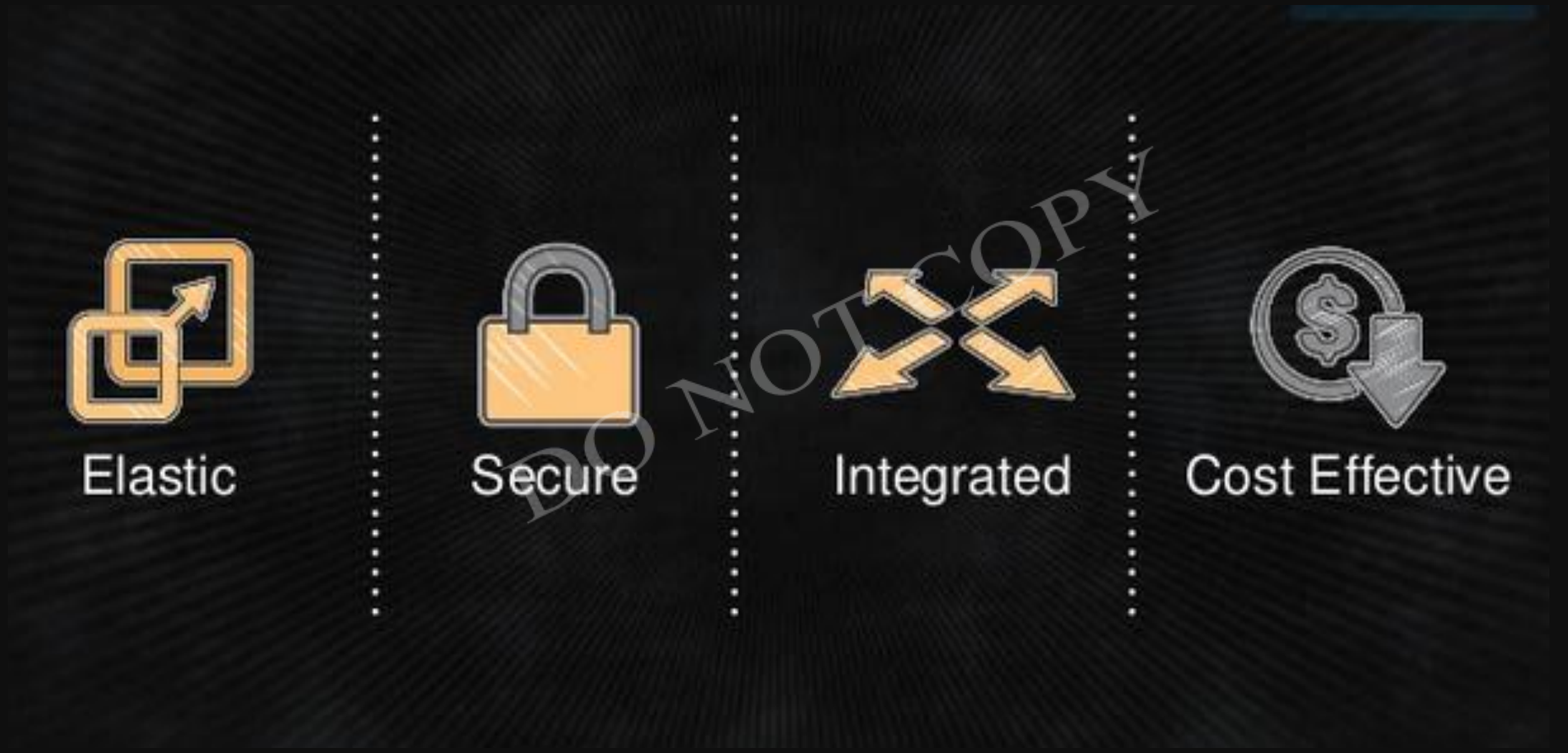
