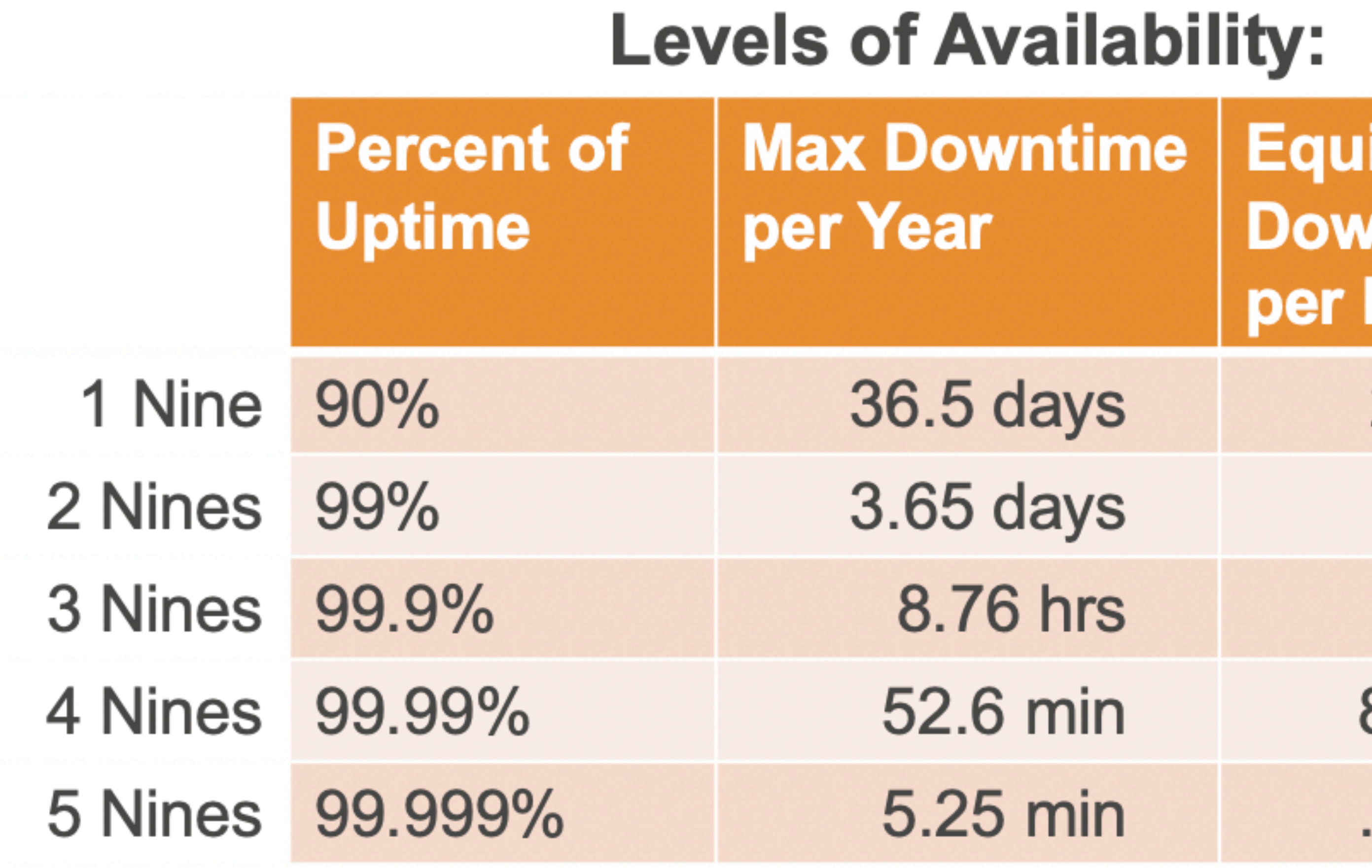
6 High Availability

Sunday, 15 September 2019 9:39 PM

ELB / Route53

“Everything fails, all the time.”

High availability (HA) is about ensuring that y downtime is minimized as much as possible, human intervention.



Levels of Availability:

	Percent of Uptime	Max Downtime per Year	Equi Dow per I
1 Nine	90%	36.5 days	2
2 Nines	99%	3.65 days	
3 Nines	99.9%	8.76 hrs	
4 Nines	99.99%	52.6 min	8
5 Nines	99.999%	5.25 min	.

your application's
, without the need for

ivalent ntime Day
2.4 hrs
14 min
86 sec
8.6 sec
86 sec

Avoid SPOF

Assume everything fails, and desi

Recovery Time Objective (**RTO**)

  How quickly must the system recover?

