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(bad ackno)*
SYN/ACK
RST
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

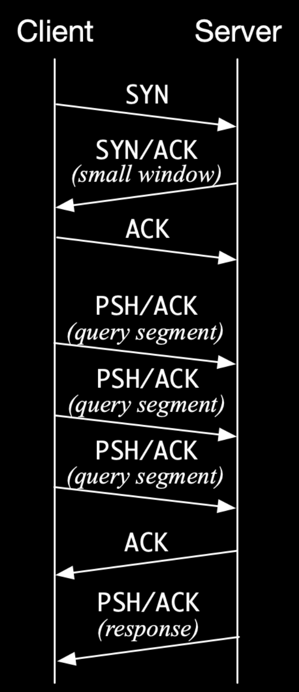
Client Server
SYN
SYN/ACK *(bad ackno)*
RST
SYN/ACK *(w/ load)*
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
FIN *(w/ load)*
SYN/ACK *(bad ackno)*
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
RST
SYN/ACK *(bad ackno)*
RST
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Iran/India

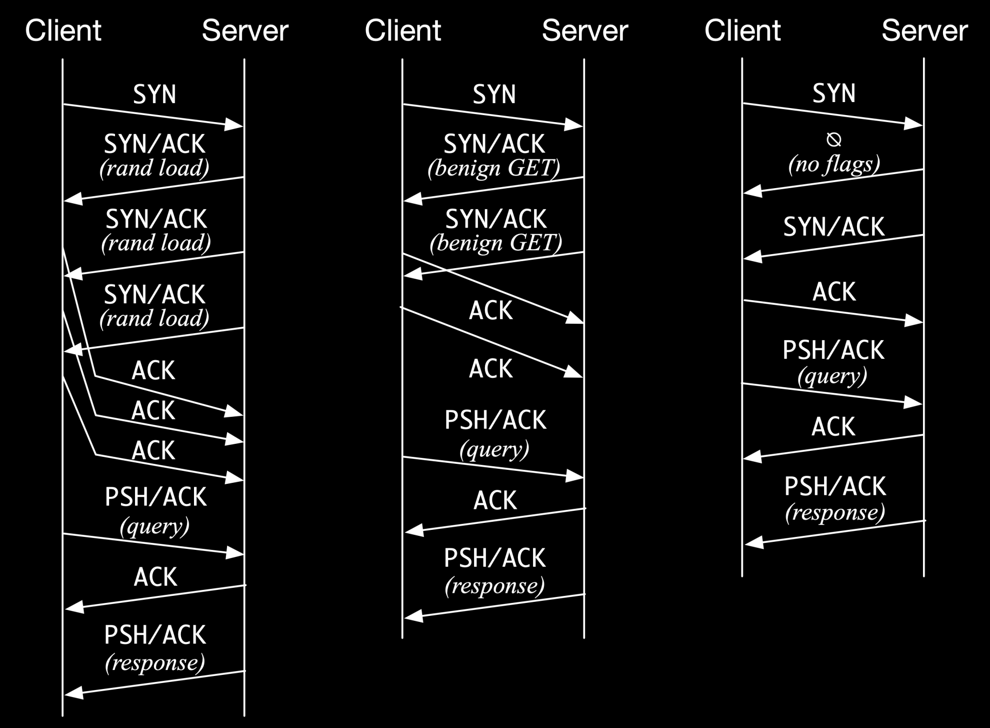
1 strategy



Client Server
SYN
SYN/ACK *(small window)*
ACK
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
PSH/ACK *(query segment)*
ACK
PSH/ACK *(response)*

Kazakhstan

3 strategies



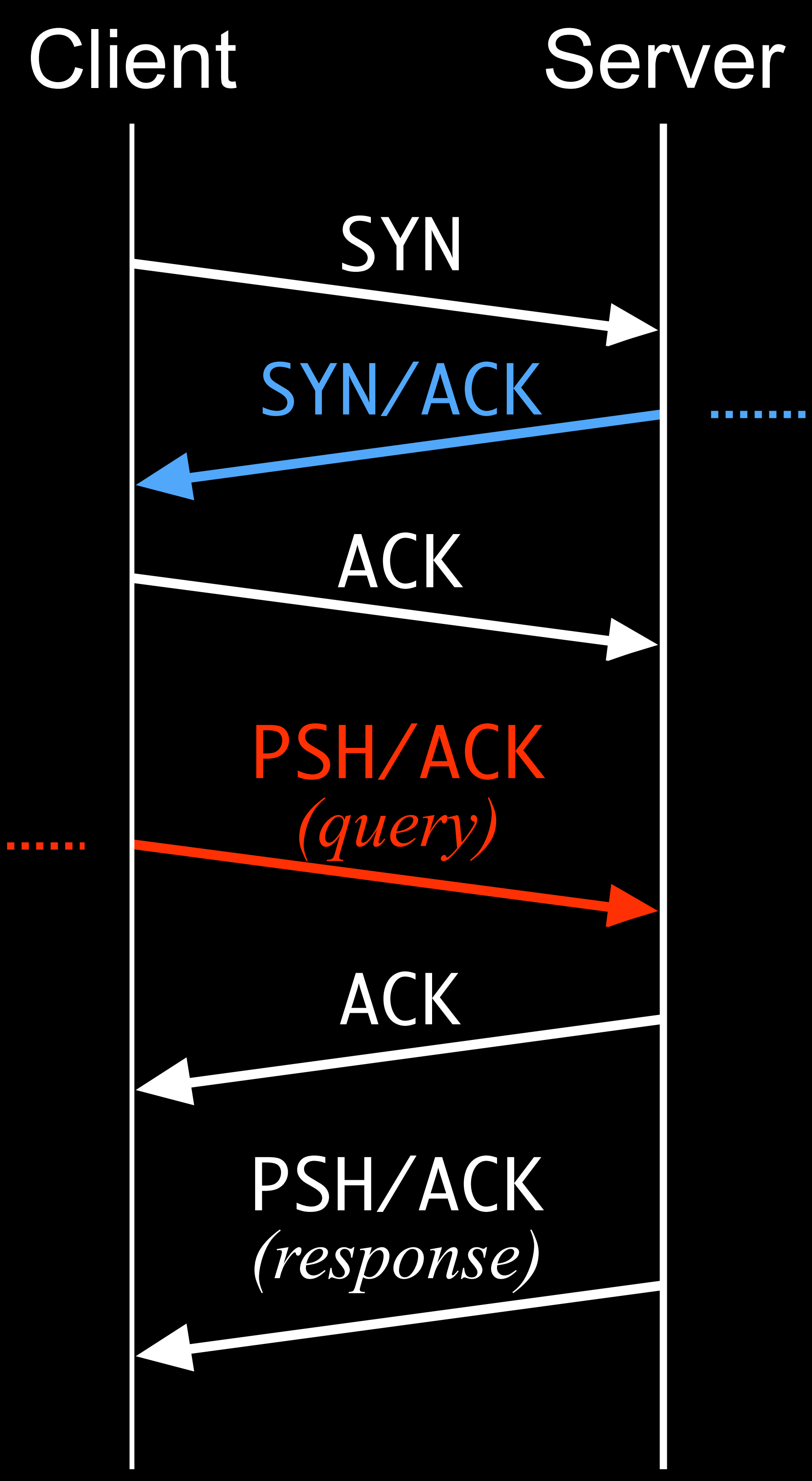
Client Server
SYN
SYN/ACK *(rand load)*
SYN/ACK *(rand load)*
SYN/ACK *(rand load)*
ACK
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
SYN/ACK *(benign GET)*
SYN/ACK *(benign GET)*
ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

Client Server
SYN
⦸ *(no flags)*
SYN/ACK
ACK
PSH/ACK *(query)*
ACK
PSH/ACK *(response)*