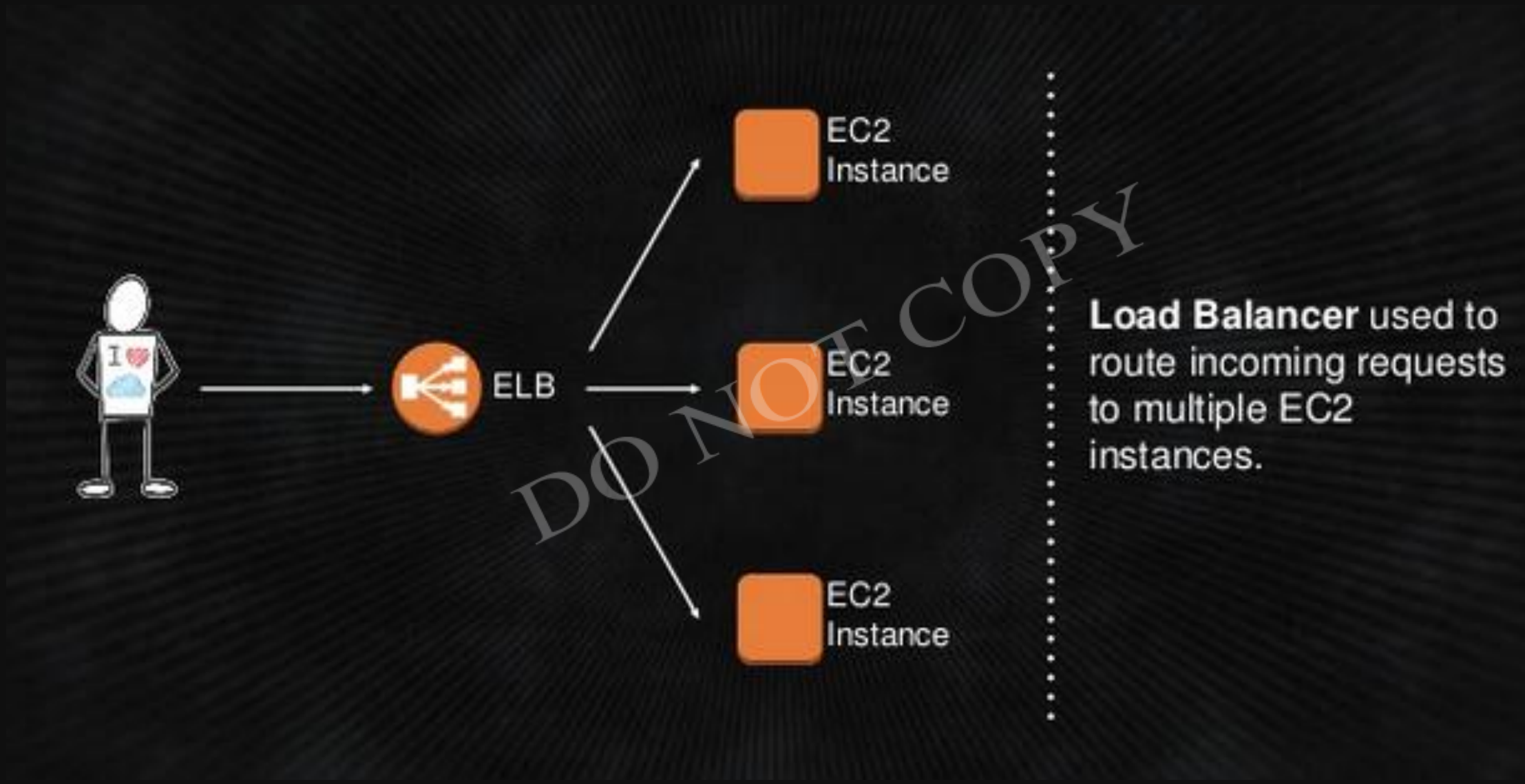
AMAZON WEB SERVICES ELB

Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances.




EC2
Instance
EC2
Instance
EC2
Instance
ELB
Load Balancer used to route incoming requests to multiple EC2 instances.

EC2-Classic

Load balance over classic EC2 instances.

Support for public IP addresses only.

No control over the load balancer security group.

EC2-VPC

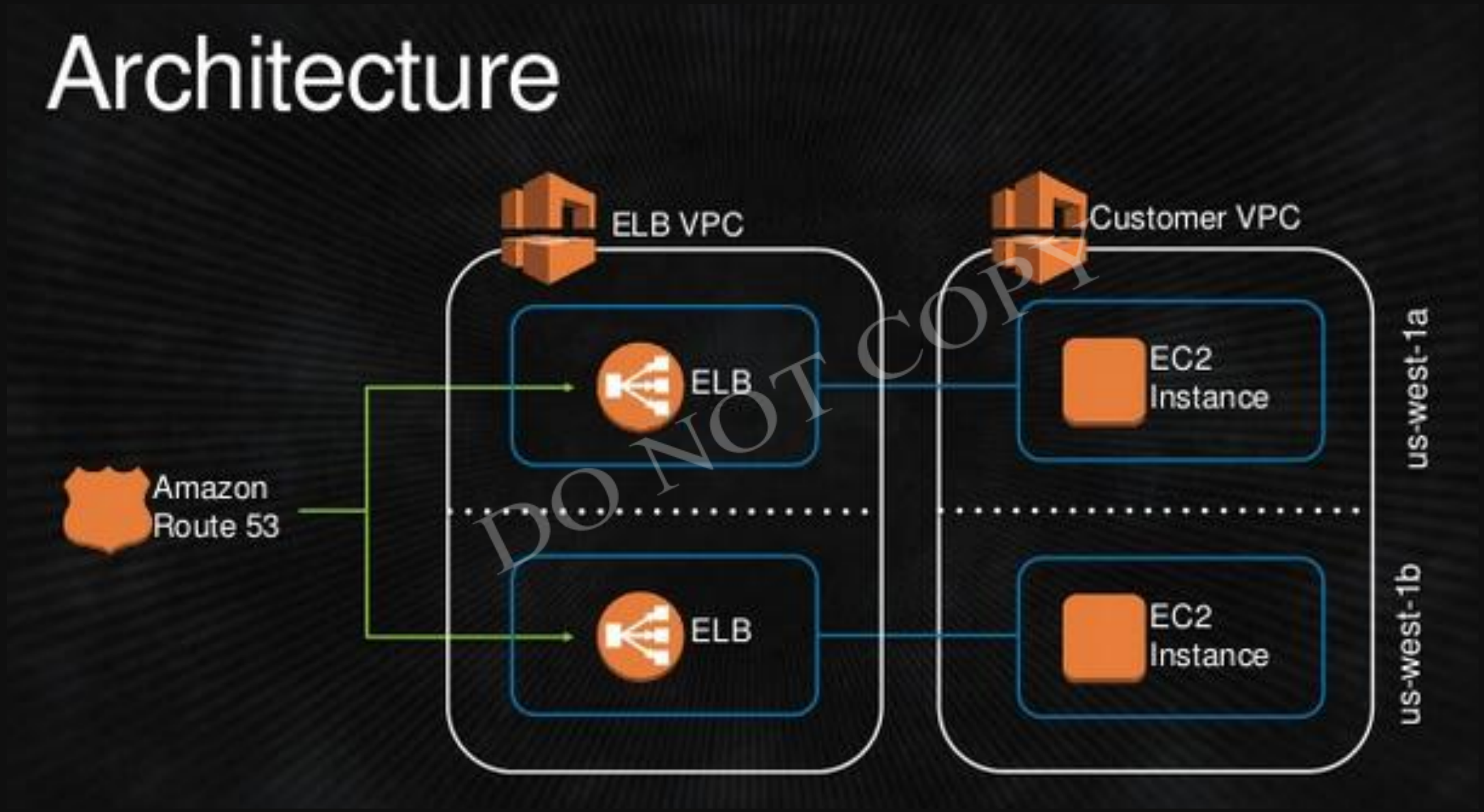
Load balance over EC2 instances within a VPC.

Support for both public and private IP addresses.

Full control over the load balancer security group.

Tightly integrated into the associated VPC and subnets.