Recovery Point Objective (**RPO**)

  How much data can you afford to lose?

How much money do you need to inv
objectives?

Fault tolerance:
The built-in redundancy of an application's compo

Recoverability:
The process, policies, and procedures related to
catastrophic event.

Scalability:
The ability of an application to accommodate gro

gn backwards.

vest to meet those

onents.

restoring service after a

wth without changing

design.

HA on-premises **vs** HA on AWS

Ex: single RDS instance vs Multi-AZ

AWS services and high availabi

- Inherently HA services
- HA with right architecture

ELB

s

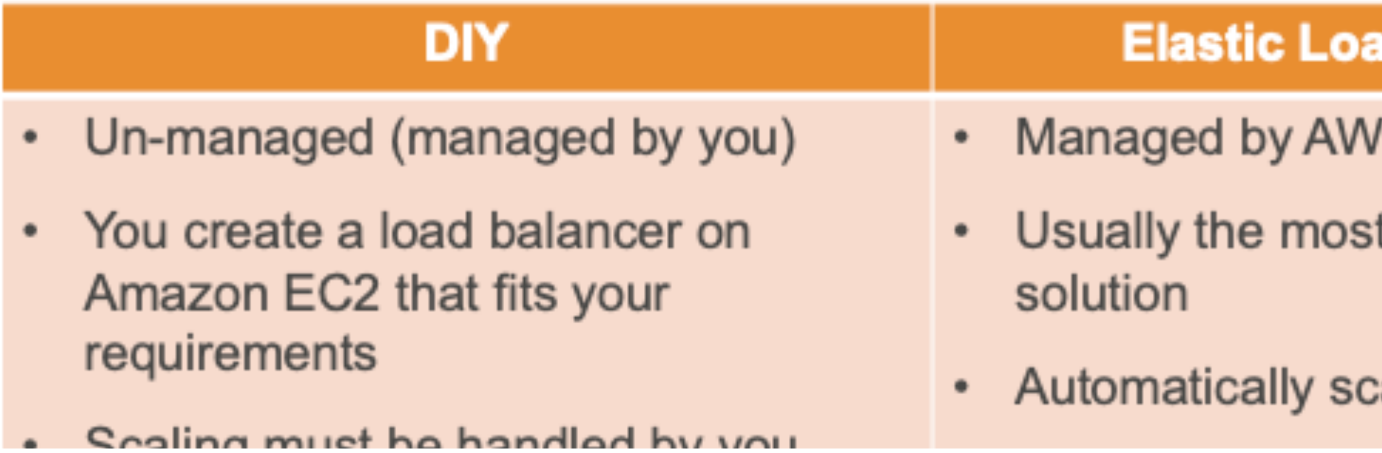
lity

A managed load balancing service that distributes traffic across multiple Amazon EC2 instances.

Elastic Load Balancing:

- Distributes load between instances.
- Recognizes and responds to unhealthy i
- Can be public or internal-facing.
- Uses HTTP, HTTPS, and TCP protocols
- Each load balancer is given a public DN
  - **Internet-facing** load balancers have DNS publicly resolve to the public IP addresses balancer's nodes.

Internal load balancers have DNS names w to the private IP addresses of the load balan



DIY	Elastic Loa
• Un-managed (managed by you) • You create a load balancer on Amazon EC2 that fits your requirements • Scaling must be handled by you	• Managed by AW • Usually the most solution • Automatically sc

s incoming application

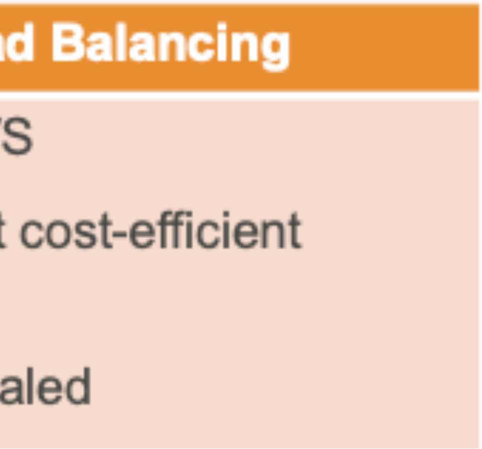
nstances.

.

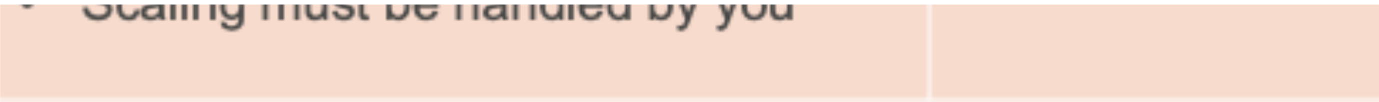
S name.