None of these require *any* client-side deployment

Server-side evasion “shouldn’t” work

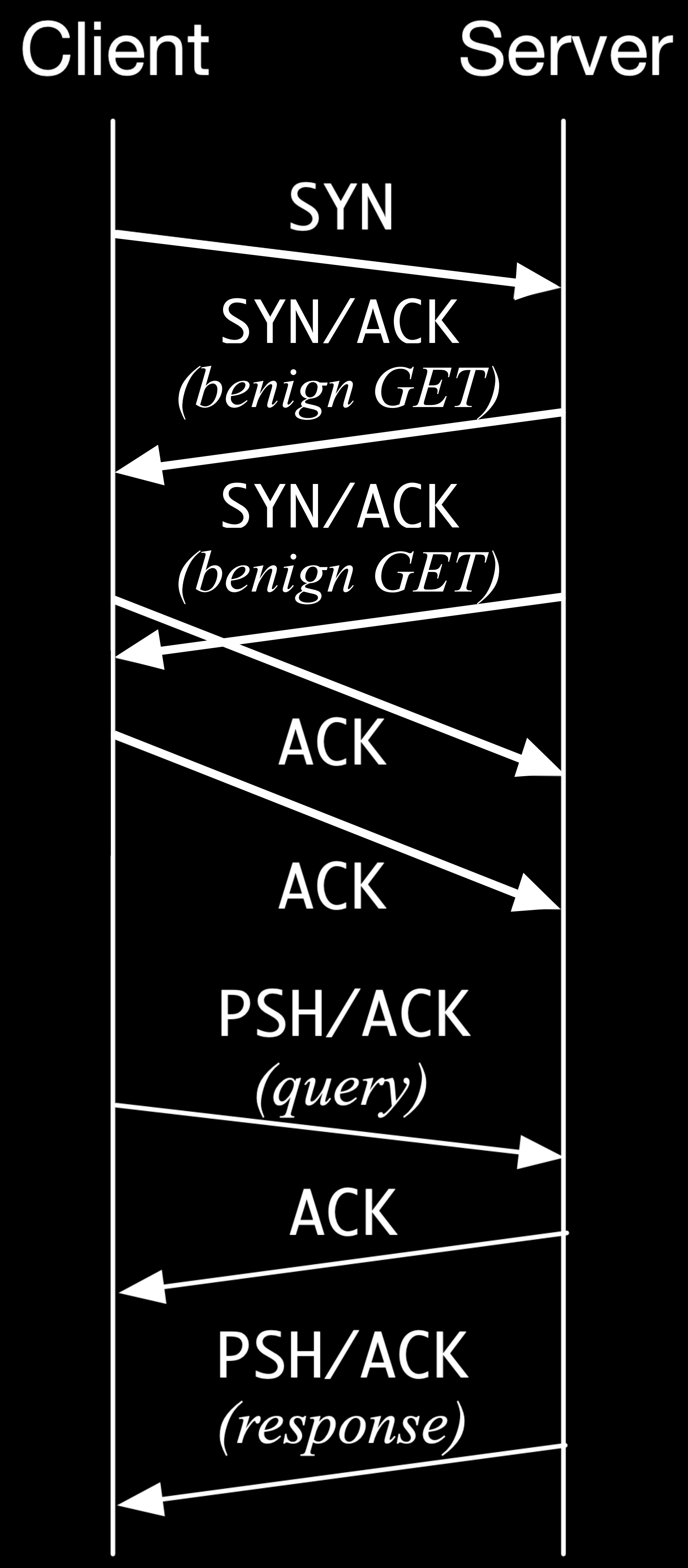


All a server does before client is censored

Censored keyword

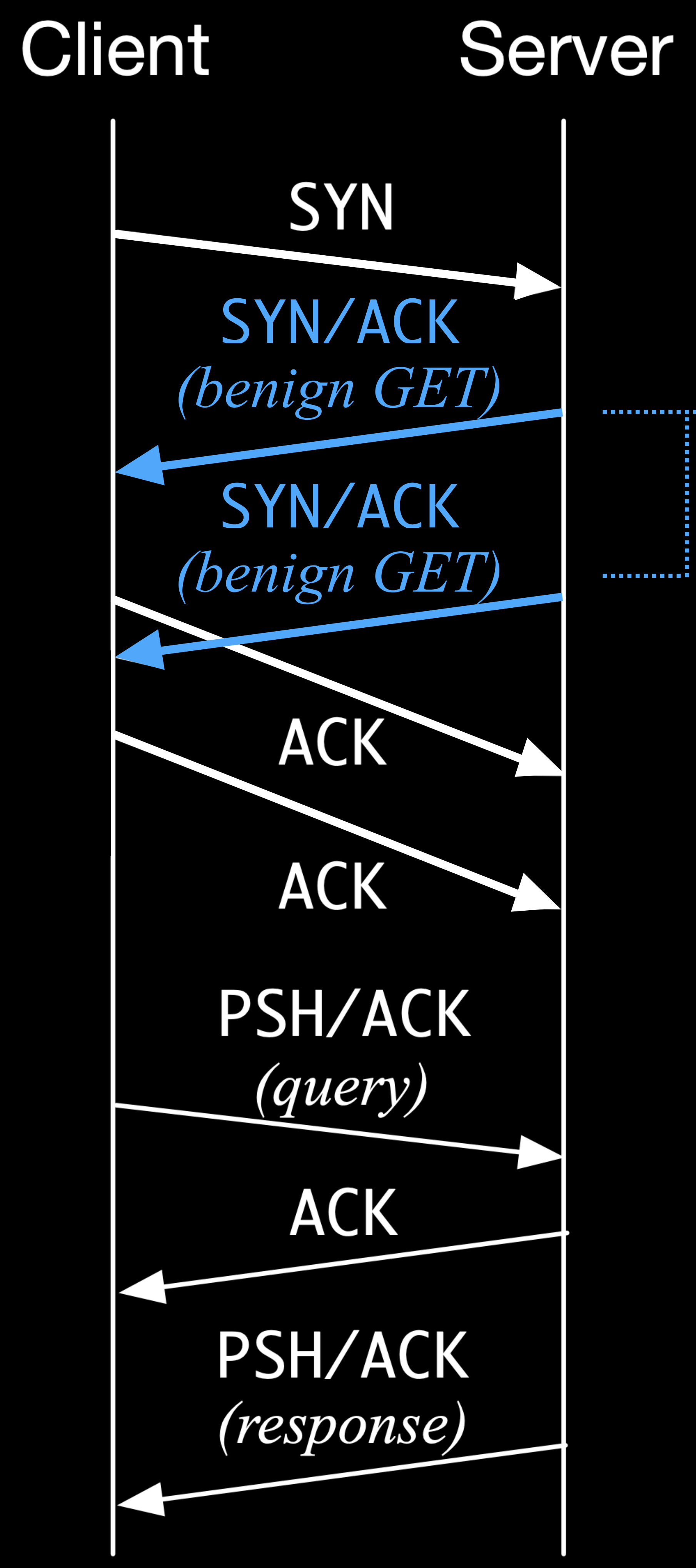
Server-side evasion results

🇰🇿 Double benign-GETs



Server-side evasion results

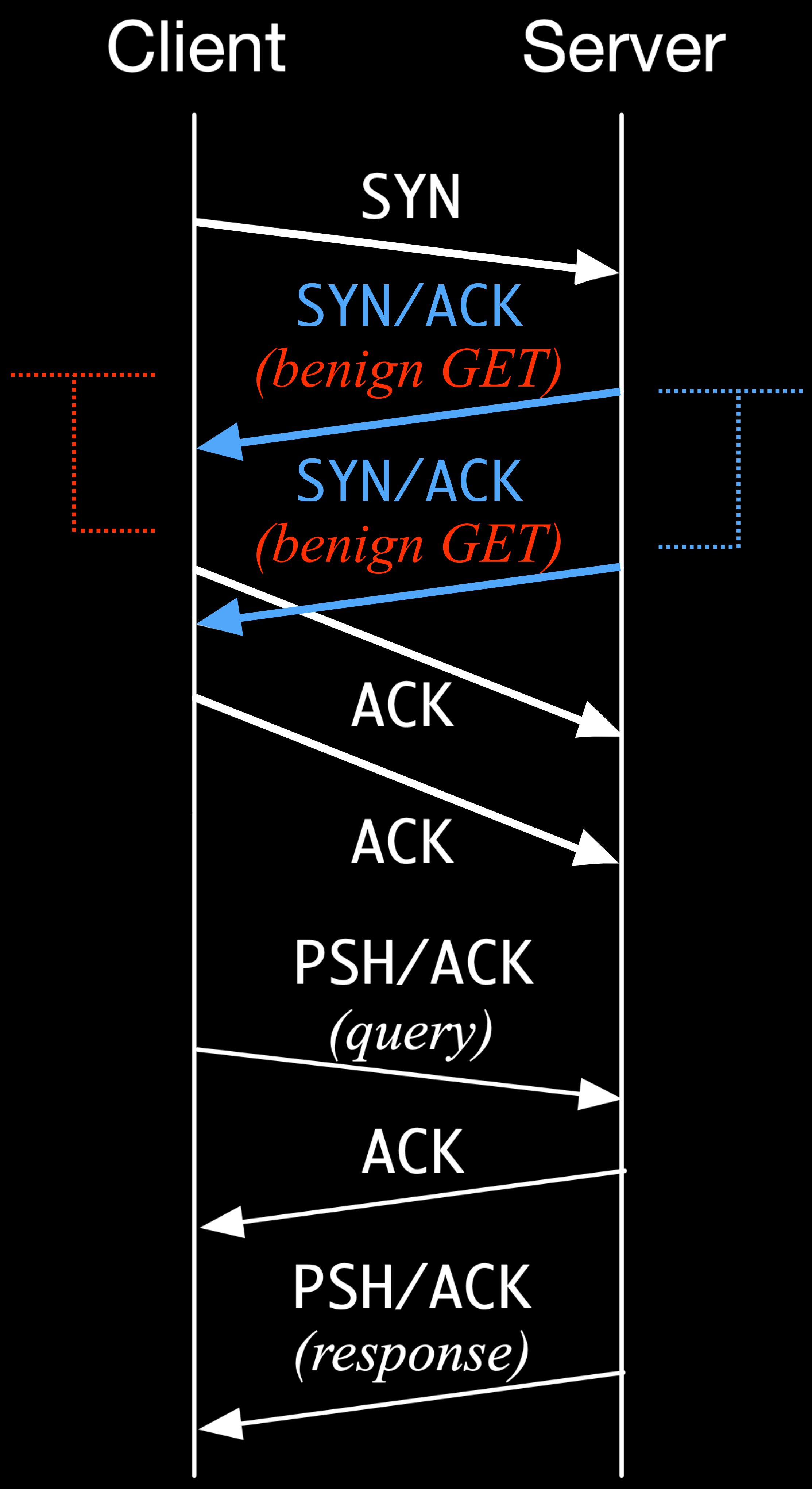
Double benign-GETs



Server sends uncensored GETs inside *two* SYN/ACKs

