
AWS
re:Invent

NET402

Elastic Load Balancing: A year of innovations

Jon Zobrist

SysDev Manager

ELB Customer Success

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Intro – ELB launches recap

ALB – gRPC

HTTP DeSync Mitigation Mode

GWLB – Gateway Load Balancer



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Elastic Load Balancing new features

- UDP listeners on NLB support for IP targets / AWS Fargate
- ALPN on NLB TLS listeners
- Weighted target groups
- Kubernetes ingress controller
- IPv6 on NLB
- ALB on AWS Outposts
- WAF fail open for ALB



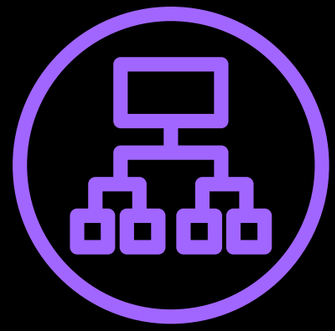


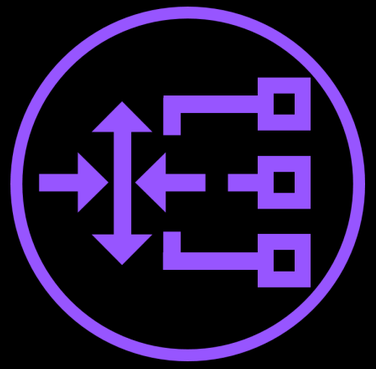




Elastic Load Balancing new features

- gRPC & HTTP/2 to targets
- HTTP DeSync mitigation mode
- Gateway Load Balancer





gRPC and HTTP/2 to targets

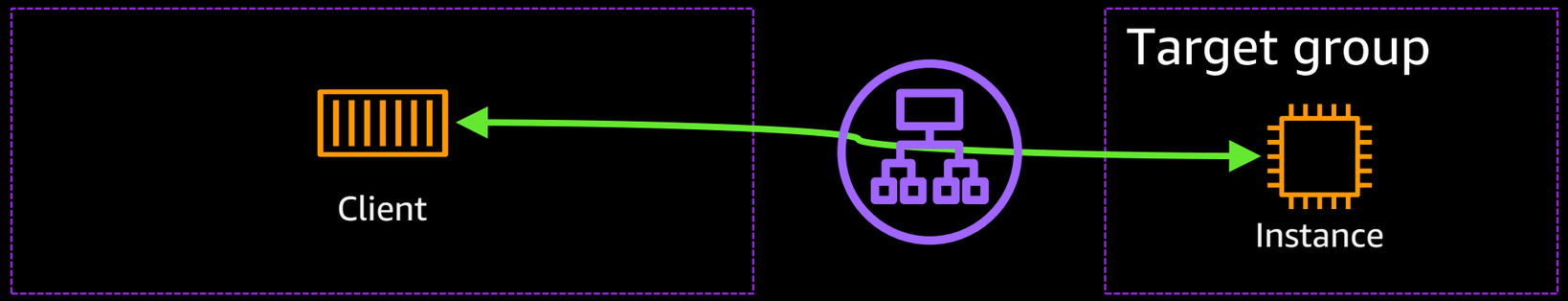


AWS re:Invent

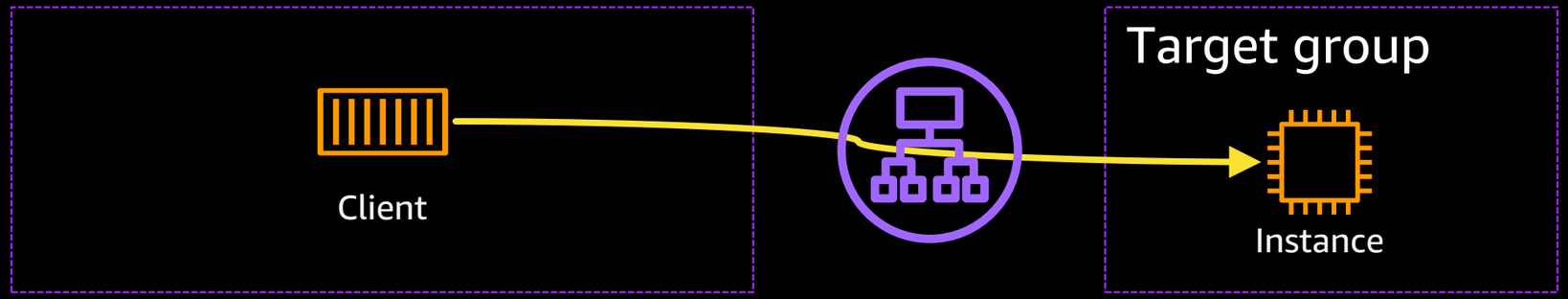
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

gRPC and HTTP/2 Application Load Balancer

- gRPC



- HTTP/2 to backends



gRPC

Unary calls

Streaming:

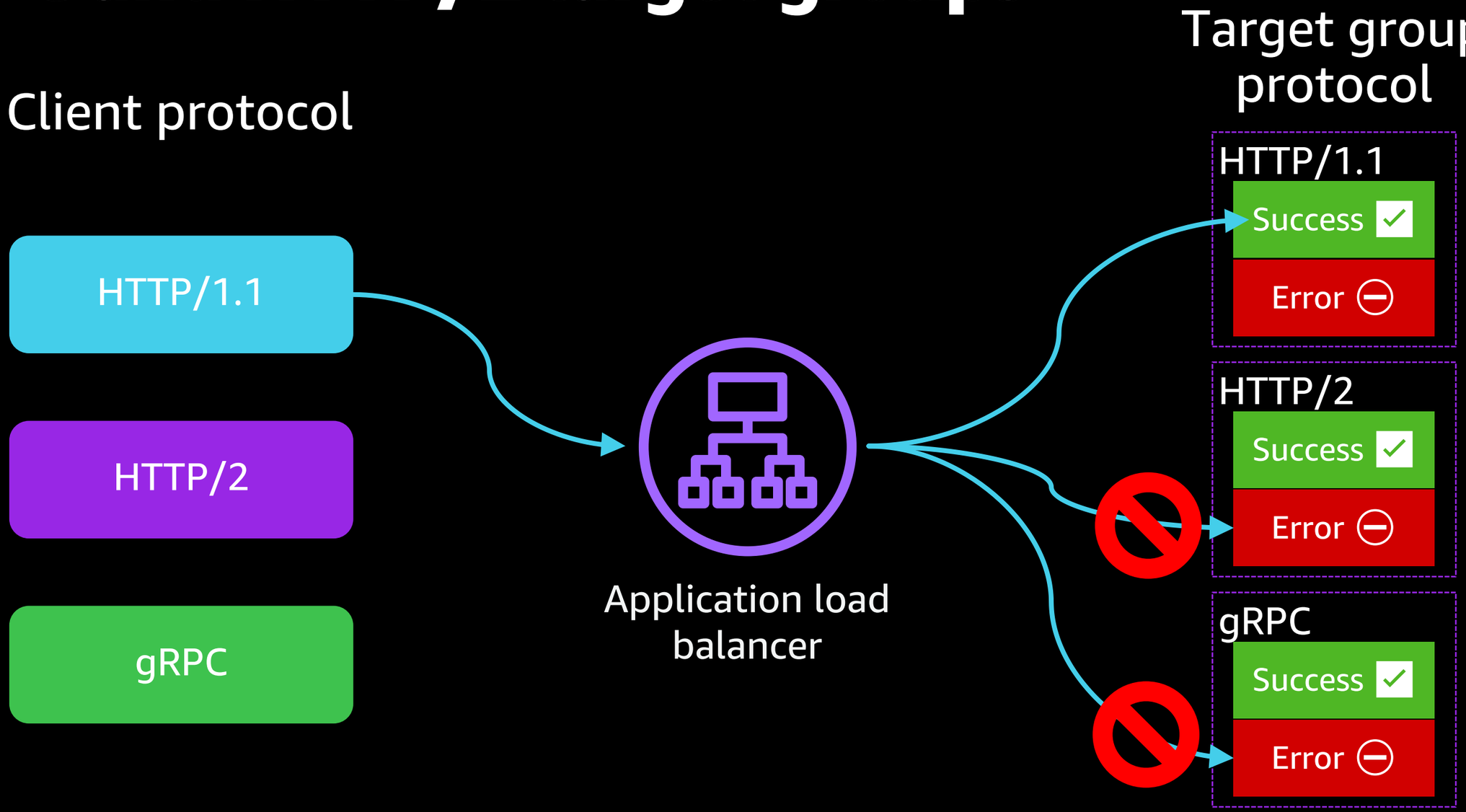
Client

Server

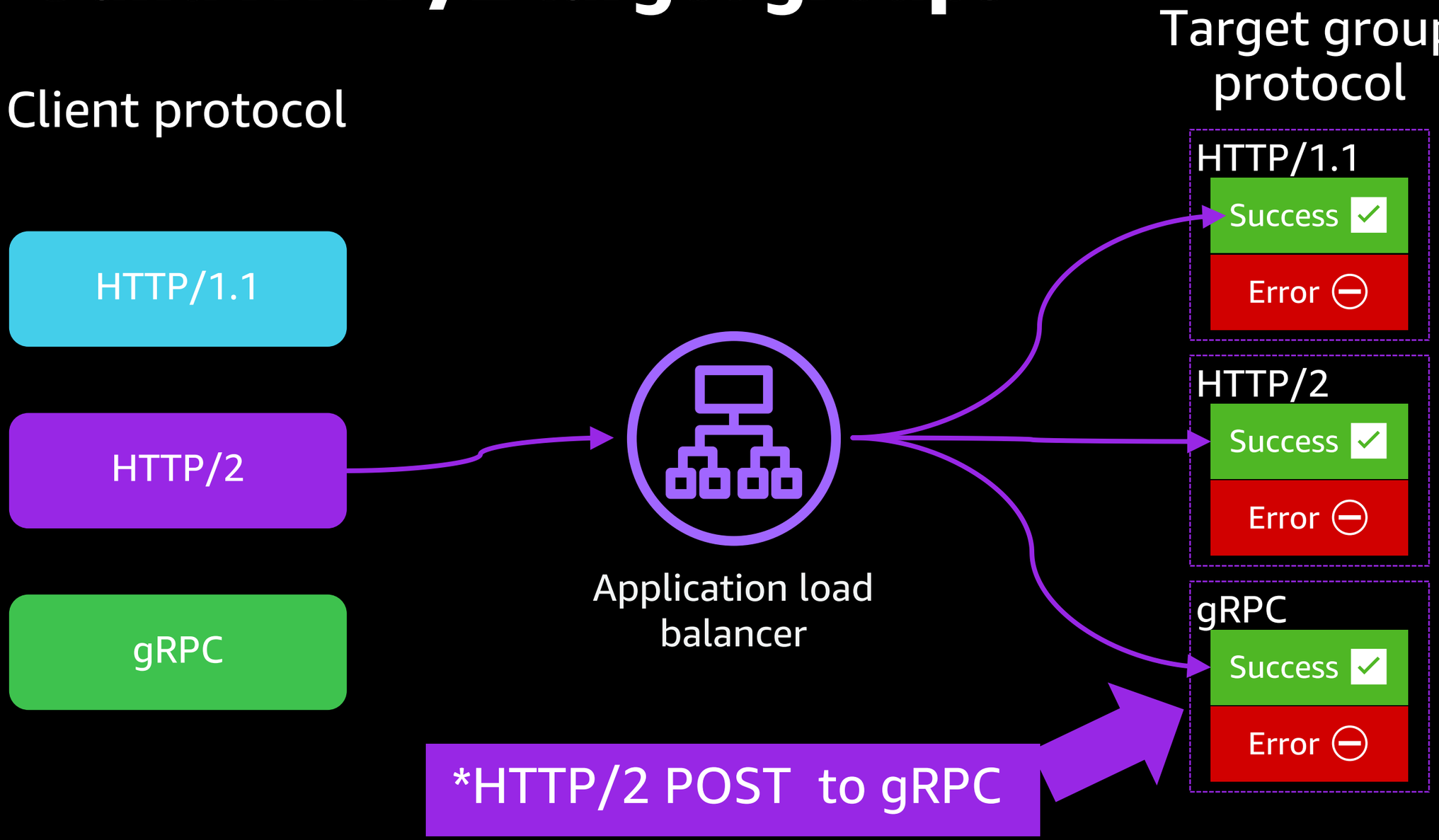
Bi-directional

HTTP/2

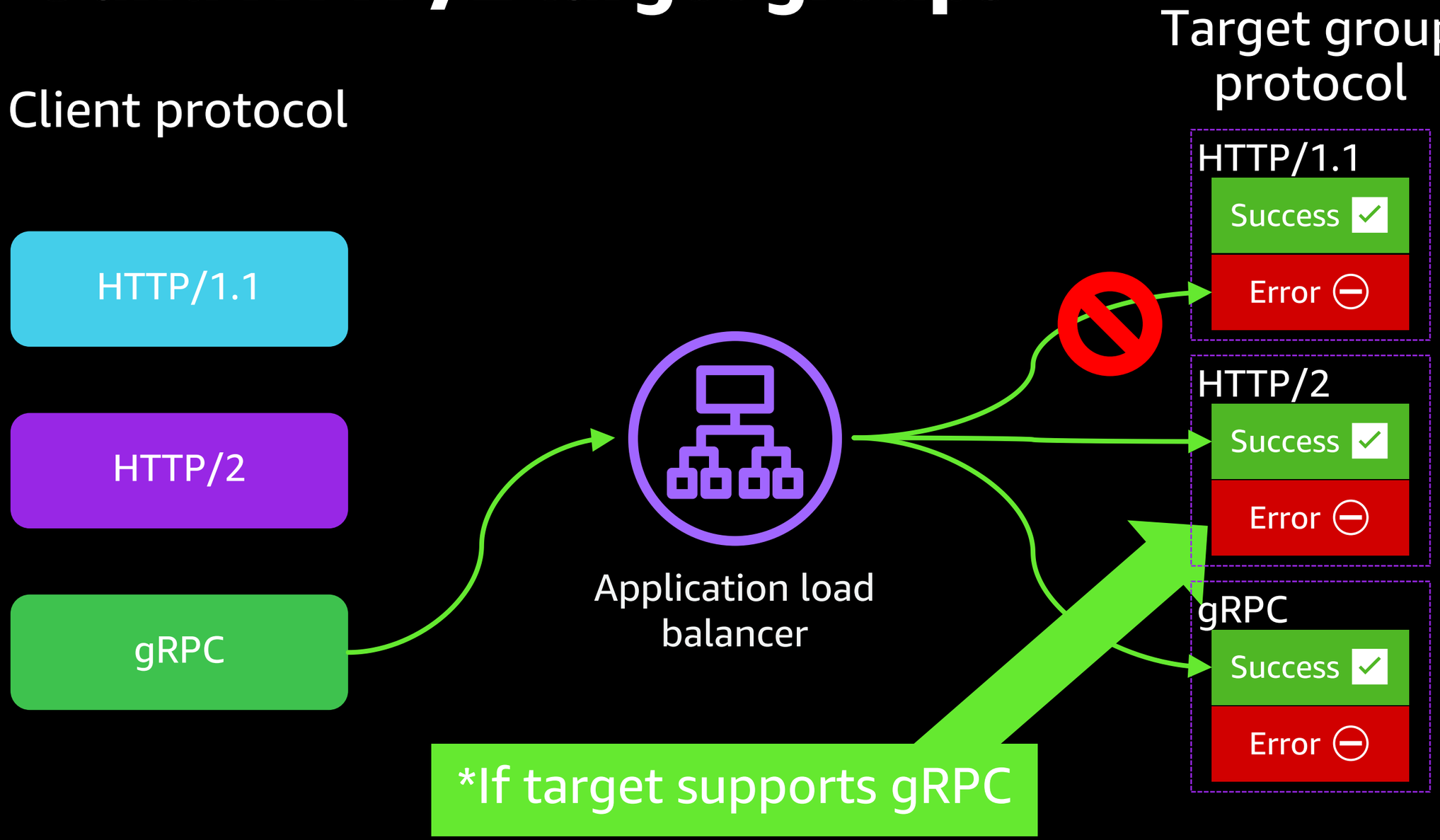
gRPC and HTTP/2 target groups



gRPC and HTTP/2 target groups



gRPC and HTTP/2 target groups

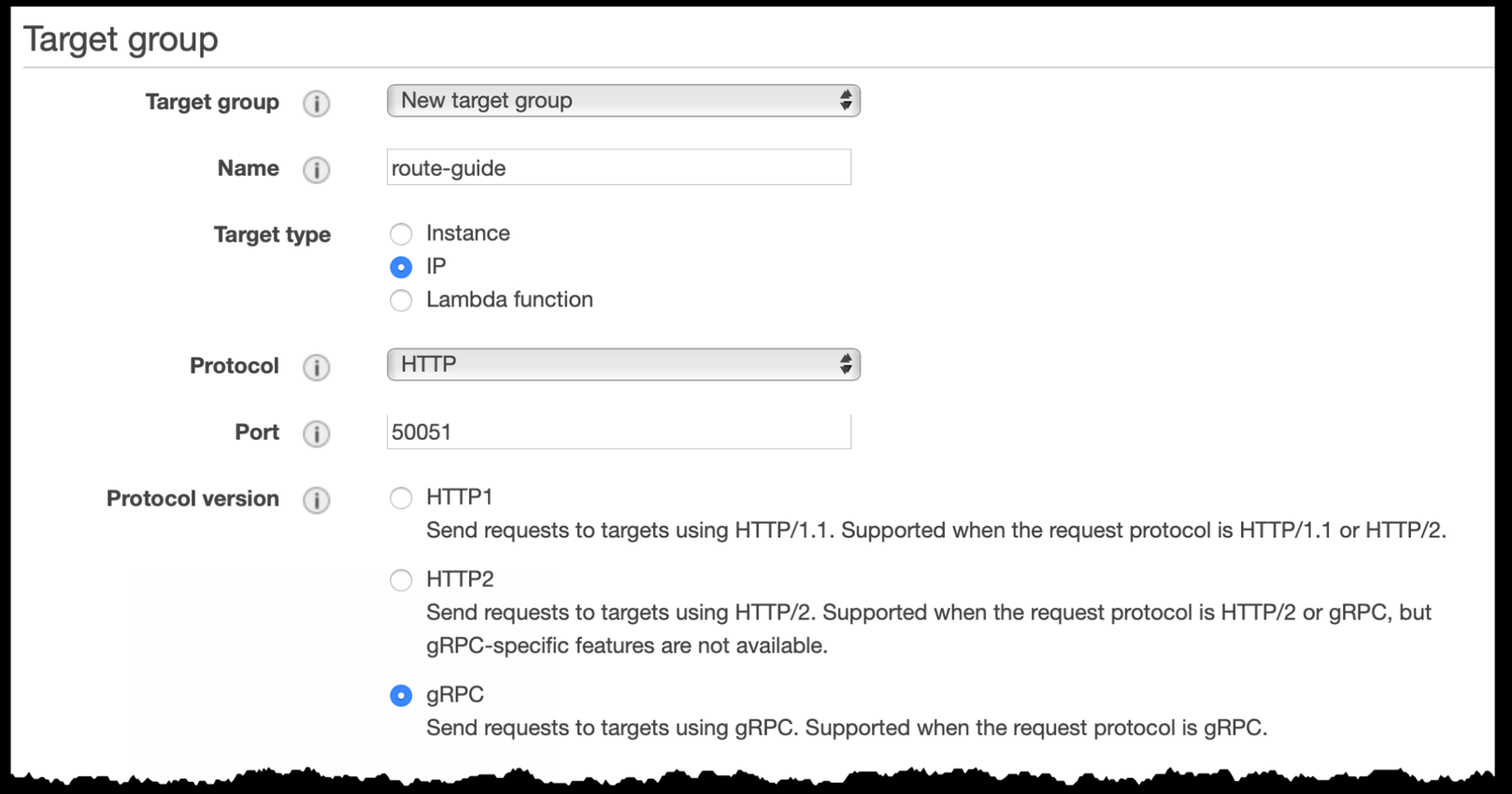


gRPC and HTTP/2 to targets

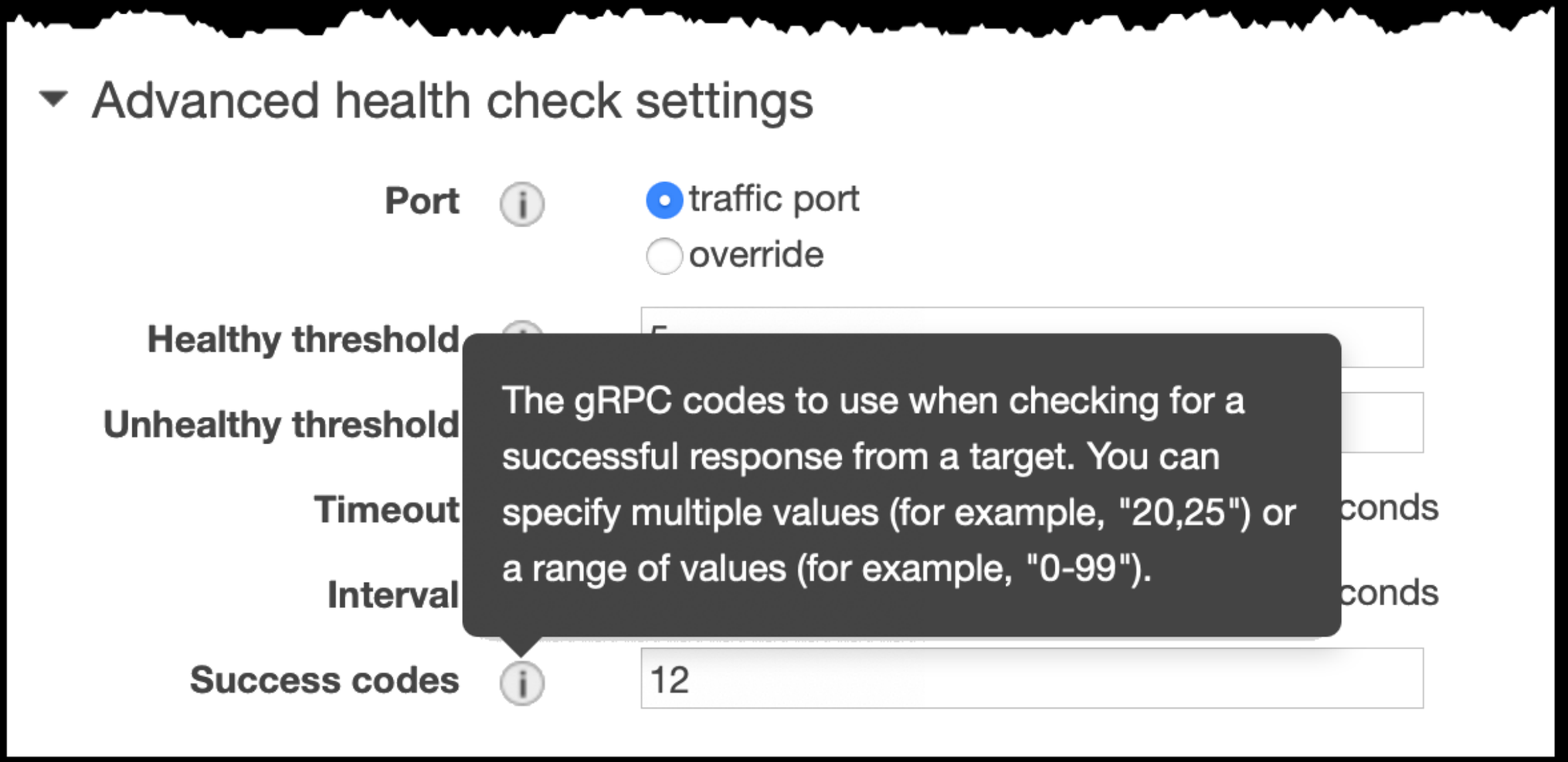
- Configure target group to be HTTP/2 or gRPC
  - Health checks come in target protocol version
- HTTP/2 target groups support gRPC *one way only*

Request protocol	Protocol version	Result
HTTP/2	HTTP/1.1	Success ✅
gRPC	HTTP/1.1	Error ⛔
HTTP/1.1	gRPC or HTTP/2	Error ⛔
gRPC	HTTP/2	Success ✅ if target supports gRPC
HTTP/2	gRPC	Success ✅ if a POST request
gRPC	gRPC	Success ✅