Architecture



TCP/SSL

Incoming client connection bound to server connection

No header modification

Proxy Protocol prepends source and destination IP and ports to request

Round robin algorithm used for request routing

HTTP/HTTPS

Connection terminated at the load balancer and pooled to the server

Headers may be modified

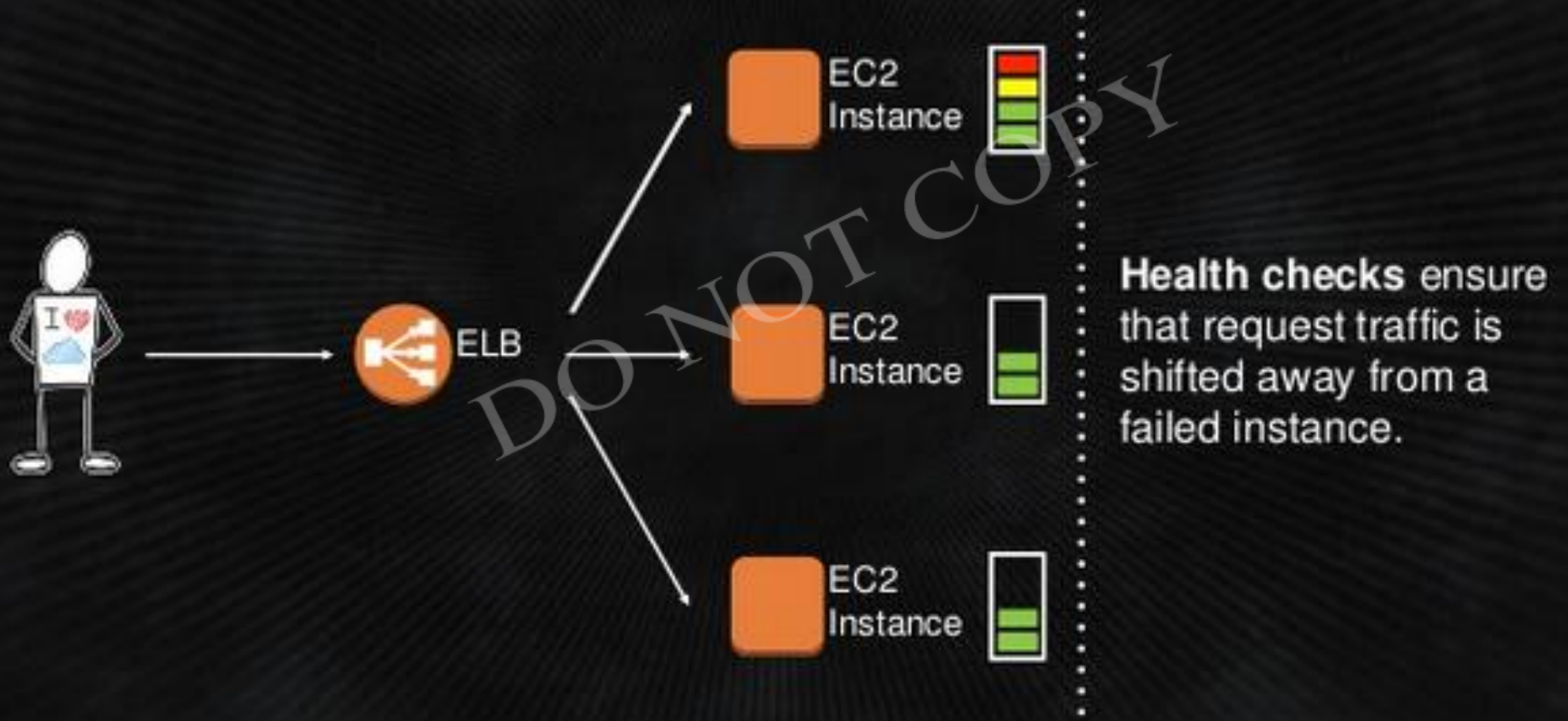
X-Forwarded-For header contains client IP address

Least outstanding requests algorithm used for request routing

Sticky session support available

Health checks allow for traffic to be shifted away from failed instances

Health Checks



Health Checks

Clip slide

Support for TCP and HTTP health checks.

Customize the frequency and failure thresholds.

Must return a 2xx response.

Consider the depth and accuracy of your health checks.

Idle timeouts allow for connections to be closed by the load balancer when no longer in use.

Idle Timeouts

Length of time that an idle connection should be kept open.