Server-side evasion results

Double benign-GETs

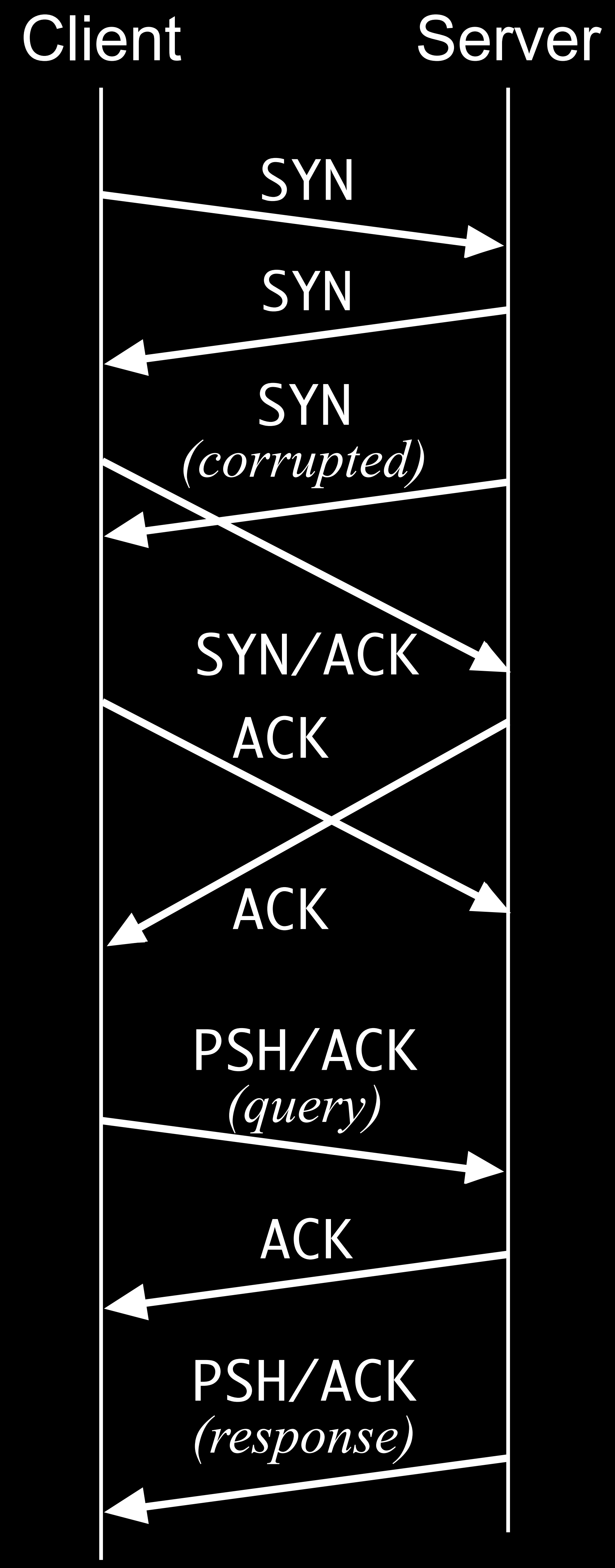


Censor confuses connection direction

Server sends uncensored GETs inside *two* SYN/ACKs

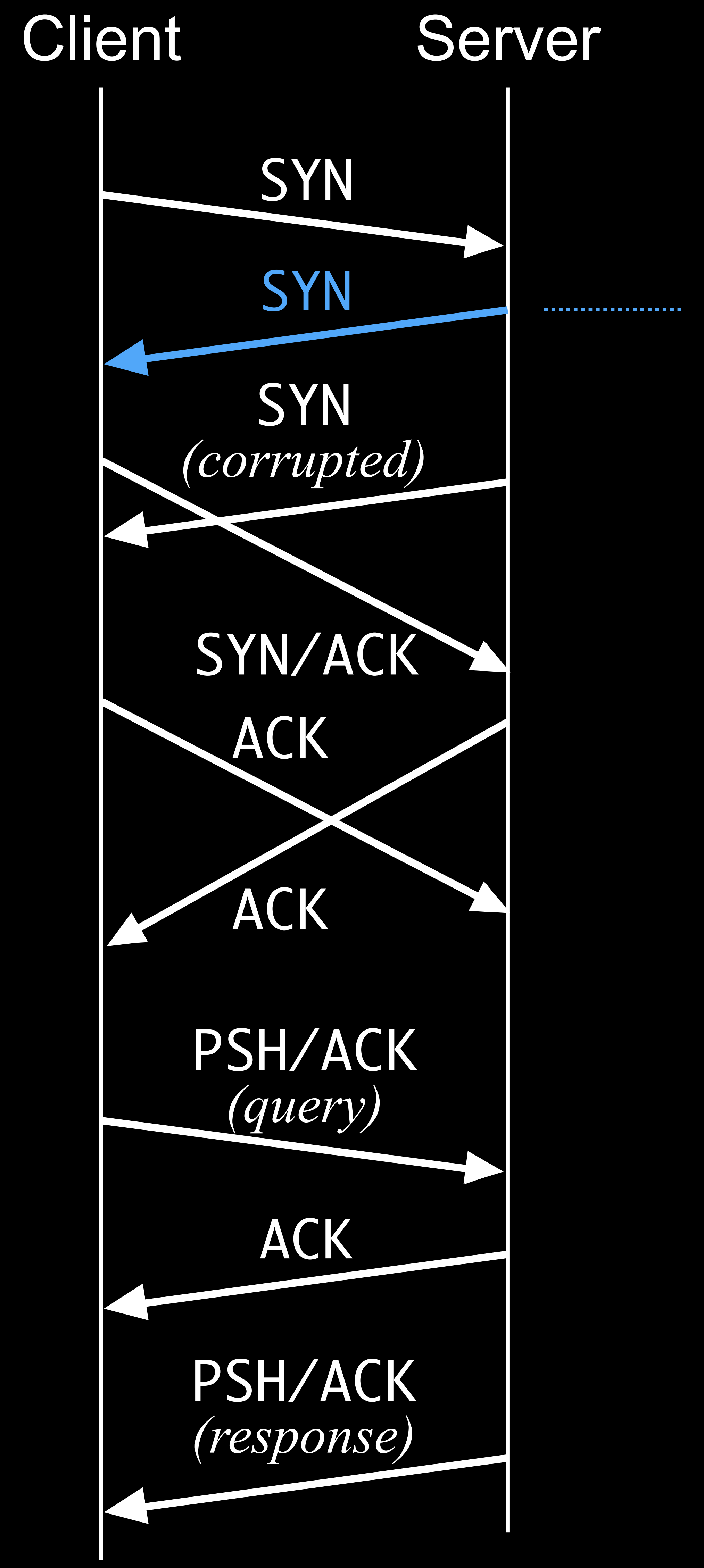
Server-side evasion results

Simultaneous-open-based desynchronization



Server-side evasion results

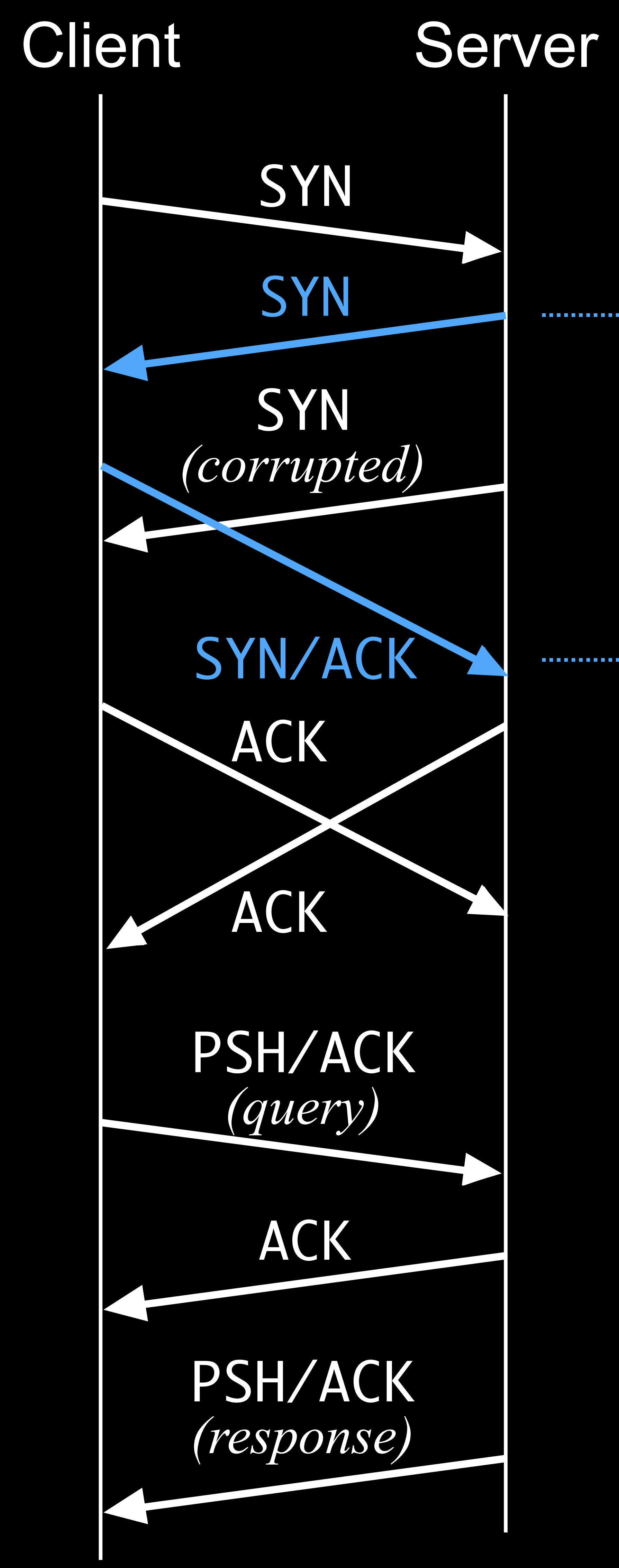
Simultaneous-open-based desynchronization