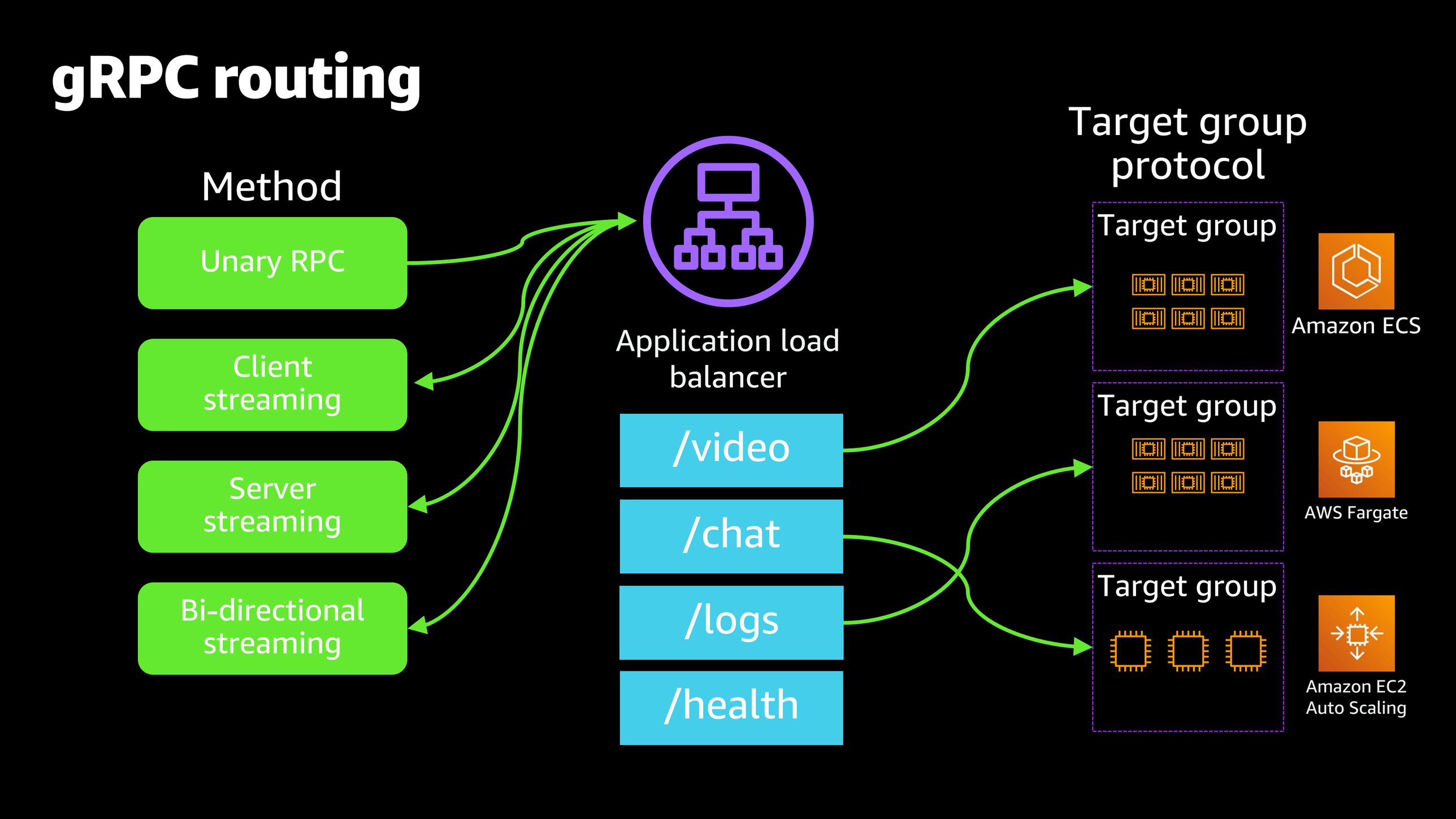
gRPC target group configuration



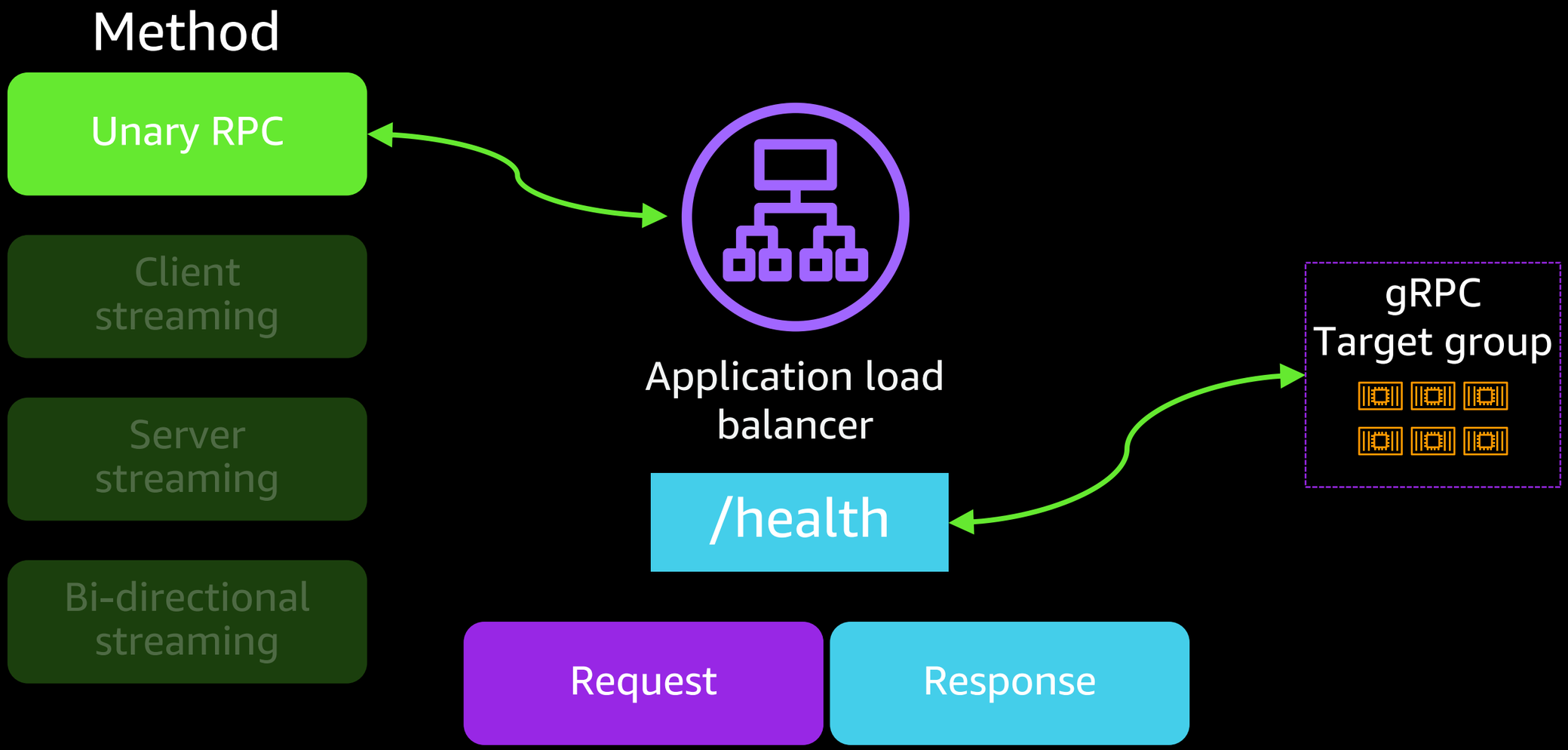
gRPC health check



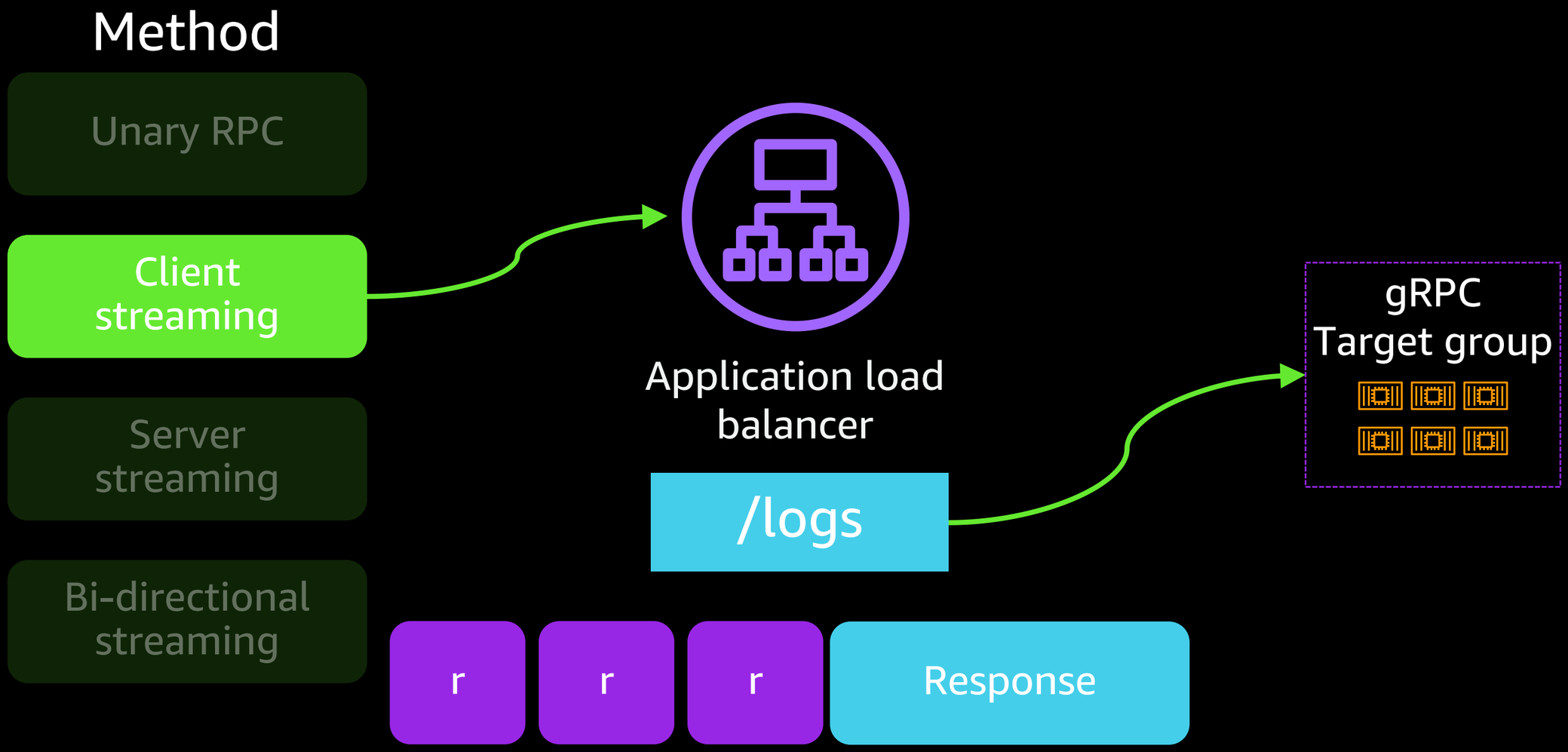
gRPC routing



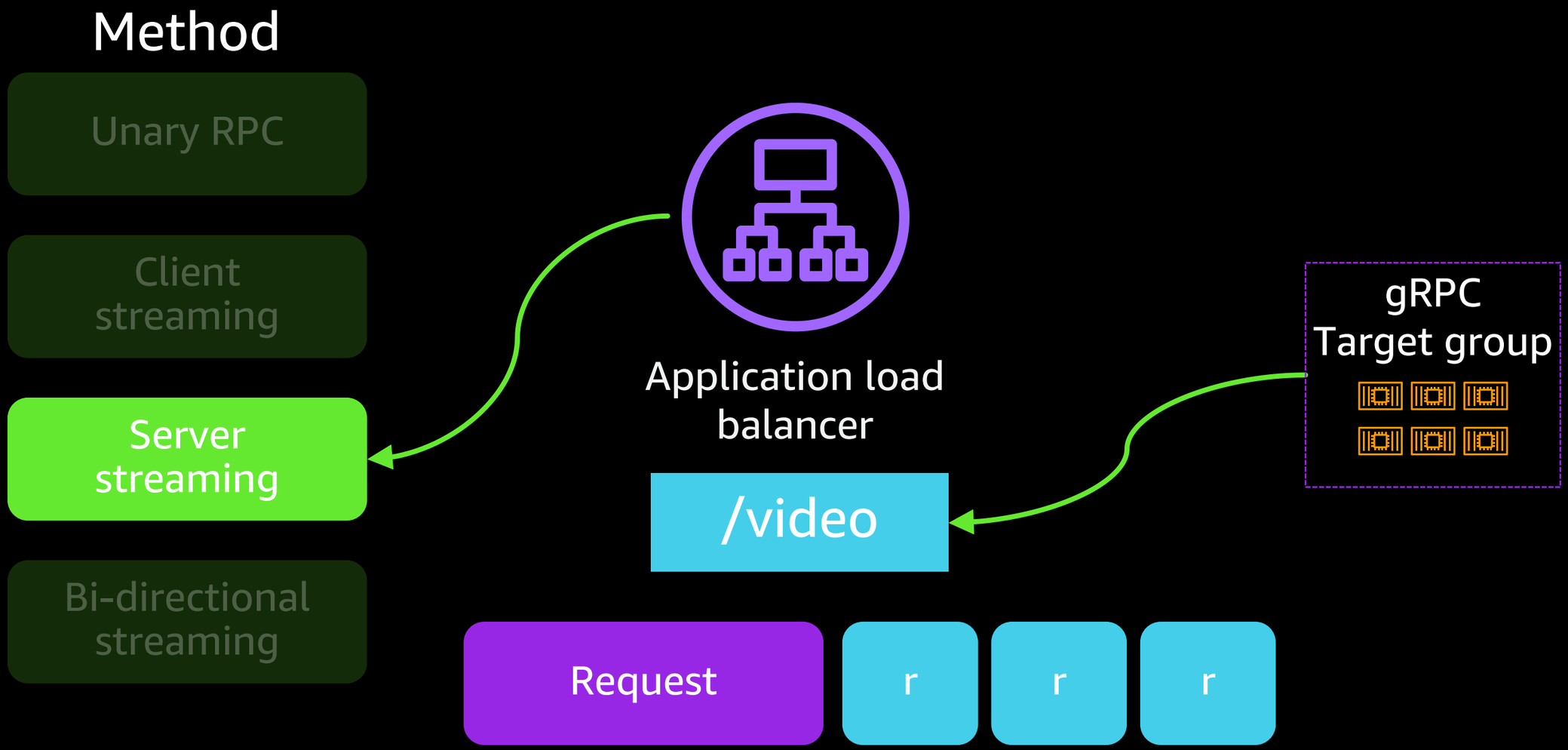
gRPC unary call



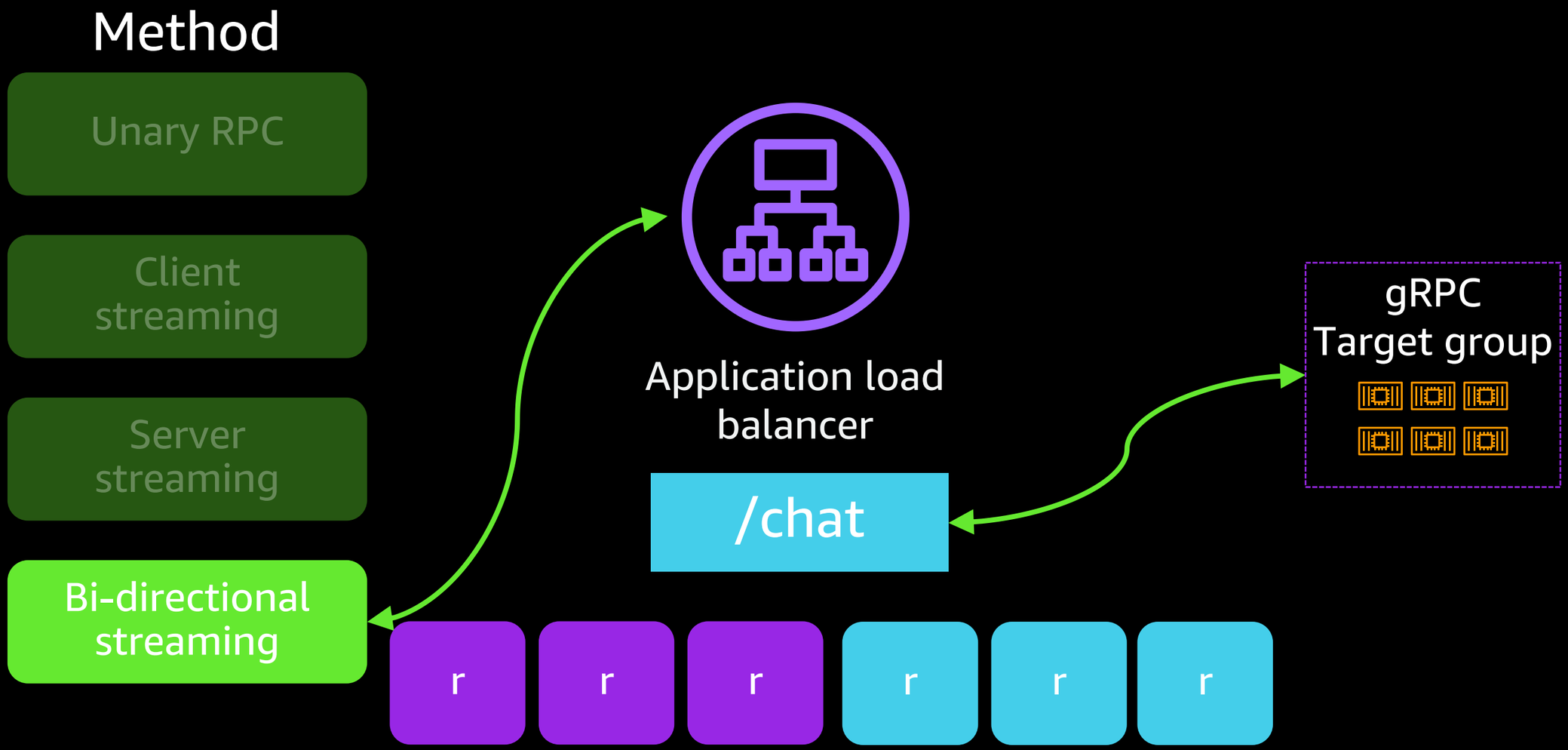
gRPC client streaming



gRPC server streaming



gRPC bi-directional streaming



HTTP DeSync Mitigation Mode



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

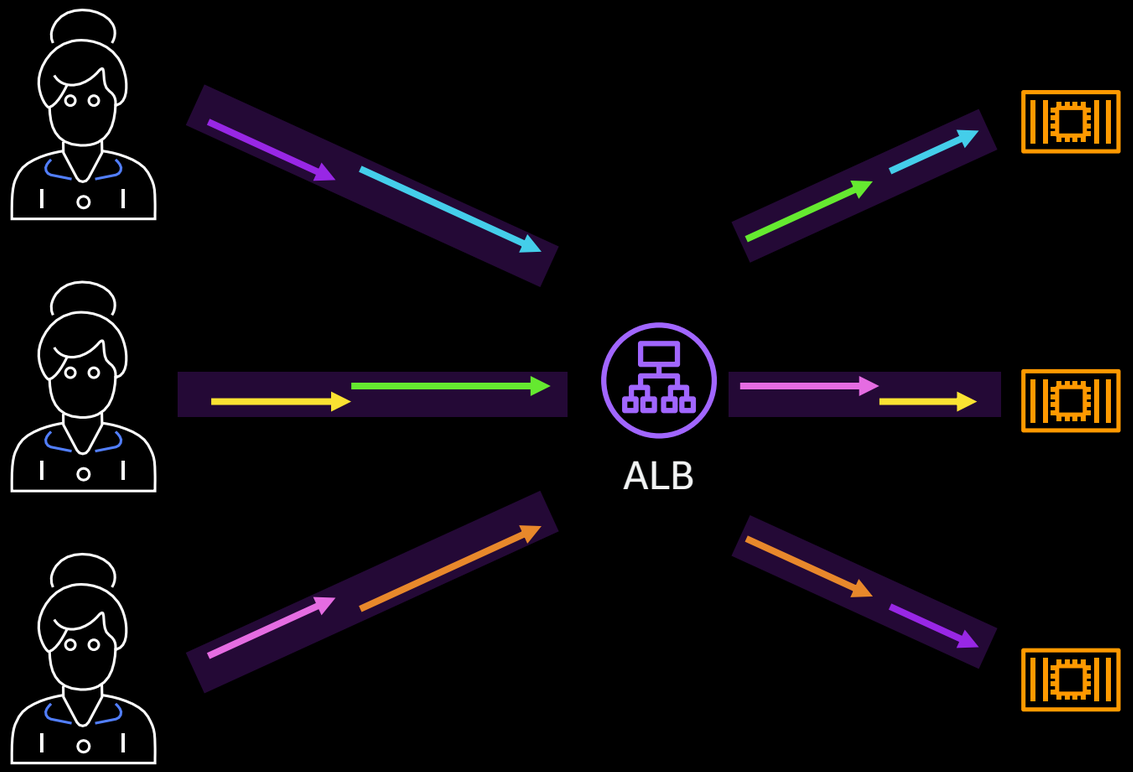
HTTP DeSync mitigation

What is HTTP DeSync?