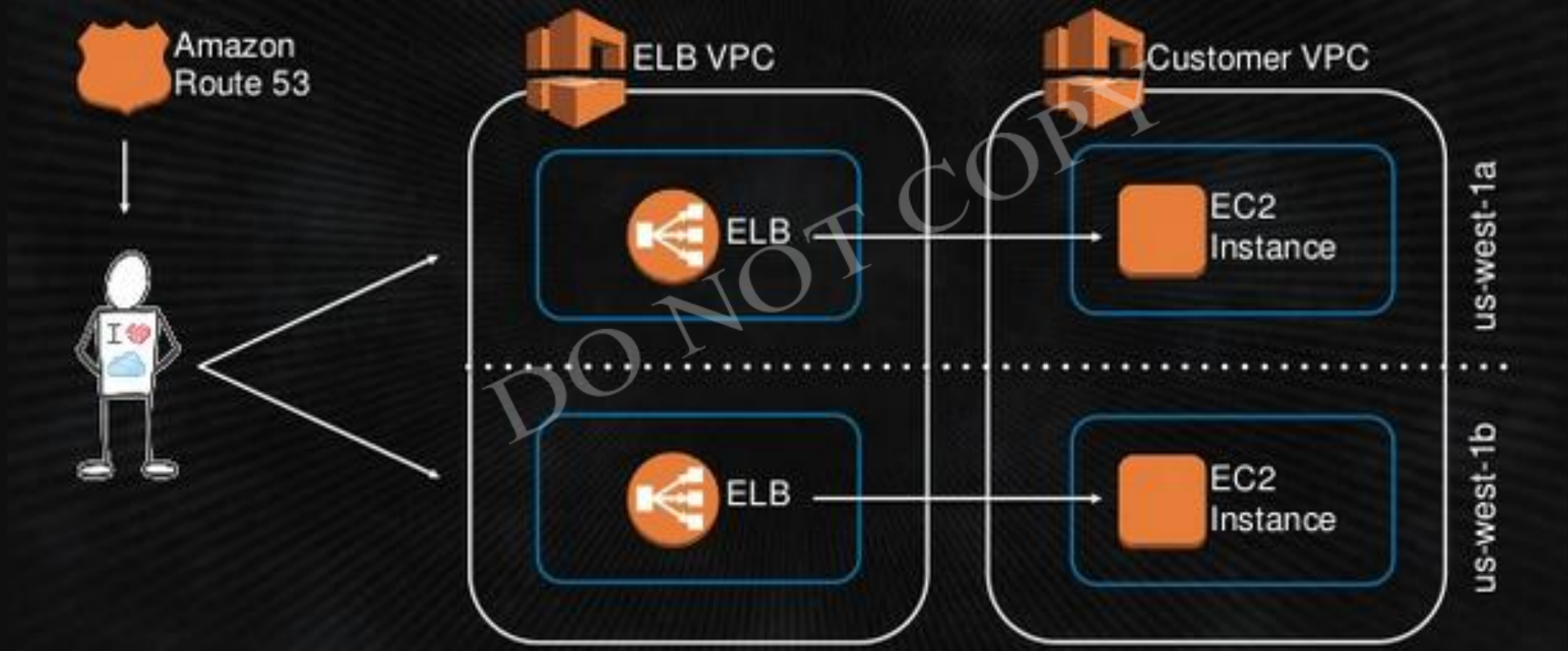
For both client and back-end connections.

Defaults to 60 seconds but can be set between 1 and 3,600 seconds.

Timeouts should decrease as you go up the stack.