TCP simultaneous open

Server-side evasion results

Simultaneous-open-based desynchronization

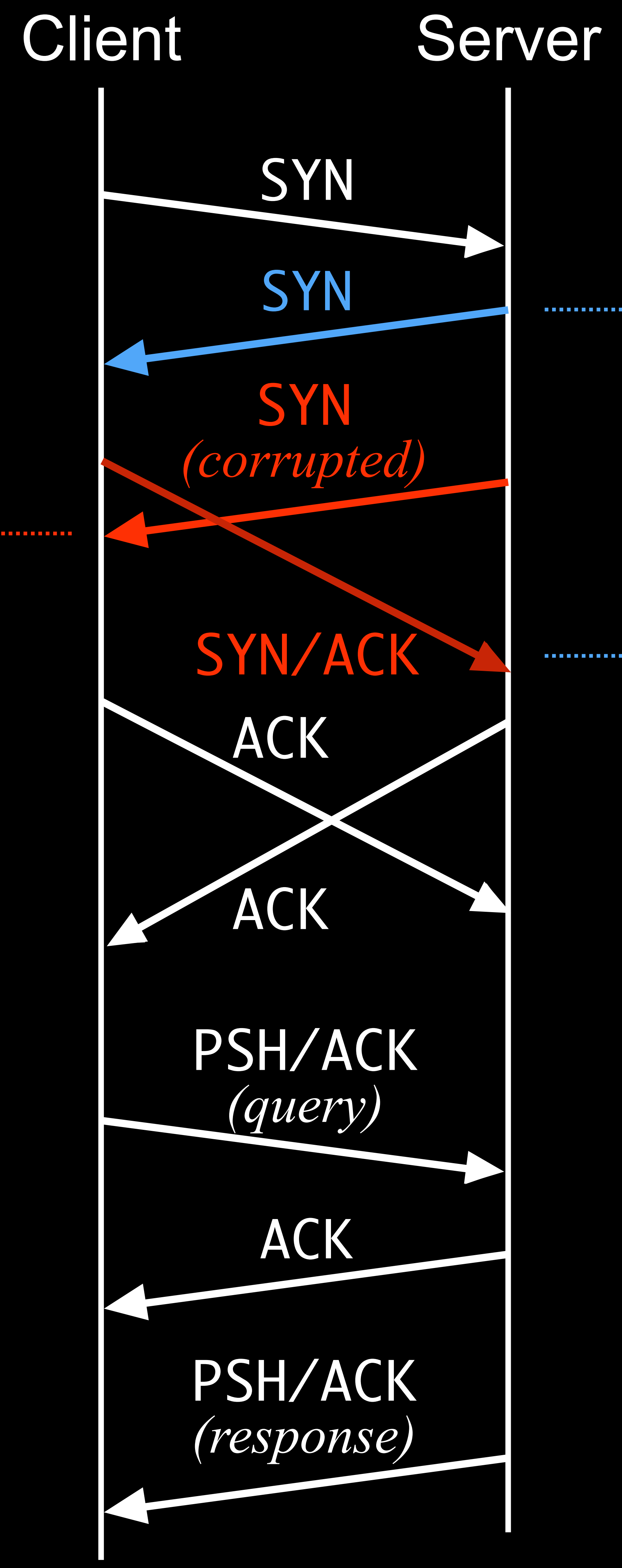


TCP simultaneous open

Client sends a SYN/ACK

Server-side evasion results

Simultaneous-open-based desynchronization



TCP simultaneous open

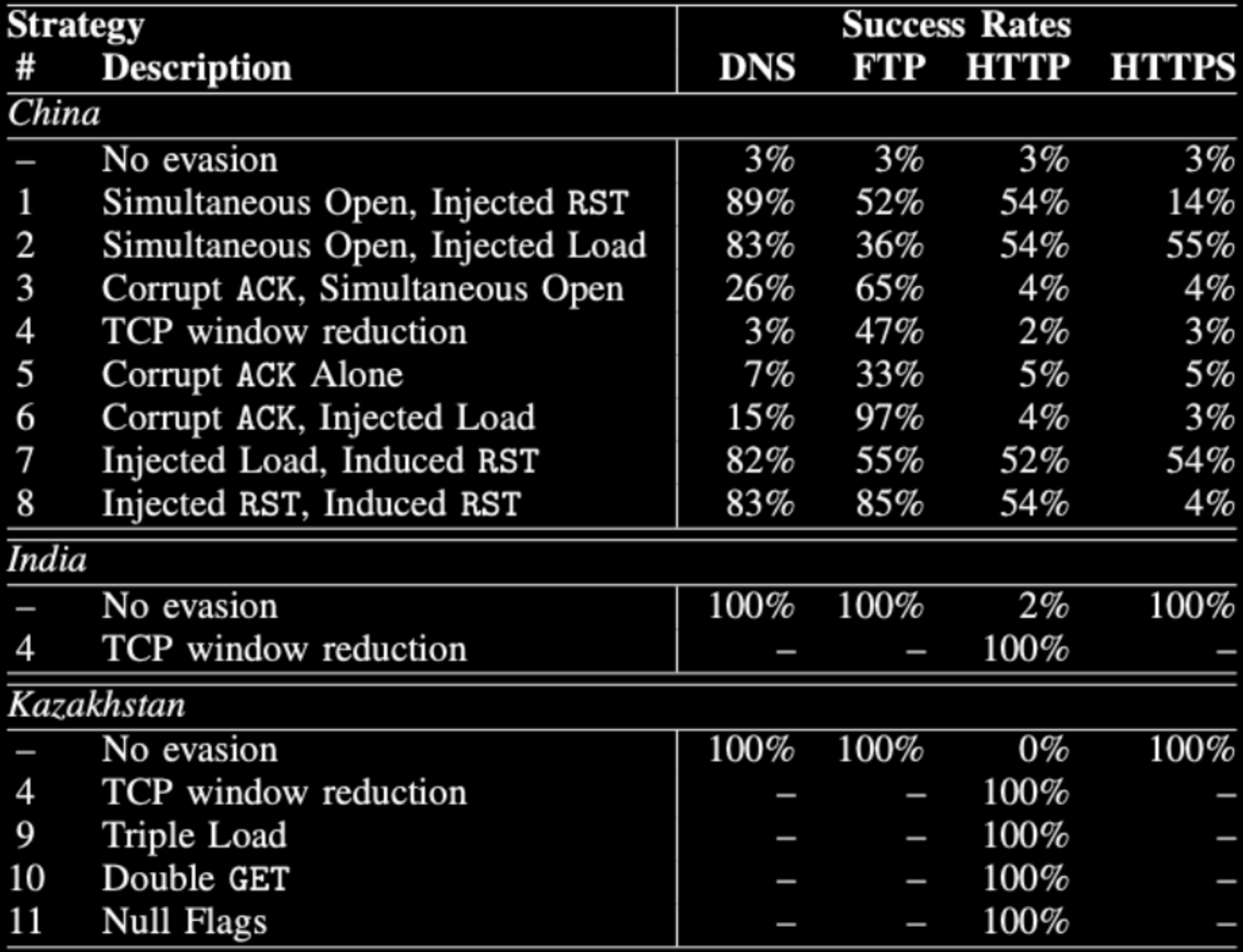
Censor de-synchronizes

Client sends a SYN/ACK

Demo

UNIVERSITY OF
MARYLAND

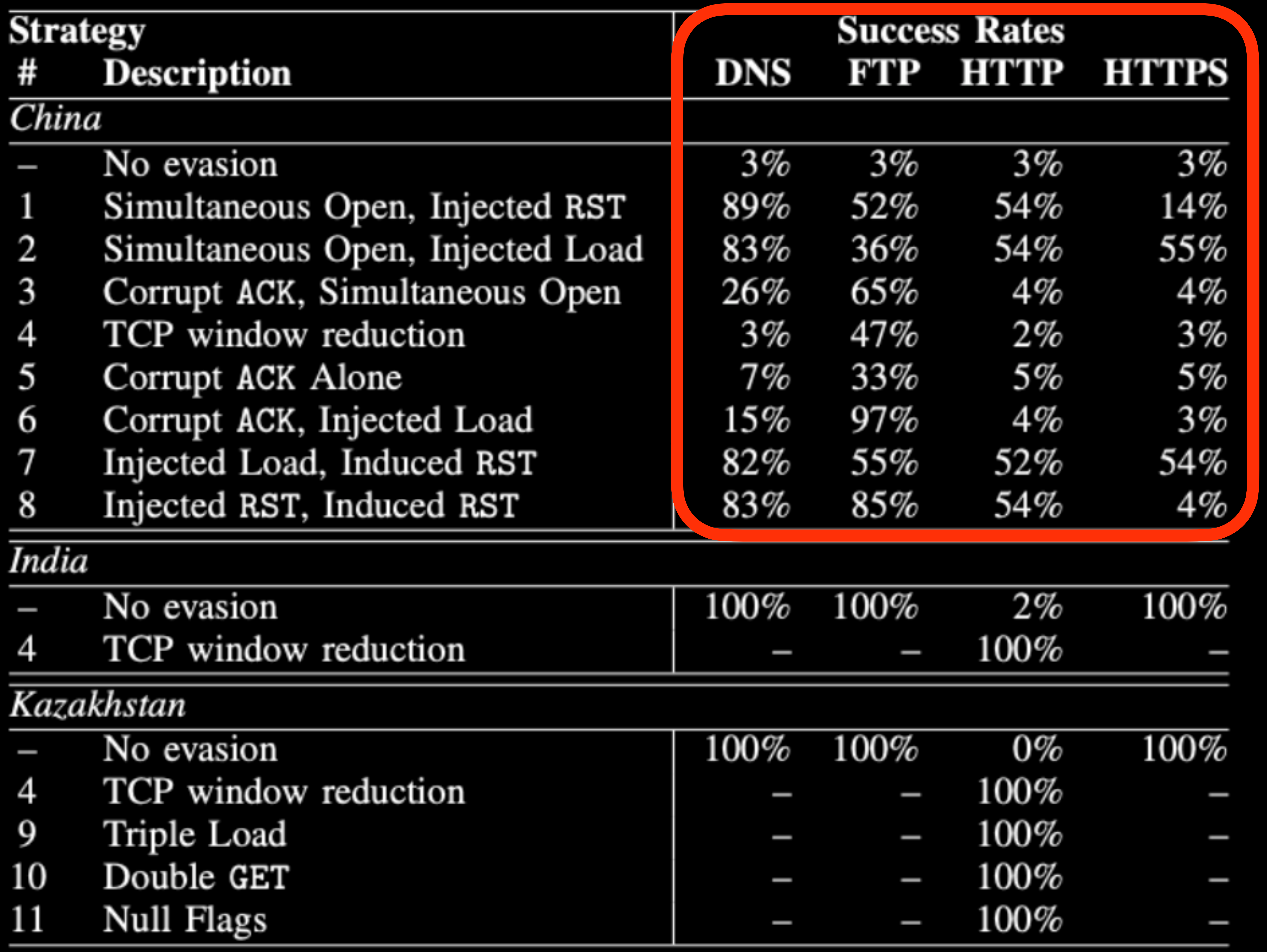
New Model for Chinese Censorship



Strategy		Success Rates			
#	Description	DNS	FTP	HTTP	HTTPS
China					
–	No evasion	3%	3%	3%	3%
1	Simultaneous Open, Injected RST	89%	52%	54%	14%
2	Simultaneous Open, Injected Load	83%	36%	54%	55%
3	Corrupt ACK, Simultaneous Open	26%	65%	4%	4%
4	TCP window reduction	3%	47%	2%	3%
5	Corrupt ACK Alone	7%	33%	5%	5%
6	Corrupt ACK, Injected Load	15%	97%	4%	3%
7	Injected Load, Induced RST	82%	55%	52%	54%
8	Injected RST, Induced RST	83%	85%	54%	4%
India					
–	No evasion	100%	100%	2%	100%
4	TCP window reduction	–	–	100%	–
Kazakhstan					
–	No evasion	100%	100%	0%	100%
4	TCP window reduction	–	–	100%	–
9	Triple Load	–	–	100%	–
10	Double GET	–	–	100%	–
11	Null Flags	–	–	100%	–

All of the server-side strategies operate strictly during the TCP 3-way handshake