S names which
s of the load

hich publicly resolve
cer's nodes.



d Balancing
'S
t cost-efficient
aled



- Scaling must be handled by you

Elastic Load Balancing provides th features:

- Health checks
- Cross-zone load balancing
- Proxy Protocol
- Sticky sessions
- Connection draining

Classic Load Balancer

Application Load Balancer

Network Load Balancer

ne following

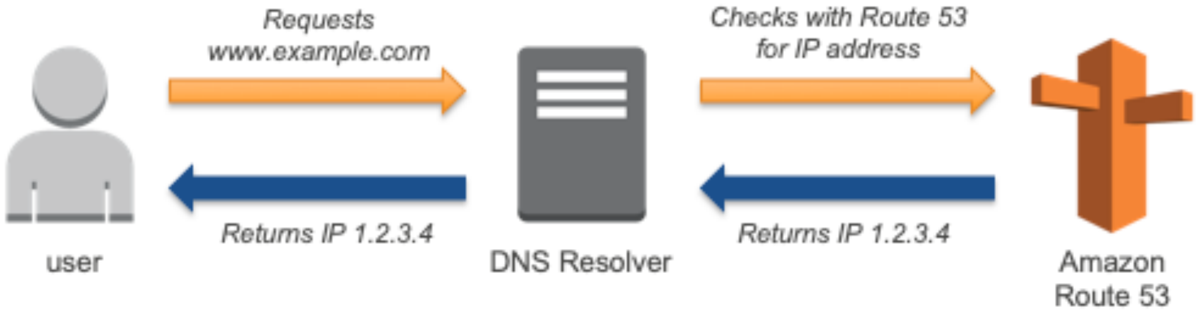
Improve the architecture exercis

se

EIP for HA

What about HA across regions?

Route 53 DNS Resolution

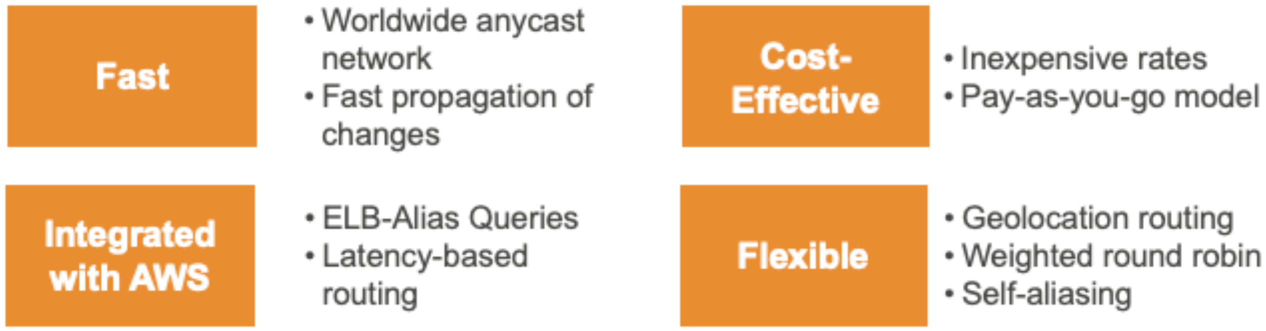


Reliable

- Redundant locations
- Backed with 100% Service Level Agreement (SLA)

Easy to Use

- Console
- Programmatic API
- Domain name management



Fast
- Worldwide anycast network
- Fast propagation of changes

Cost-Effective
- Inexpensive rates
- Pay-as-you-go model

Integrated with AWS
- ELB-Alias Queries
- Latency-based routing

Flexible
- Geolocation routing
- Weighted round robin
- Self-aliasing

Kind of Routing Route 53 suppo

- Simple routing: Single server environme
- Weighted round robin: Assign weights to sets to specify the frequency.
- Latency-based routing: Helps to improve applications.
- Health check and DNS failover: Fail over your primary site becomes unreachable.
- Geolocation routing: Specify **geographi** continent, by country, or by state in the U

orts?

nts.