Using multiple Availability Zones

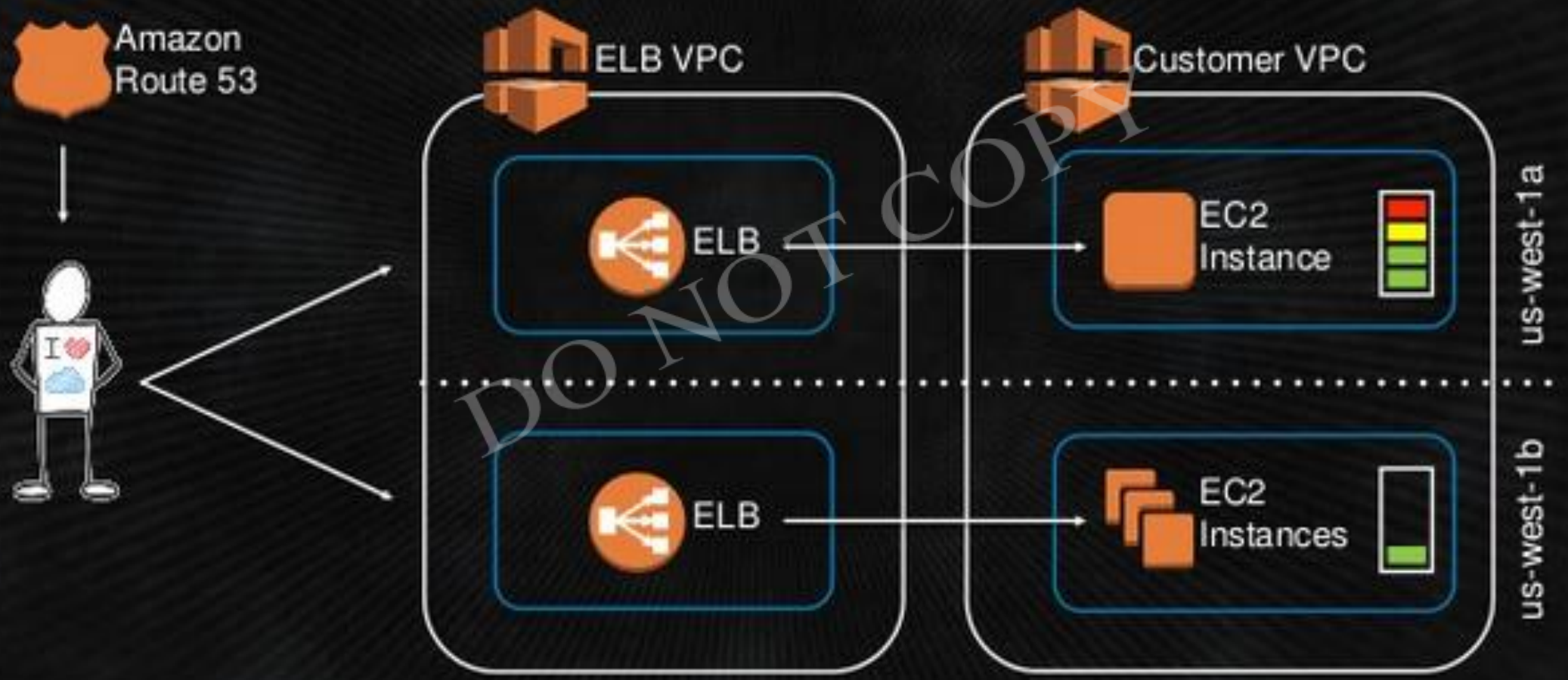
Multiple Availability Zones



Always associate two or more subnets in different zones with the load balancer

Using multiple Availability Zones does bring a few challenges.

Imbalanced Instance Capacity



Cross-Zone Load Balancing

Load balancer absorbs impact of DNS caching.

Eliminates imbalances in back-end instance utilization.

Requests distributed evenly across multiple Availability Zones.

Check connection limits before enabling.

No additional bandwidth charge for cross-zone traffic.

Understanding DNS

Each load balancer domain may contains multiple records.

Round robin used to balance traffic between Availability Zones.

DNS records will to change over time; never target IP addresses directly.

After being removed from DNS, IP addresses are drained and quarantined for up to 7 days.

SSL Offloading

Support for both SSL and HTTPs is provided.

Support for latest ciphers and protocols including Elliptical Curve Ciphers and Perfect Forward Secrecy.

Ability to fully customize ciphers and protocols to be used by each load balancer.

SSL Negotiation Suites provided to remove complexity of selecting ciphers and protocols.

Amazon CloudWatch Metrics

13 CloudWatch metrics provided for each load balancer.

Provide detailed insight into the health of the load balancer and application stack.

CloudWatch alarms can be configured to notify or take action should any metric go outside of the acceptable range.

All metrics provided at the 1-minute granularity.

CloudWatch and AutoScaling

All load balancer metrics can be used for AutoScaling.

Allow you to scale dynamically based on the load balancers view of the application.

Important to consider all metrics when using AutoScaling, may not be aware of resource contention on another metric.

You may be at peak multiple times a day.