New Model for Chinese Censorship



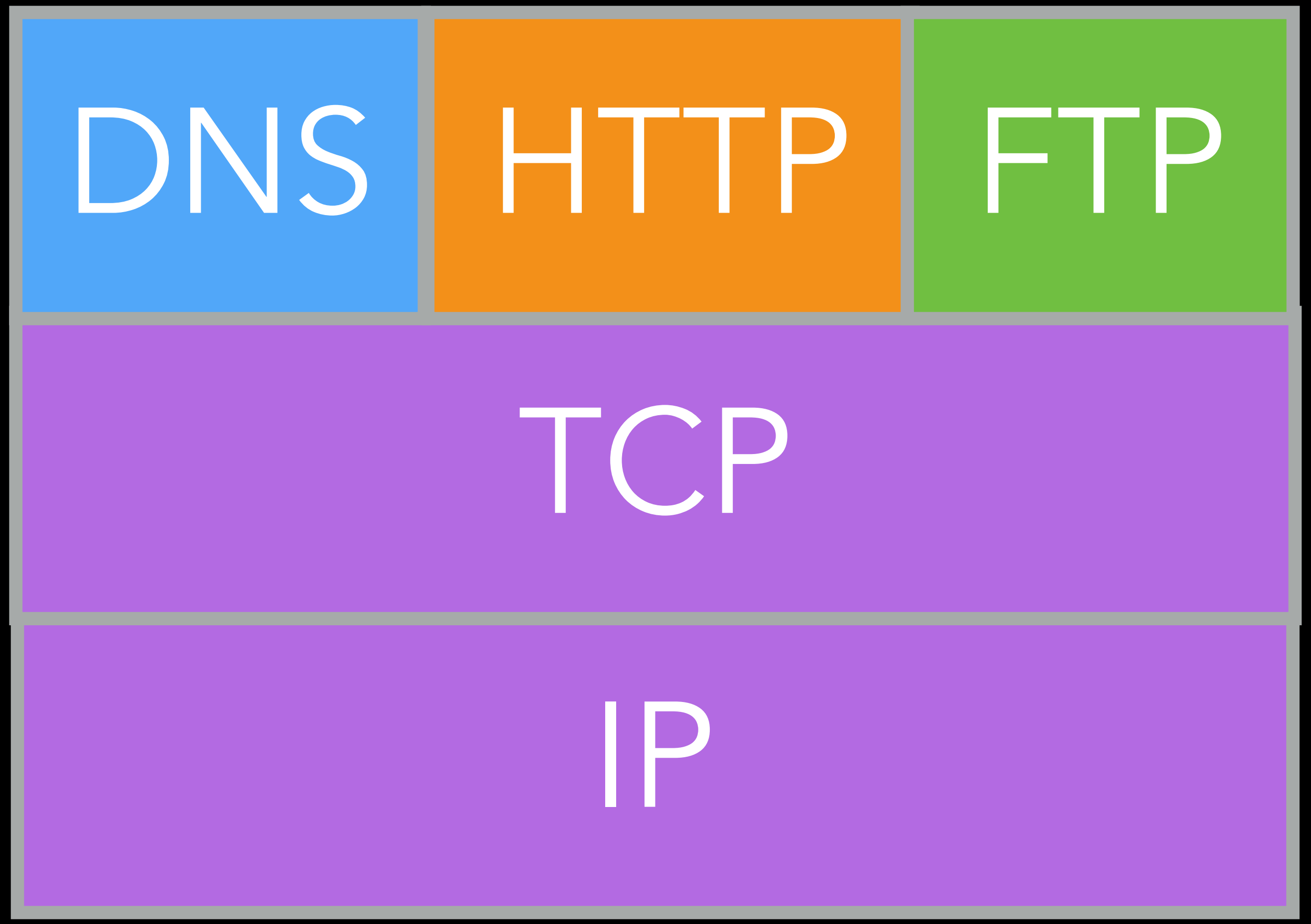
Strategy		Success Rates			
#	Description	DNS	FTP	HTTP	HTTPS
China					
–	No evasion	3%	3%	3%	3%
1	Simultaneous Open, Injected RST	89%	52%	54%	14%
2	Simultaneous Open, Injected Load	83%	36%	54%	55%
3	Corrupt ACK, Simultaneous Open	26%	65%	4%	4%
4	TCP window reduction	3%	47%	2%	3%
5	Corrupt ACK Alone	7%	33%	5%	5%
6	Corrupt ACK, Injected Load	15%	97%	4%	3%
7	Injected Load, Induced RST	82%	55%	52%	54%
8	Injected RST, Induced RST	83%	85%	54%	4%
India					
–	No evasion	100%	100%	2%	100%
4	TCP window reduction	–	–	100%	–
Kazakhstan					
–	No evasion	100%	100%	0%	100%
4	TCP window reduction	–	–	100%	–
9	Triple Load	–	–	100%	–
10	Double GET	–	–	100%	–
11	Null Flags	–	–	100%	–

All of the server-side strategies operate strictly during the TCP 3-way handshake

So why are different applications affected differently in China?