What does HTTP DeSync Mitigation Mode do?

Configuration options and recommendations

HTTP requests and proxies

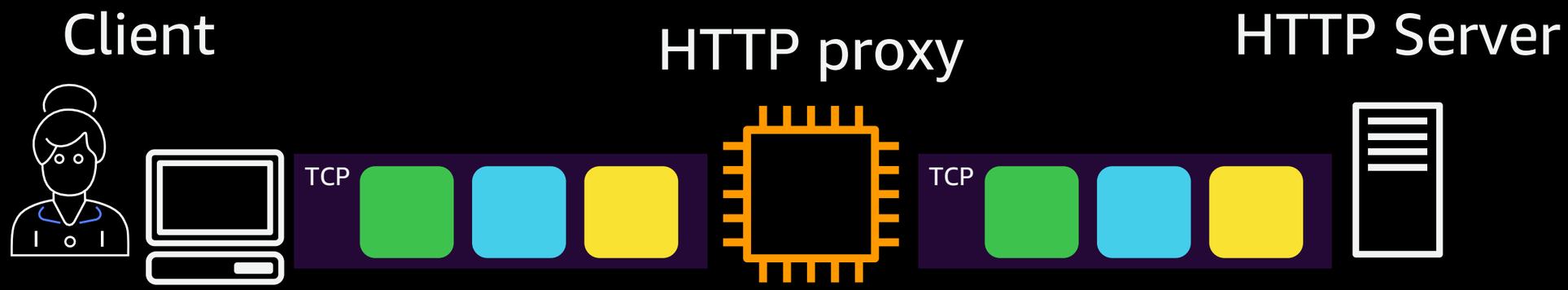


Reverse HTTP proxies

Sharing connections to targets between clients

HTTP Connection: Keep-alive is default on HTTP/1.1

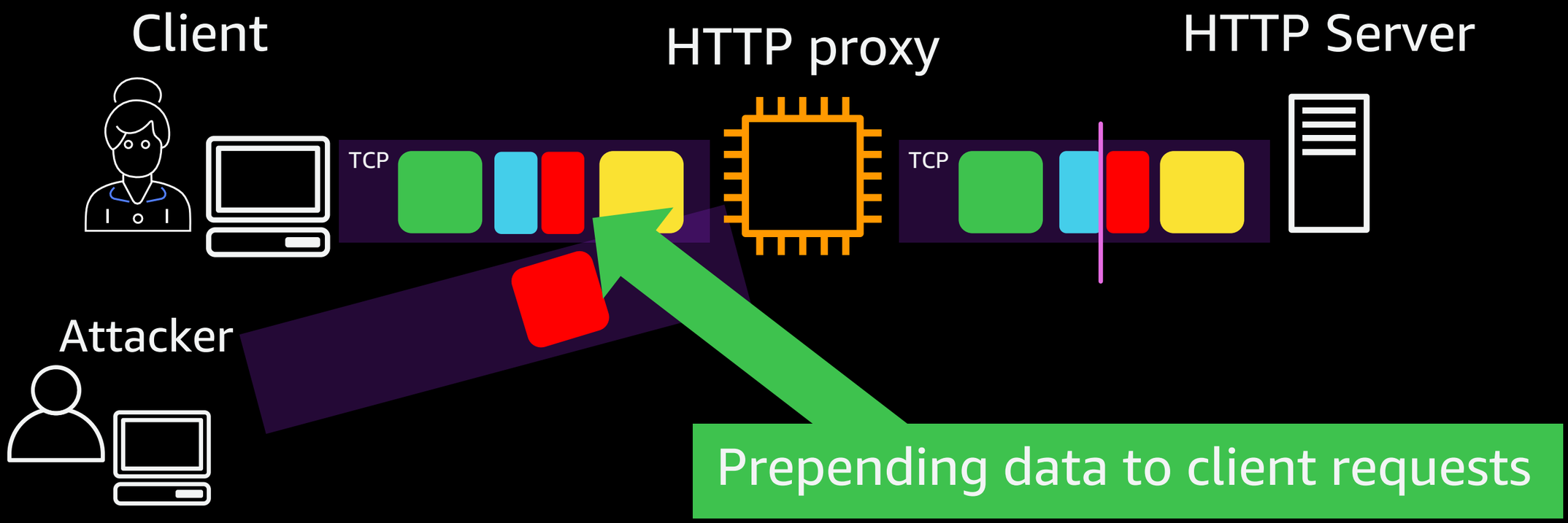
Normal HTTP communication



Layer 7 HTTP uses a layer 4 TCP connection

HTTP/1.1 requests are sequential

HTTP DeSync


Client
HTTP proxy
HTTP Server
TCP
TCP
Attacker
Prepending data to client requests

HTTP DeSync Mitigation Mode classifications