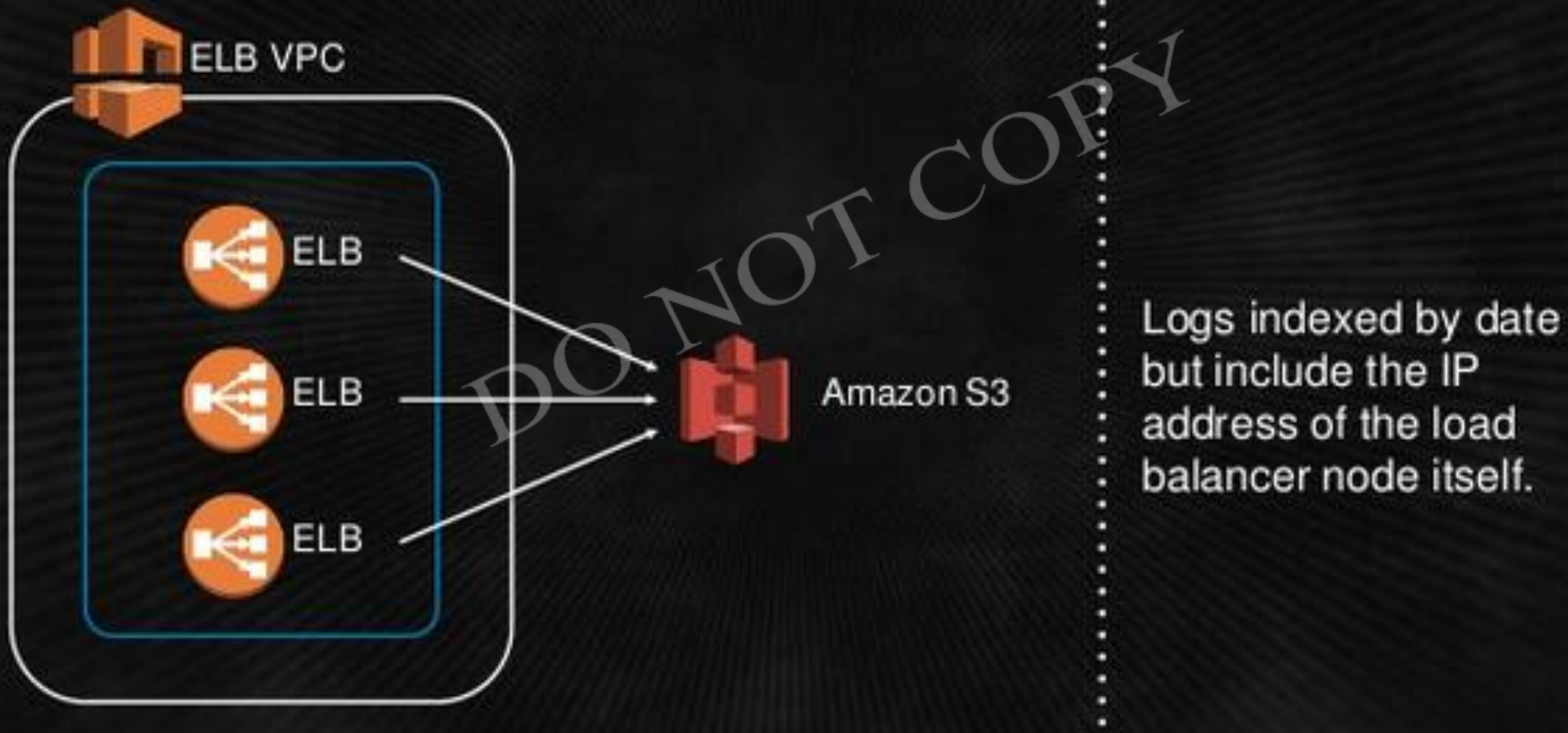
Access Logs

Provide detailed information on each request processed by the load balancer.

Includes request time, client IP address, latencies, request path, and server responses.

Delivered to an Amazon S3 bucket every 5 or 60 minutes.

Access Logs



Logs indexed by date but include the IP address of the load balancer node itself.

Access Logs

- timestamp
- elb name
- client:port
- backend:port
- request_processing_time
- backend_processing_time
- response_processing_time
- elb_status_code
- backend_state_code
- received_bytes
- sent_bytes
- “request”


```
2014-02-15T23:39:43.945958Z my-test-loadbalancer
192.168.131.39:2817 10.0.0.0.1 0.000073 0.001048 0.000057
200 200 0 29 "GET http://www.example.com:80/HTTP/1.1"
```

THANKYOU