New Model for Chinese Censorship

Sane



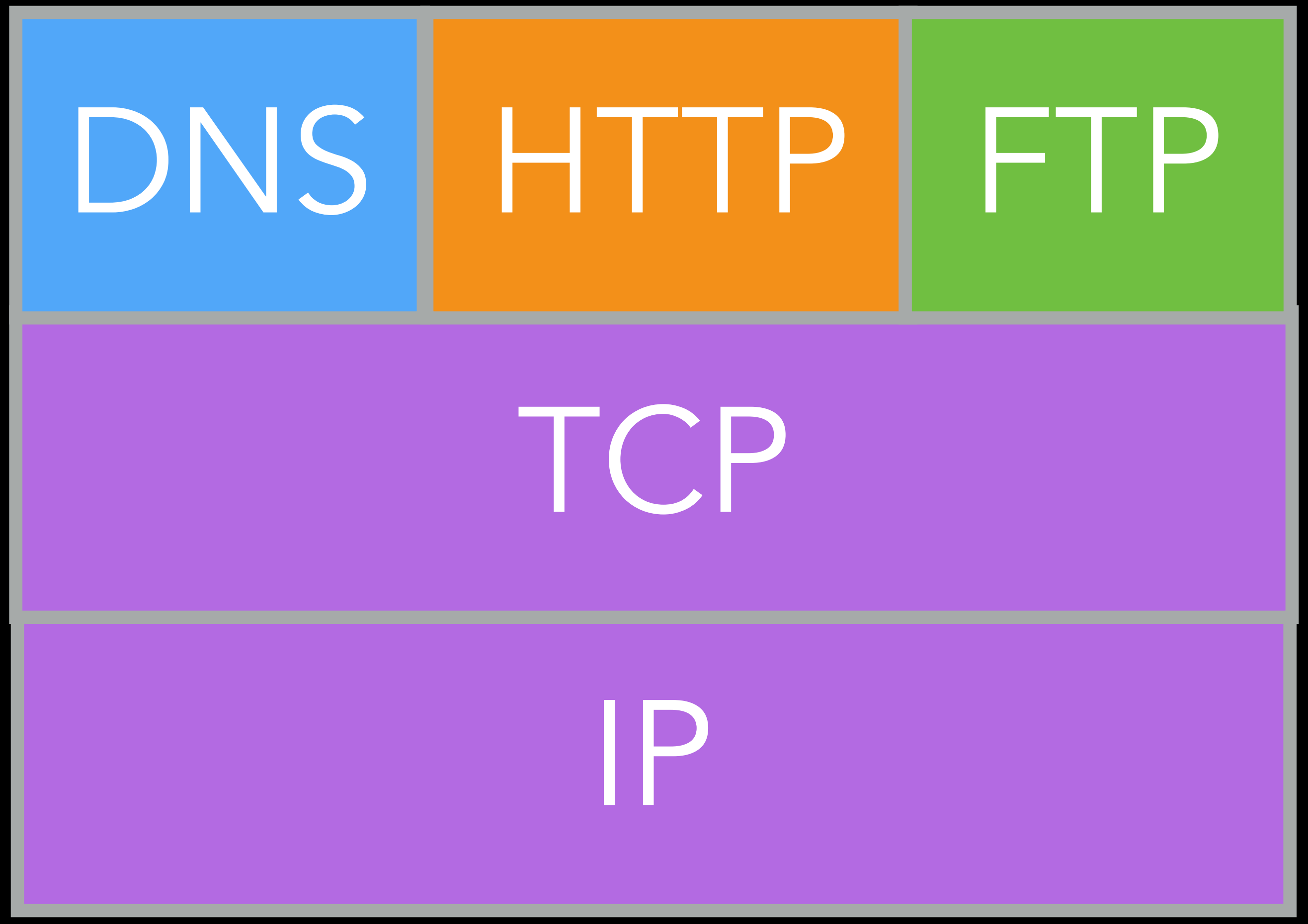
DNS HTTP FTP

TCP

IP

New Model for Chinese Censorship

Sane

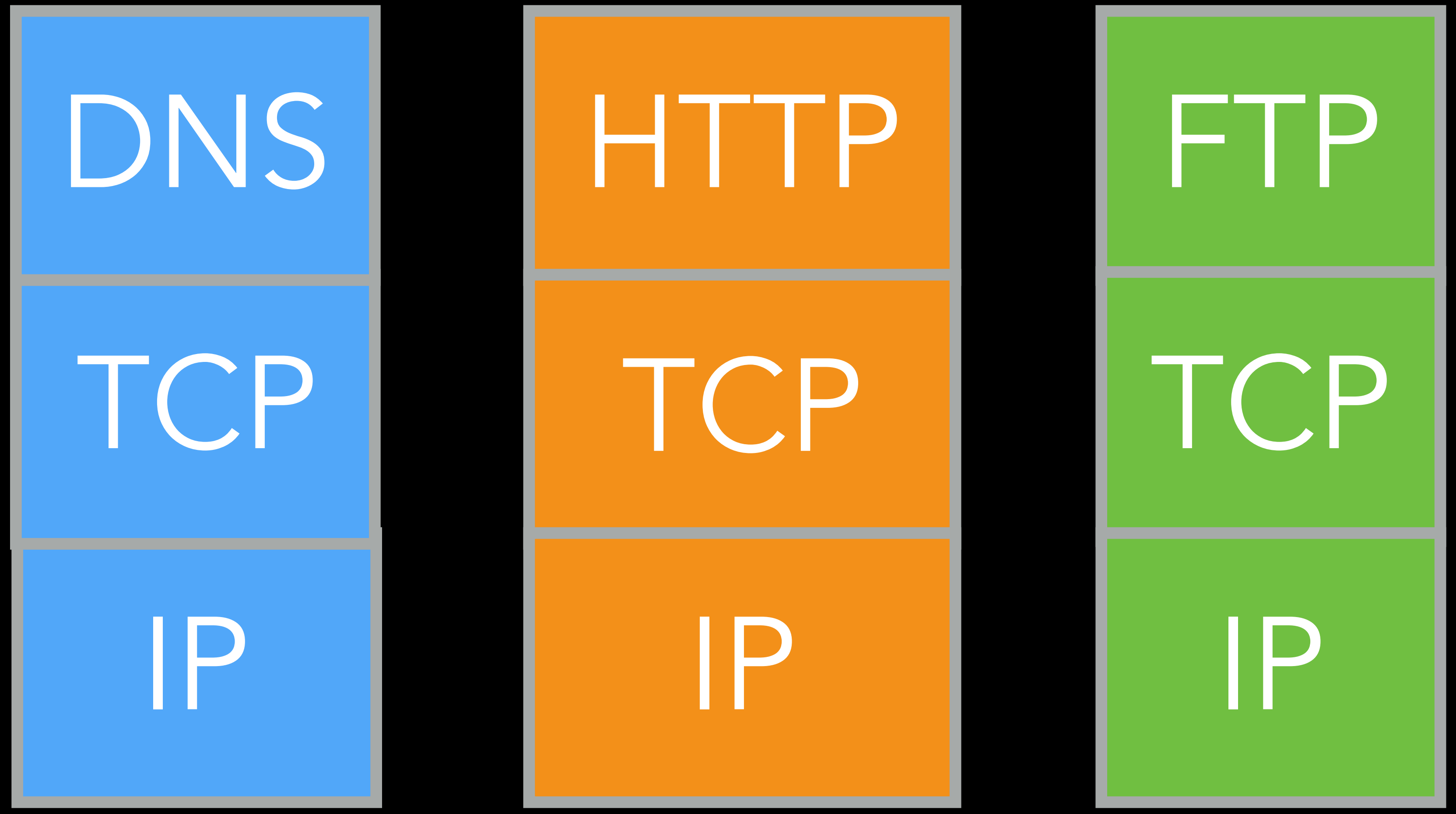


DNS HTTP FTP

TCP

IP

Apparently what’s happening



DNS TCP IP

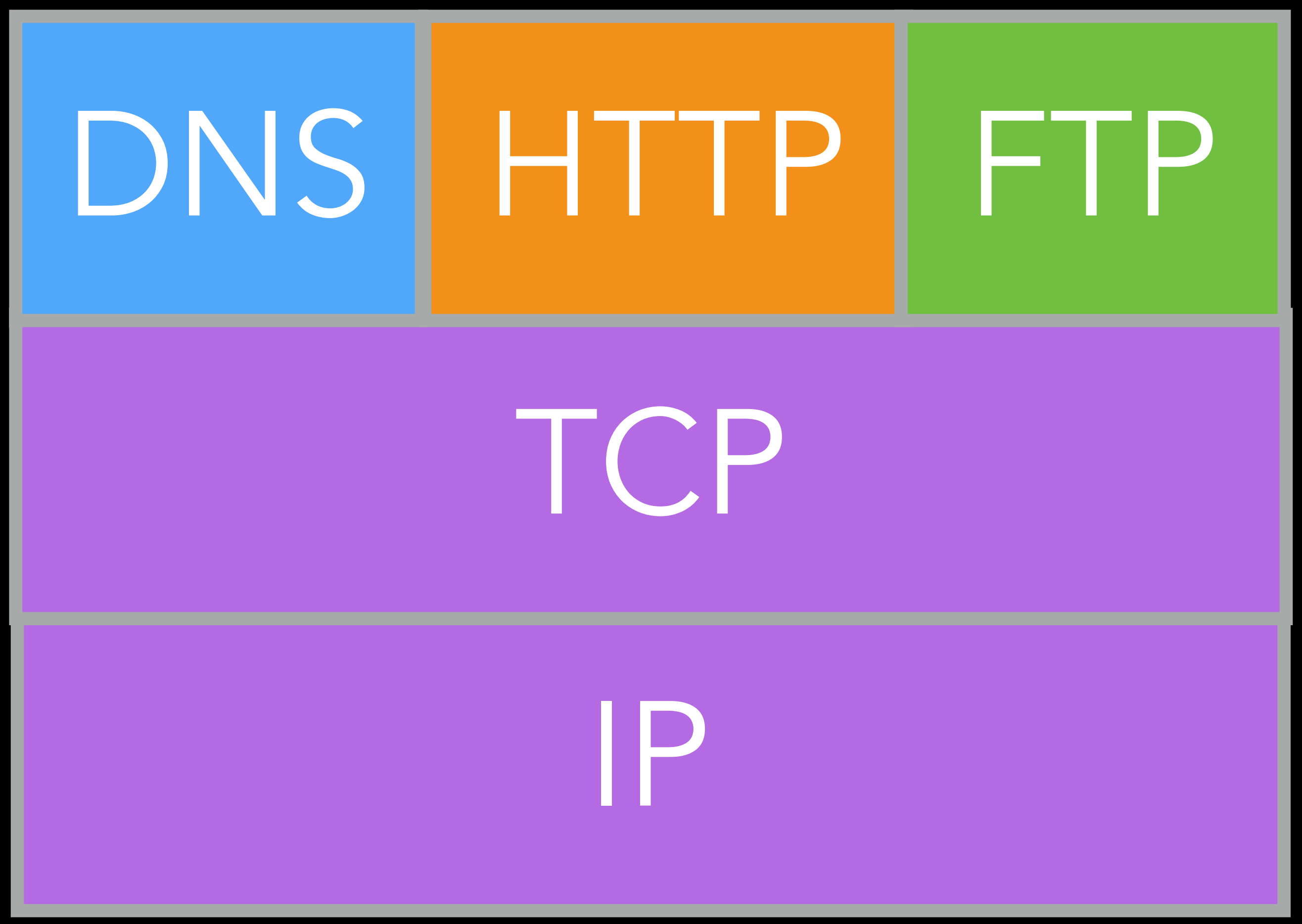
HTTP TCP IP

FTP TCP IP

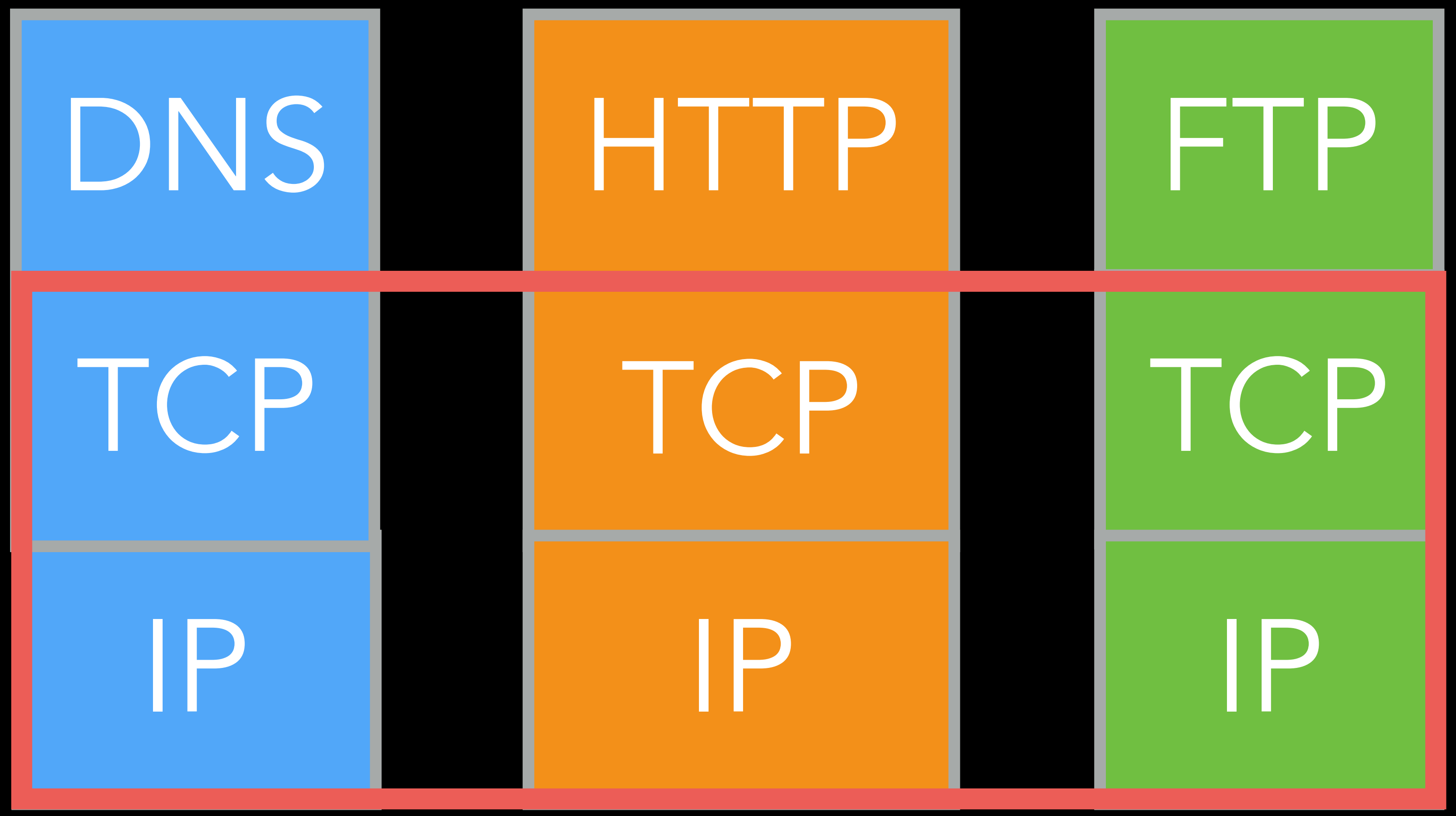
They appear to be running multiple censoring middleboxes in parallel