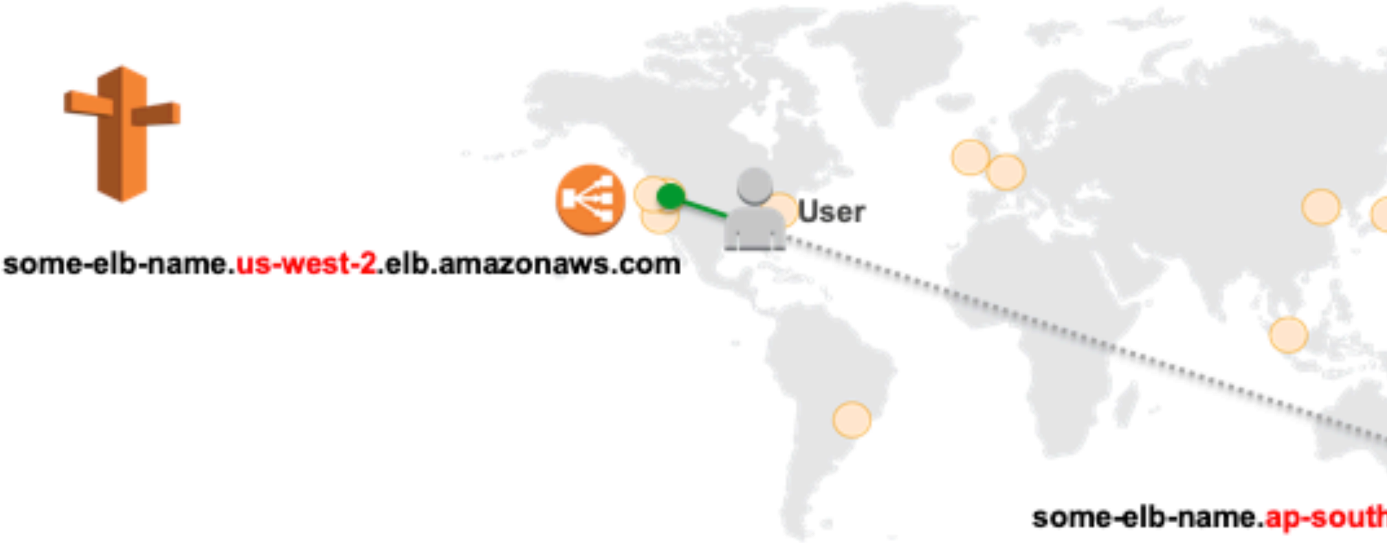
resource record

e your global

r to a backup site if

c locations by
United States.

Use Case: Multi-Region Deployment



some-elb-name.us-west-2.elb.amazonaws.com

some-elb-name.ap-south

Name	Type	Value
amgogreen.com	ALIAS	some-elb-name.us-west-2.elb.amazo
amgogreen.com	ALIAS	some-elb-name.ap-southeast-2.elb.a

Typical Architecture

Record Sets
CNAME www

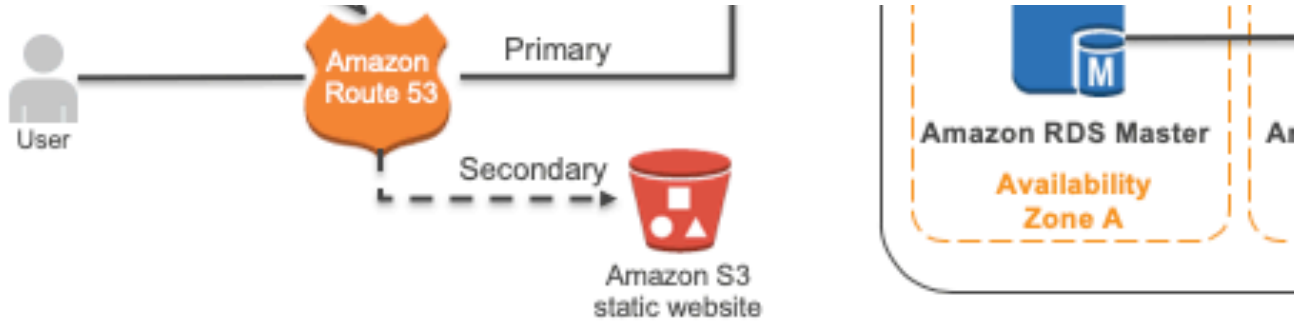
elastic_load_balancer
Routing Policy = Failover
Record Type = Primary