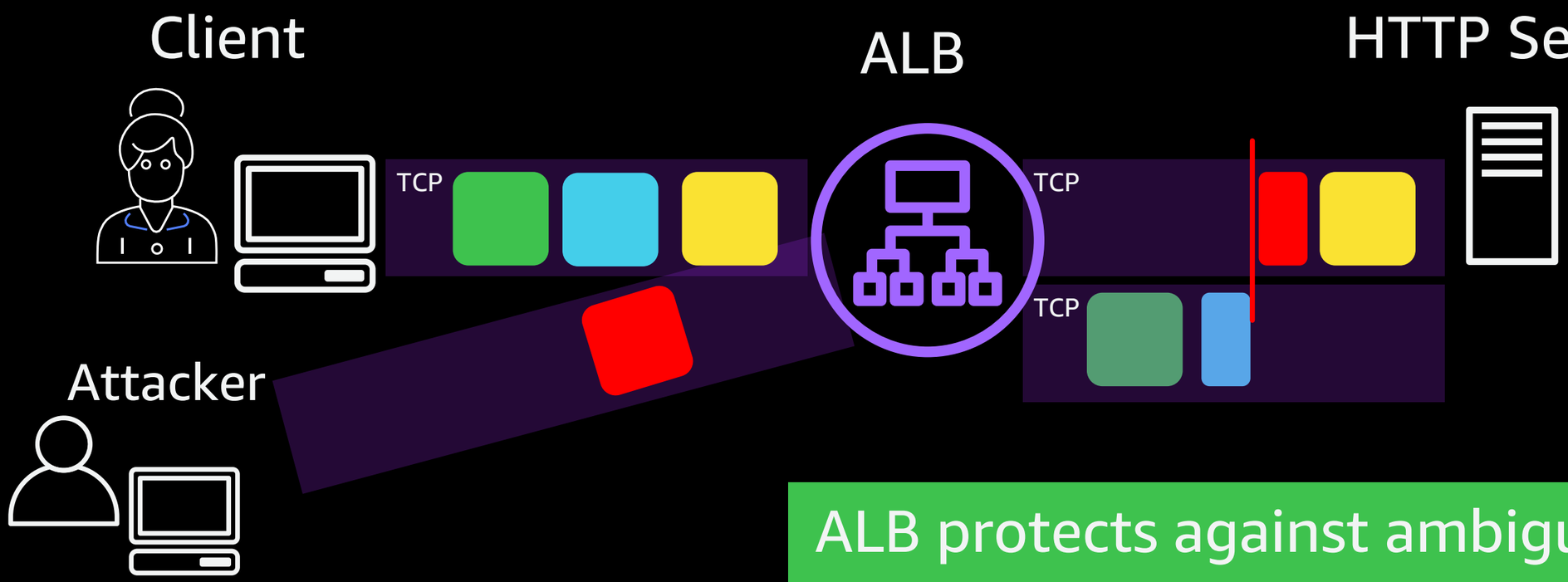
Compliant — Request complies with RFC 7230 and poses no known security threats.

Acceptable — Request does not comply with RFC 7230 but poses no known security threats.

Ambiguous — Request does not comply with RFC 7230 but poses a risk, as various web servers and proxies could handle it differently.

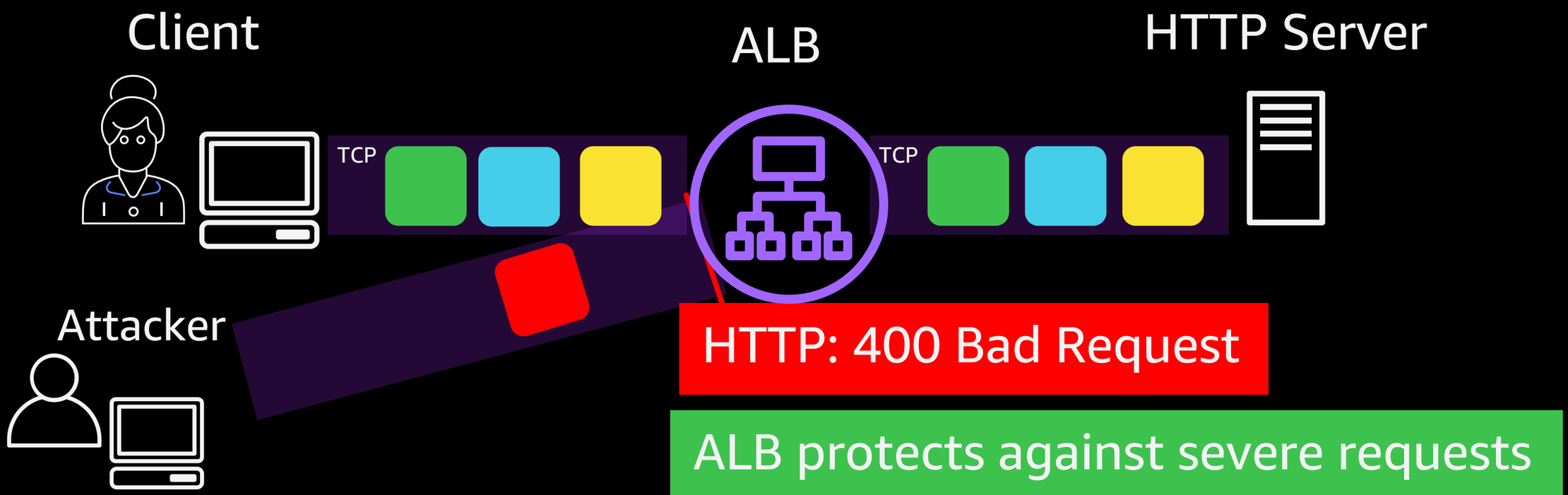
Severe — Request poses a high security risk. The load balancer blocks the request, serves a 400 response to the client, and closes the client connection.