New Model for Chinese Censorship

Sane



Apparently what's happening



They appear to be running multiple censoring middleboxes in parallel

New Model for Chinese Censorship

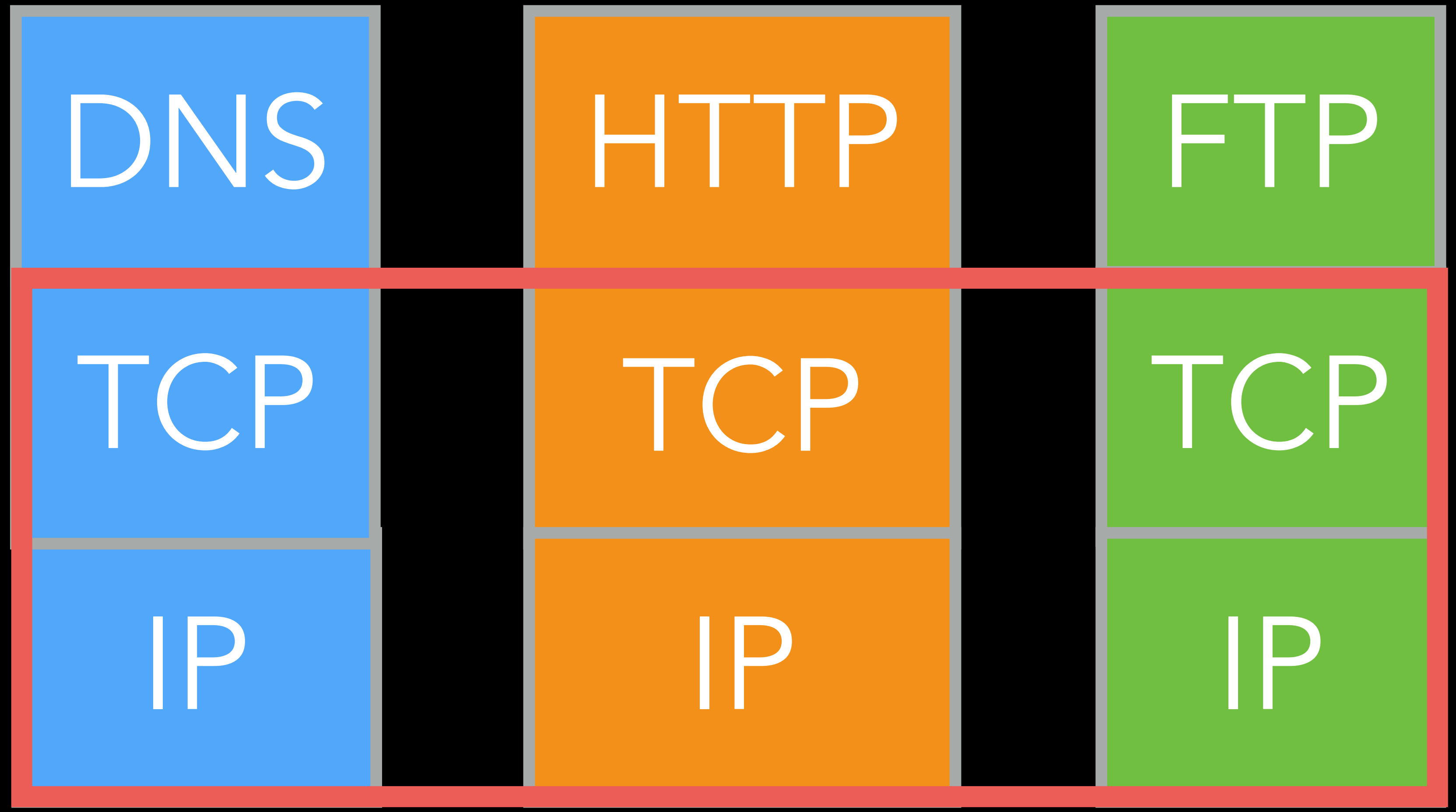
How does the censor know which one to apply to a connection?

Not port number

They appear to apply protocol fingerprinting

Basic protocol confusion could be highly effective

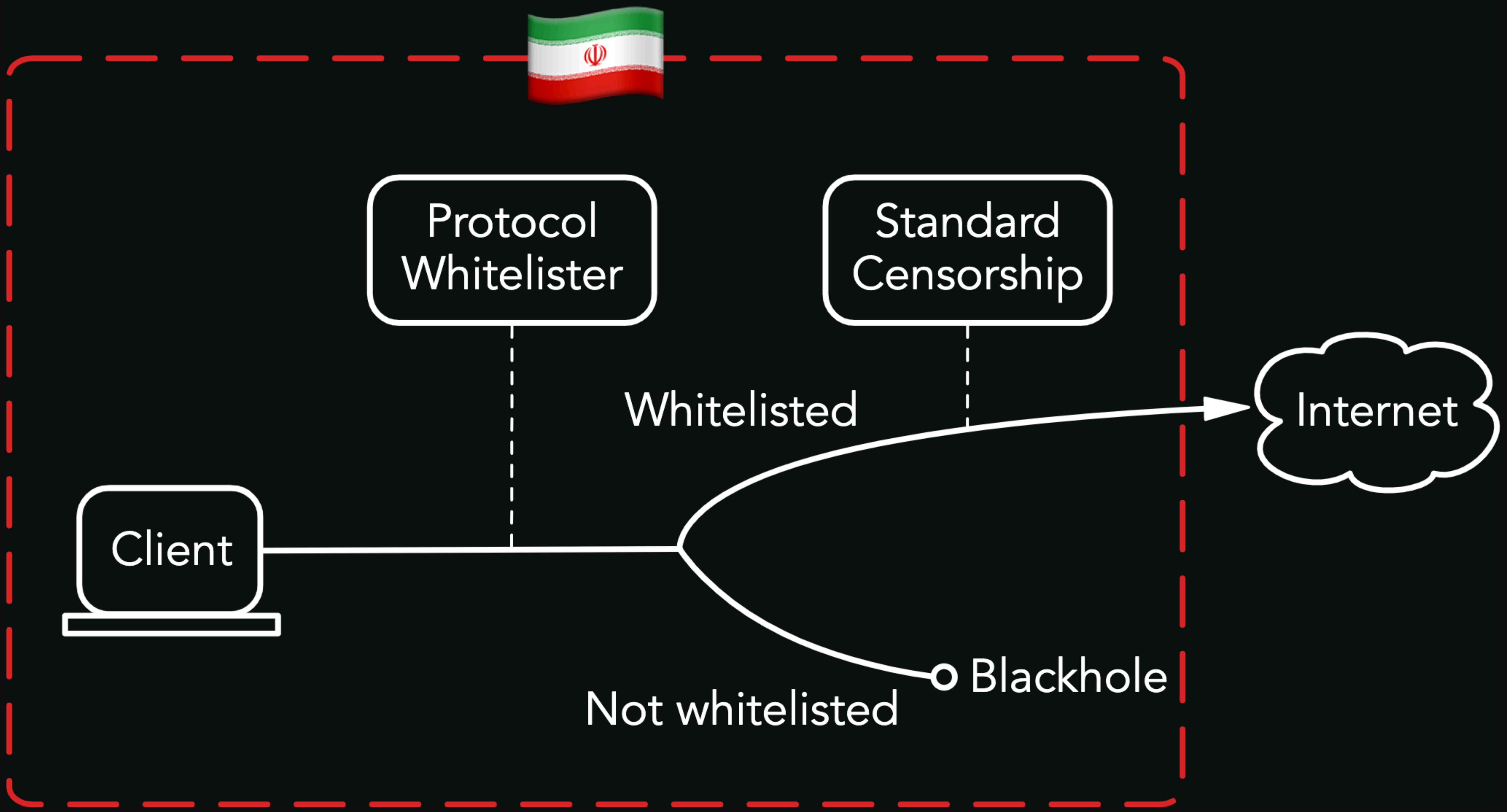
Apparently what's happening



They appear to be running multiple censoring middleboxes in parallel

Geneva defeats censorship-in-depth

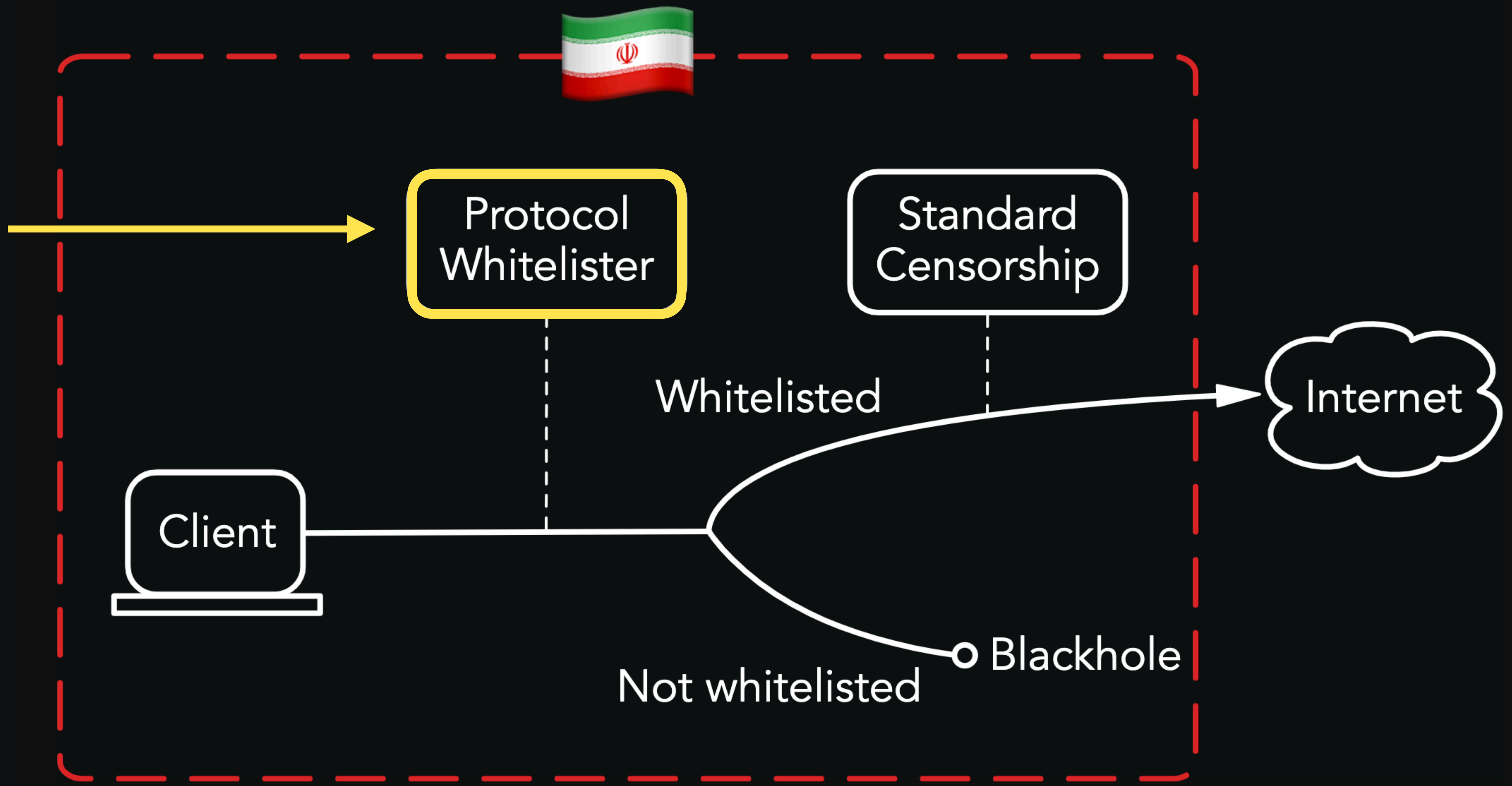
February 2020: Iran launched a new system: a protocol filter



Geneva defeats censorship-in-depth

February 2020: Iran launched a new system: a protocol filter

Censors connections that do not match protocol fingerprints

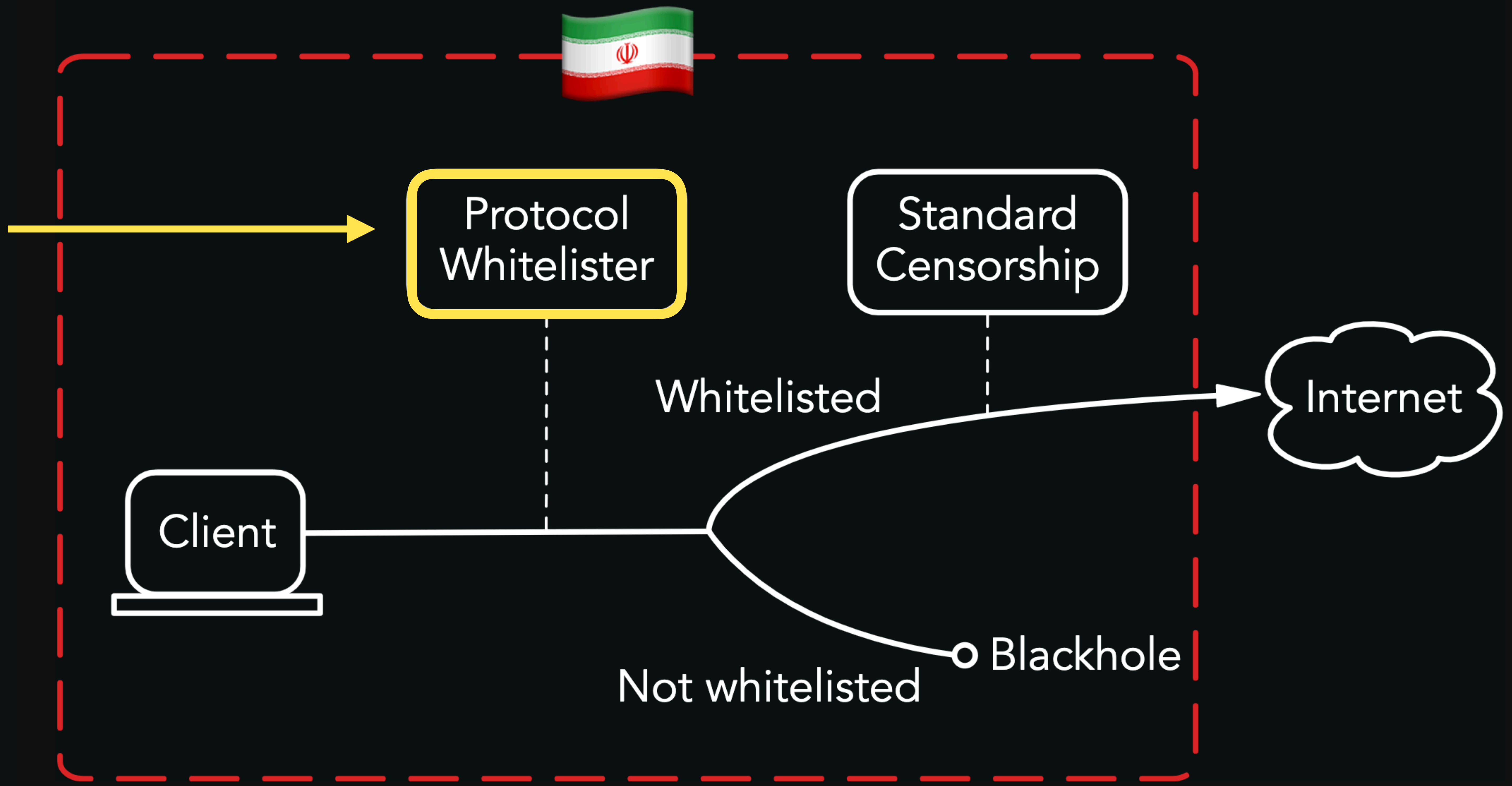


Geneva defeats censorship-in-depth

February 2020: Iran launched a new system: a protocol filter

Censors connections that do not match protocol fingerprints

Those that do match are then subjected to standard censorship

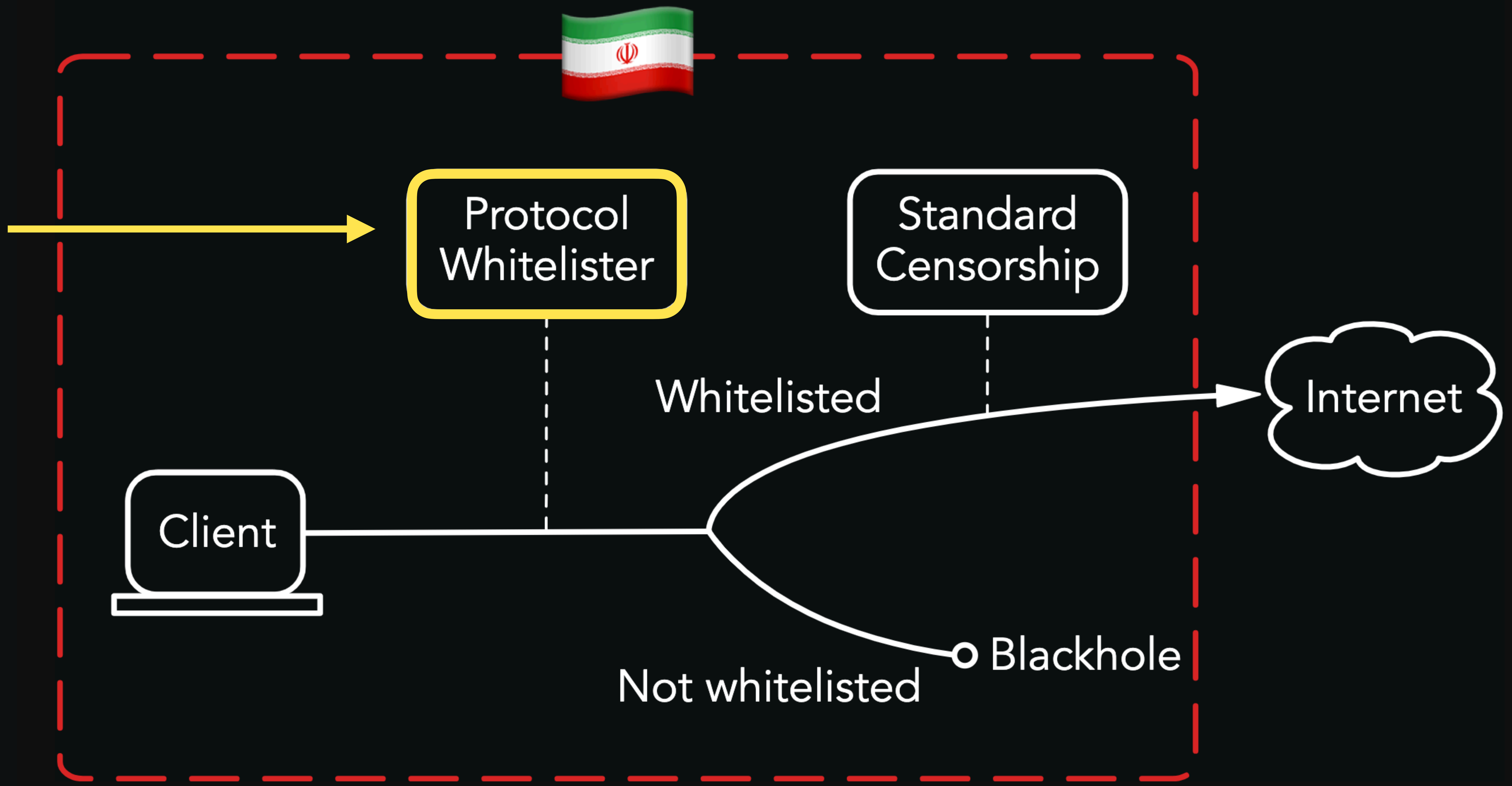


Geneva defeats censorship-in-depth

February 2020: Iran launched a new system: a protocol filter

Censors connections that do not match protocol fingerprints

Those that do match are then subjected to standard censorship



Geneva discovered 3 strategies to evade Iran's filter

Automating the arms race

⏩ AI has the potential to fast-forward the arms race *for both sides*

Automating the arms race

⏩ AI has the potential to fast-forward the arms race *for both sides*

Bugs in implementation	Easy for censors to fix the low-hanging fruit
Gaps in logic	Harder for censors to fix systemic issues

Automating the arms race

AI has the potential to fast-forward the arms race *for both sides*

Bugs in implementation	Easy for censors to fix the low-hanging fruit
Gaps in logic	Harder for censors to fix systemic issues

What is the *logical conclusion* of the arms race?

Geneva Team

Kevin Bock
Louis-Henri Merino
Tania Arya
Daniel Liscinsky
Regina Pogosian
Yair Fax

George Hughey
Kyle Reese
Jasraj Singh
Kyle Hurley
Michael Harrity

Dave Levin

UNIVERSITY OF MARYLAND

Xiao Qiang

Berkeley SCHOOL OF INFORMATION

Evolving censorship evasion

Geneva

Genetic Evasion

Client-side & Server-side

Has found dozens of strategies

Quickly discovers new strategies

Gives the advantage to evaders

Geneva code and website geneva.cs.umd.edu