HTTP DeSync mitigation connection close



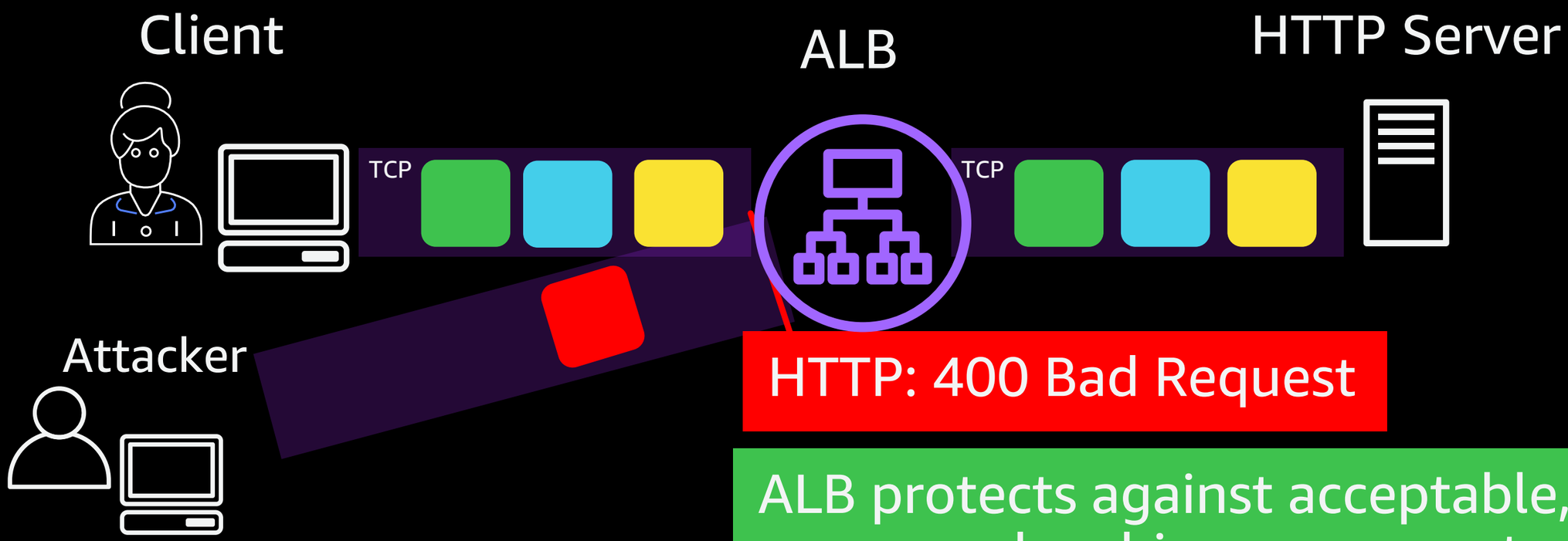
ALB protects against ambiguous requests by setting ‘Connection: close’ on **both** sides

HTTP DeSync mitigation reject request



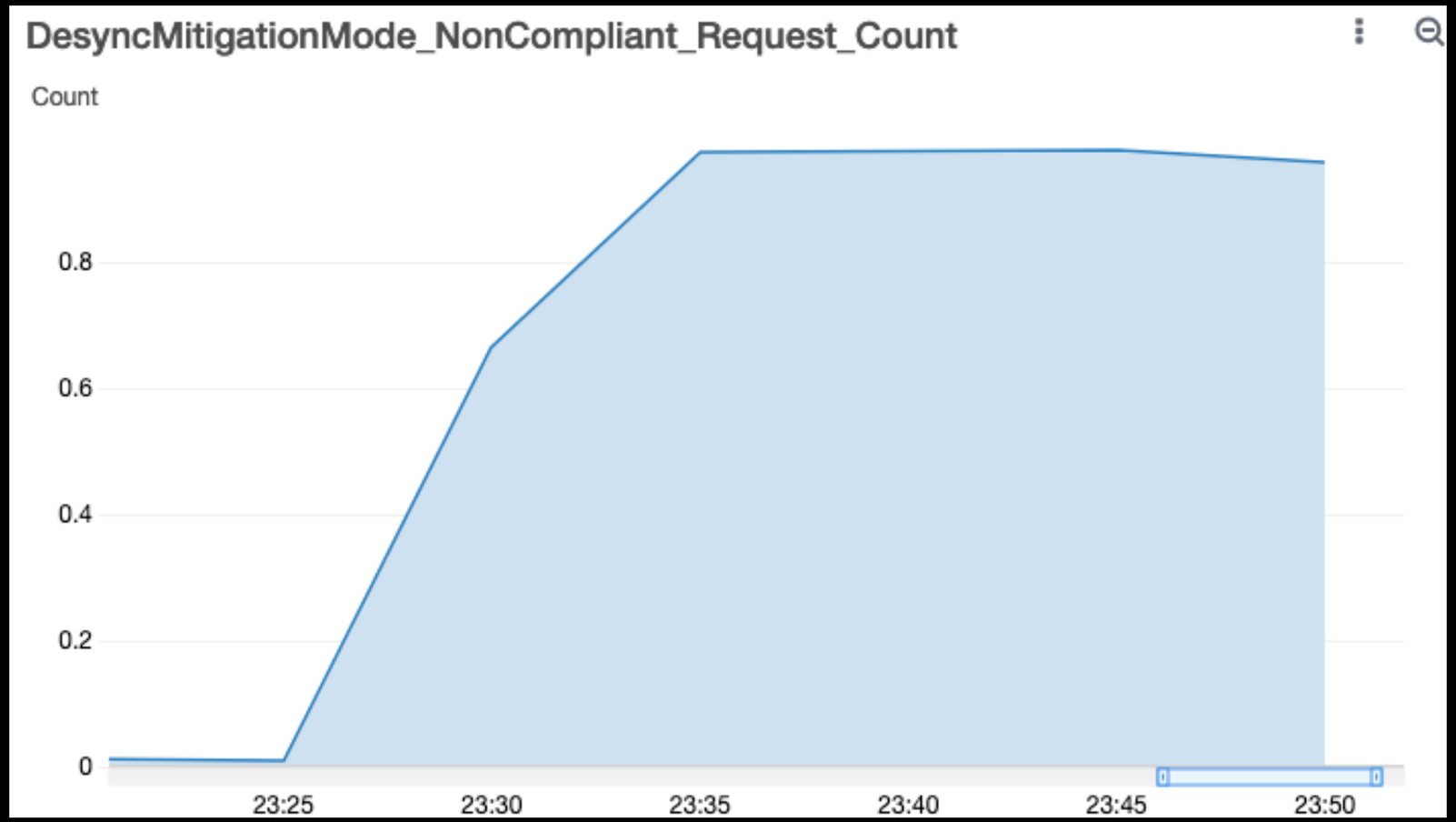
ALB protects against severe requests by rejecting with HTTP 400

HTTP DeSync mitigation strictest mode



ALB protects against acceptable, severe and ambiguous requests in *strictest* mode
by rejecting with HTTP 400

HTTP DeSync Mitigation Mode metric



HTTP DeSync Mitigation Mode access logs

classification — Acceptable, ambiguous, and severe

classification_reason — Detailed reason for classification

*https 2020-11-03T18:35:03.463678Z app/www-exampleloadbalancer-com/e6e1f2c9fbff1639 193.27.229.26:56496 172.31.5.55:32900 0.001 0.002 0.000 200 200 303 16083 "GET https:// www.exampleloadbalancer.com :443/vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36" ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 arn:aws:elasticloadbalancing:us-east-1:712882373746:targetgroup/exampleloadbalancer-auth/d00d9c1606fa635c "Root=1-5fa1a2d7-5931f02f629bb3fd42d66797" "-" "arn:aws:acm:us-east-1:712882373746:certificate/cb40a5dd-2ea1-4d1e-bf6d-275c60763403" 0 2020-11-03T18:35:03.233000Z "authenticate,forward" "-" "-" "172.31.5.55:32900" "200" "Ambiguous" "**UndefinedContentLengthSemantics**"*

HTTP DeSync access logs classification reason

AmbiguousUri	The request URI contains control characters.	Ambiguous
BadContentLength	The Content-Length header contains a value that cannot be parsed or is not a valid number.	Severe
BadHeader	A header contains a null character or carriage return.	Severe
BadTransferEncoding	The Transfer-Encoding header contains a bad value.	Severe
BadUri	The request URI contains a null character or carriage return.	Severe
BadMethod	The request method is malformed.	Severe
BadVersion	The request version is malformed.	Severe
BothTeClPresent	The request contains both a Transfer-Encoding header and a Content-Length header.	Ambiguous
DuplicateContentLength	There are multiple Content-Length headers with the same value.	Ambiguous
EmptyHeader	A header is empty or there is a line with only spaces.	Ambiguous
GetHeadZeroContentLength	There is a Content-Length header with a value of 0 for a GET or HEAD request.	Acceptable
MultipleContentLength	There are multiple Content-Length headers with different values.	Severe
MultipleTransferEncodingChunked	There are multiple Transfer-Encoding: chunked headers.	Severe
NonCompliantHeader	A header contains a non-ASCII or control character.	Acceptable
NonCompliantVersion	The request version contains a bad value.	Acceptable
SpaceInUri	The request URI contains a space that is not URL encoded.	Acceptable
SuspiciousHeader	There is a header that can be normalized to Transfer-Encoding or Content-Length using common text normalization techniques.	Ambiguous
UndefinedContentLengthSemantics	There is no Content-Length header defined for a GET or HEAD request.	Ambiguous
UndefinedTransferEncodingSemantics	There is no Transfer-Encoding header defined for GET or HEAD request.	Ambiguous

See the docs

HTTP DeSync summary

Request classification	Monitor	Defensive	Strictest
Compliant	Allowed	Allowed	Allowed
Acceptable	Allowed	Allowed	Blocked
Ambiguous	Allowed	Allowed	Blocked
Severe	Allowed	Blocked	Blocked

HTTP DeSync Mitigation Mode summary

Available on all HTTP ELBs — Application and Classic Load Balancers

Default defensive mode

Configuration options — monitoring, defensive, strictest

Metrics — non-compliant requests

ALB access logs show selections

INTRODUCING

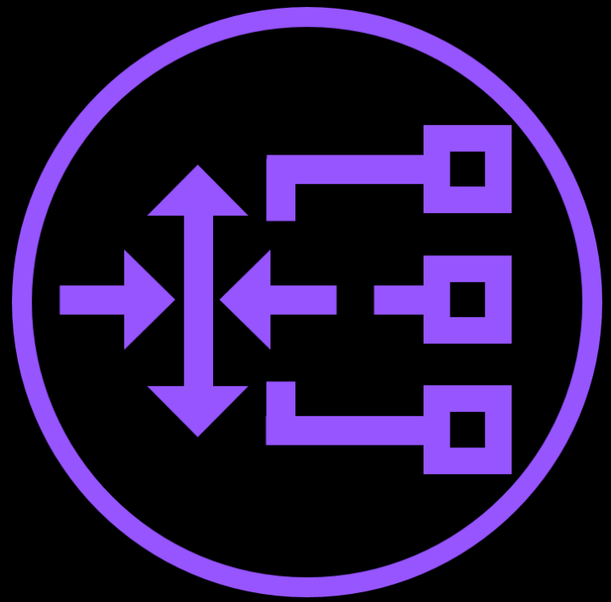
Gateway Load Balancer

Reduce complexity and deploy faster

Elastically scale and reduce costs

Improve appliance availability

Supported by leading appliance vendors



Network appliances



Transparent to network traffic



Security, monitoring, analytics, and other use cases

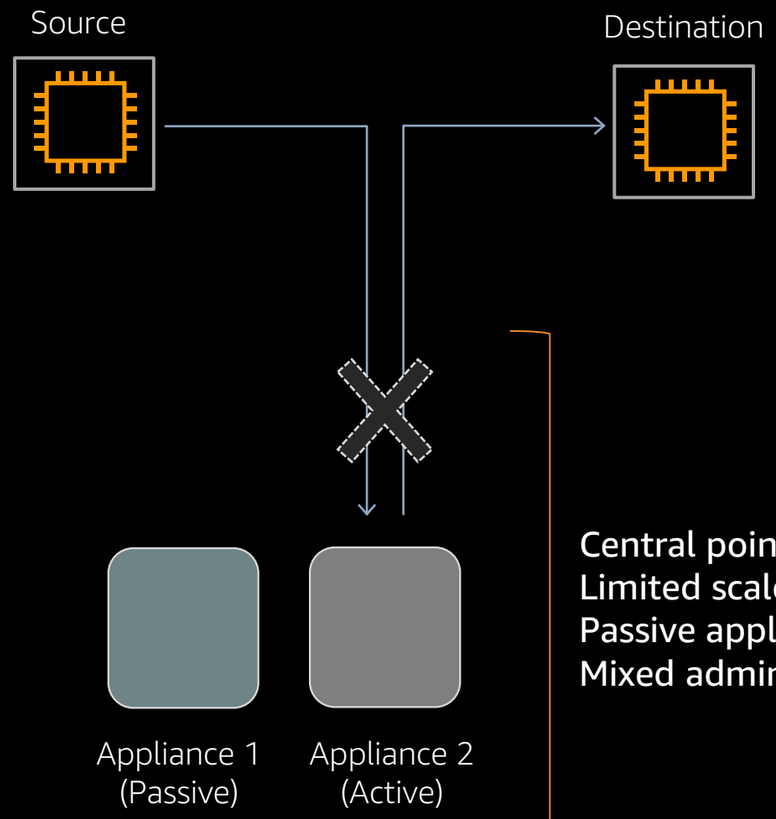


Often required by policy, or due to expertise and investment

Use the same Network Appliances on AWS and Hybrid Environments

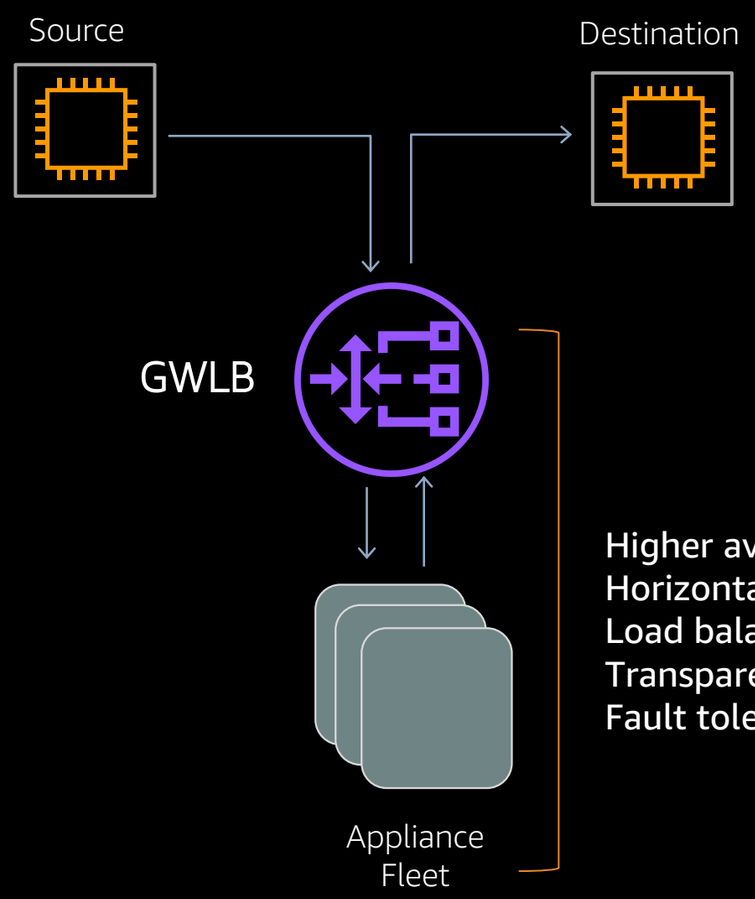
Challenges today and benefits now available

Before Gateway Load Balancer



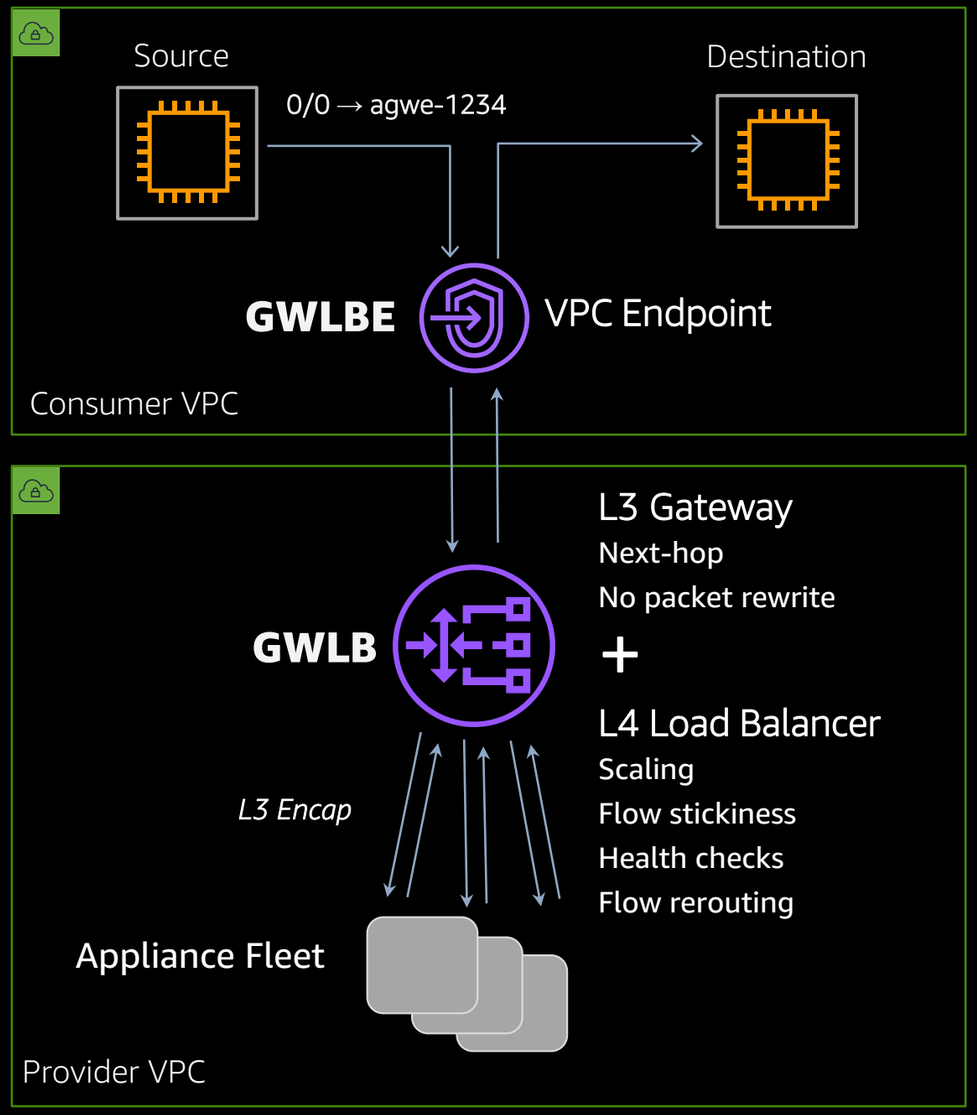
Central point of failure
Limited scale
Passive appliances sit idle
Mixed admin domains

After Gateway Load Balancer



Higher availability
Horizontal scaling
Load balancing
Transparent
Fault tolerant

Gateway Load Balancer: At-a-glance



Components

- Gateway Load Balancer Endpoint (GWLBE) - A new type of VPC endpoint that can be a next-hop in a VPC route table
- Gateway Load Balancer (GWLB) - A new type of load balancer that includes L3 Gateway + L4 Load Balancer capabilities
- Both components powered by AWS Hyperplane

Benefits

- Provide horizontal scaling to appliances
- Provide fault tolerance to appliances
- Transparent to network traffic, no change to source traffic
- Separate security and user admin domains, share across different VPCs and AWS accounts
- Provide appliance-as-a-service, (e.g. Firewall-as-a-service)

Deployment

- Create GWLB and appliance fleet using steps similar to NLB
- Send traffic to GWLB/GWLBE by updating VPC route tables

Launch partners

GATEWAY LOAD BALANCER



















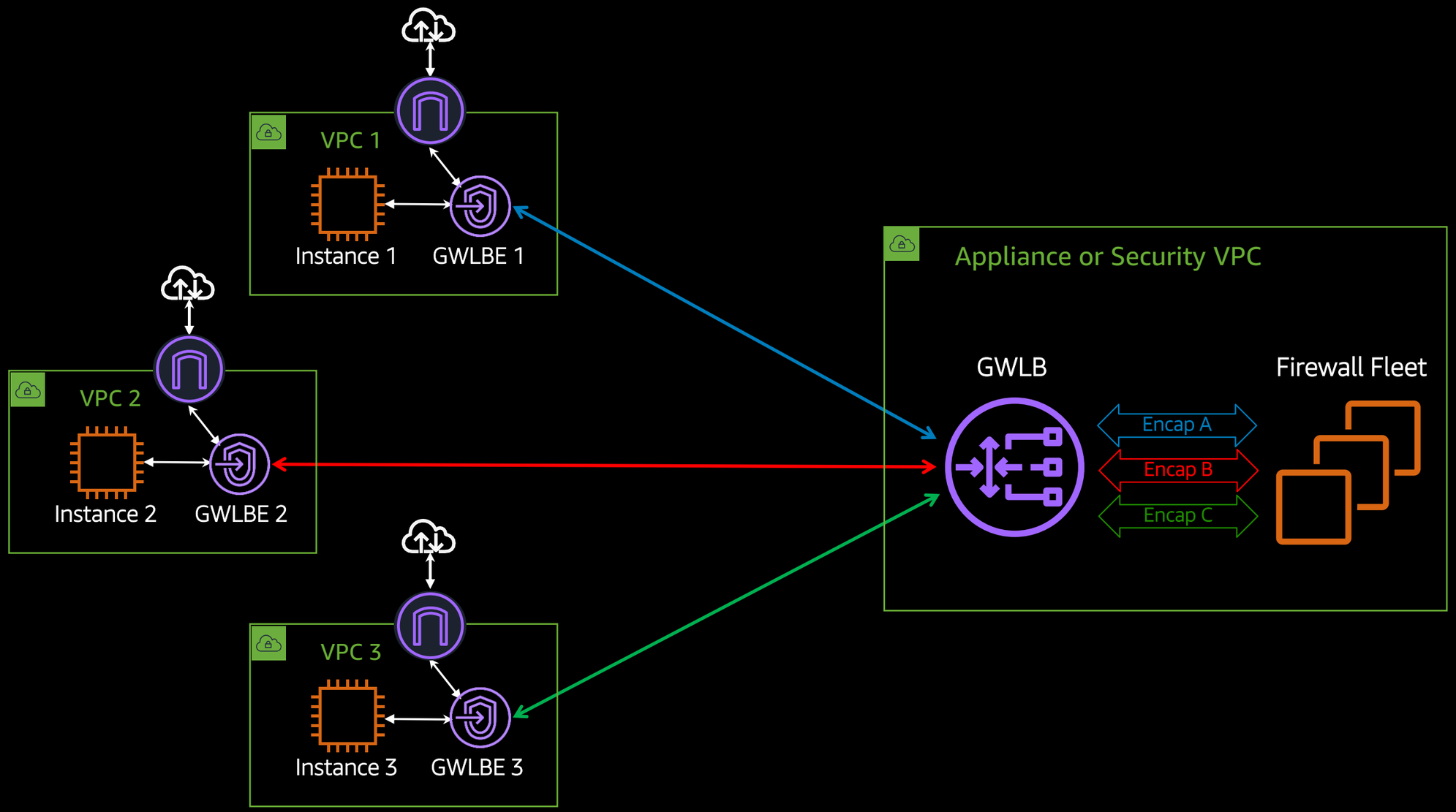




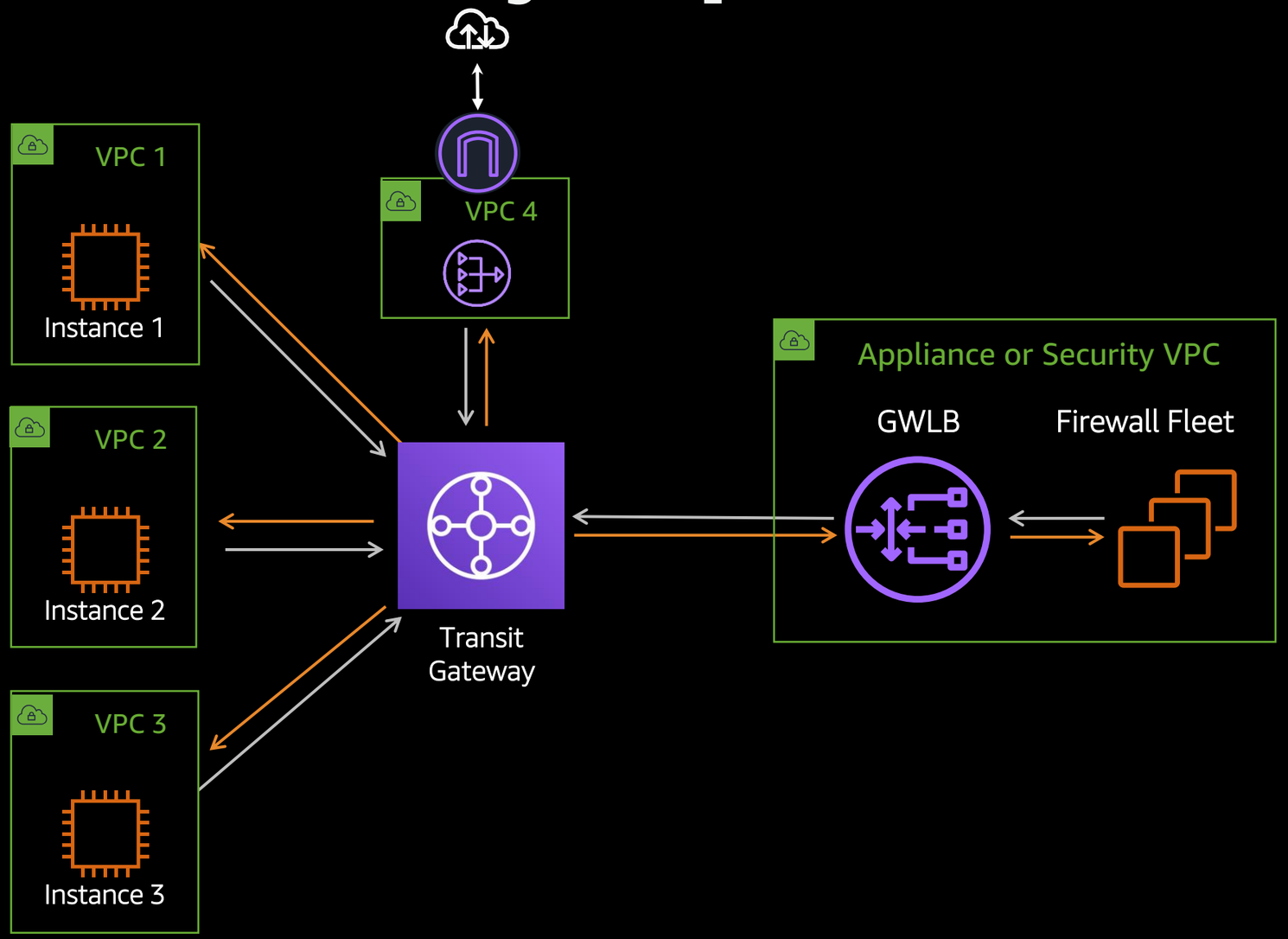




Distributed security inspection with GWLBE

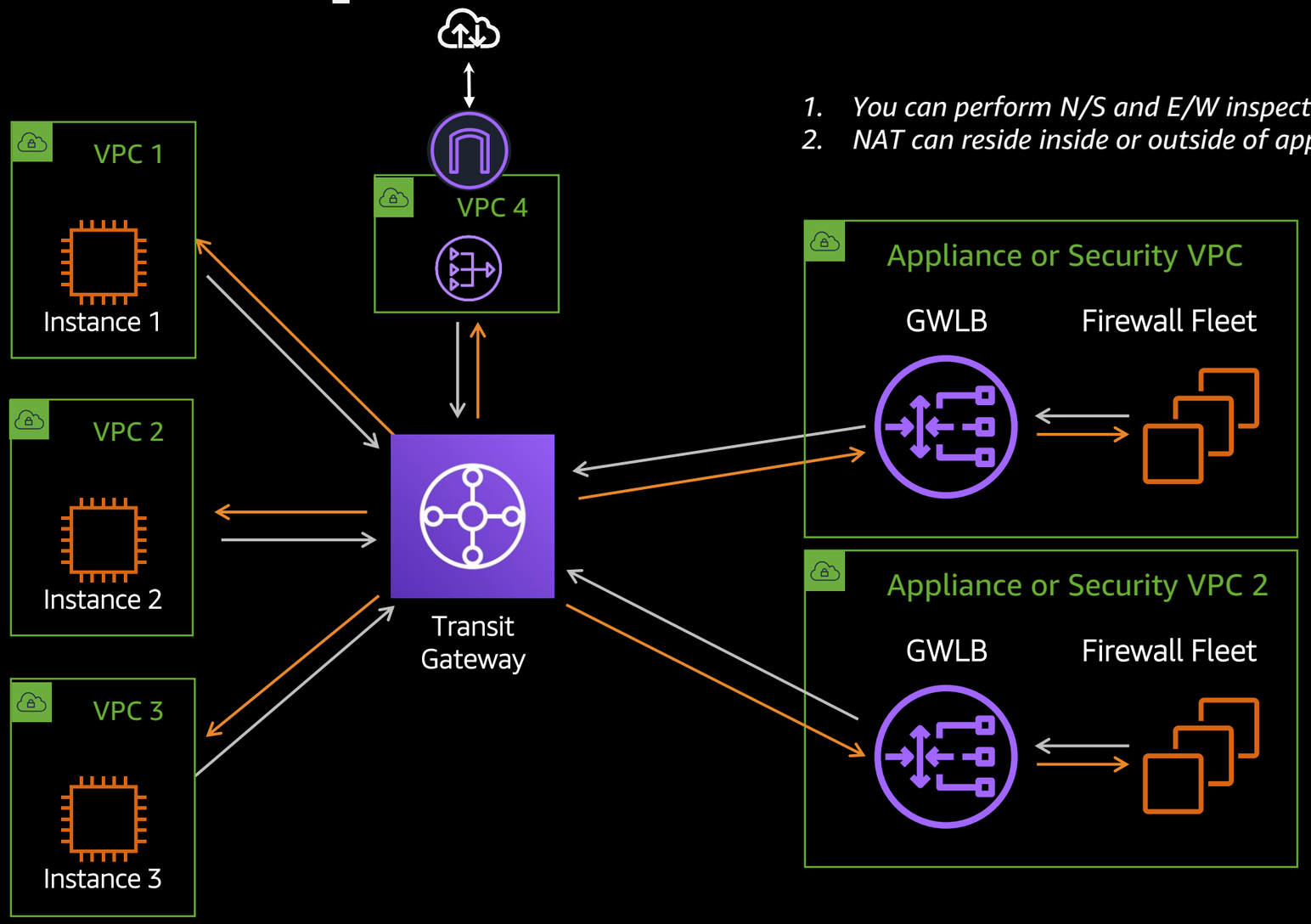


Centralized security inspection with TGW



North-south inspection
(Traffic to/from internet)

VPC to VPC Inspection with TGW



1. *You can perform N/S and E/W inspection using same fleet.*
2. *NAT can reside inside or outside of appliance fleet VPC*

North-south inspection
(Traffic to/from internet)

East-west inspection
(VPC to VPC traffic)

Related session

- NET301: Choosing the right load balancer for serverless applications



Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.