Amazon S3 website
Routing Policy = Failover
Record Type = Secondary

AWS
Amazon
w/CloudV

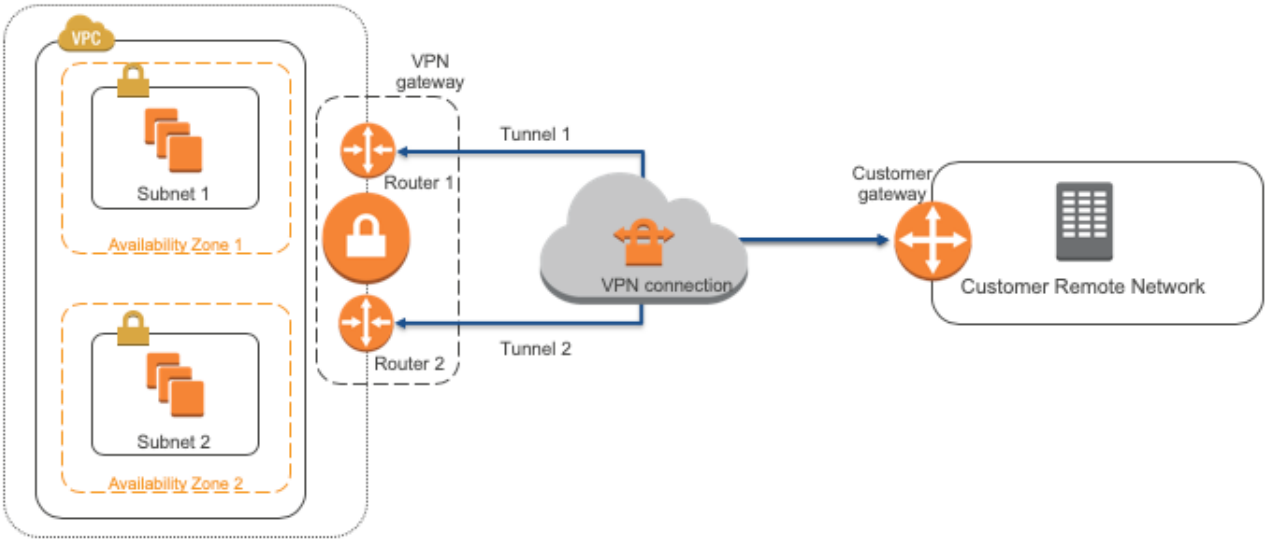
east-2.elb.amazonaws.com

naws.com
mazonaws.com



- Create a domain that uses Amazon Route 53 as the DNS service: see http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/
- Migrate an existing domain to Amazon Route 53: see http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/MigratingDNS.html
- Create a subdomain that uses Amazon Route 53 without migrating the parent do http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/CreatingNewSubdo
- Migrate a subdomain to Amazon Route 53 without migrating the parent domain: s http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/MigratingSubdoma

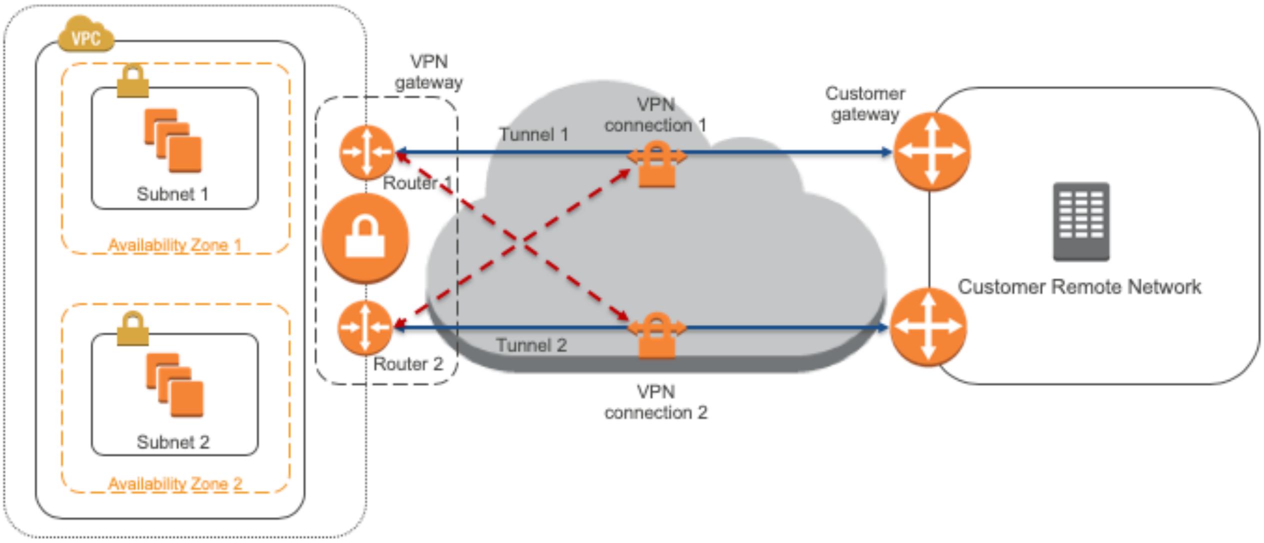
Create alias record sets for Elastic Load Balancing: http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-se

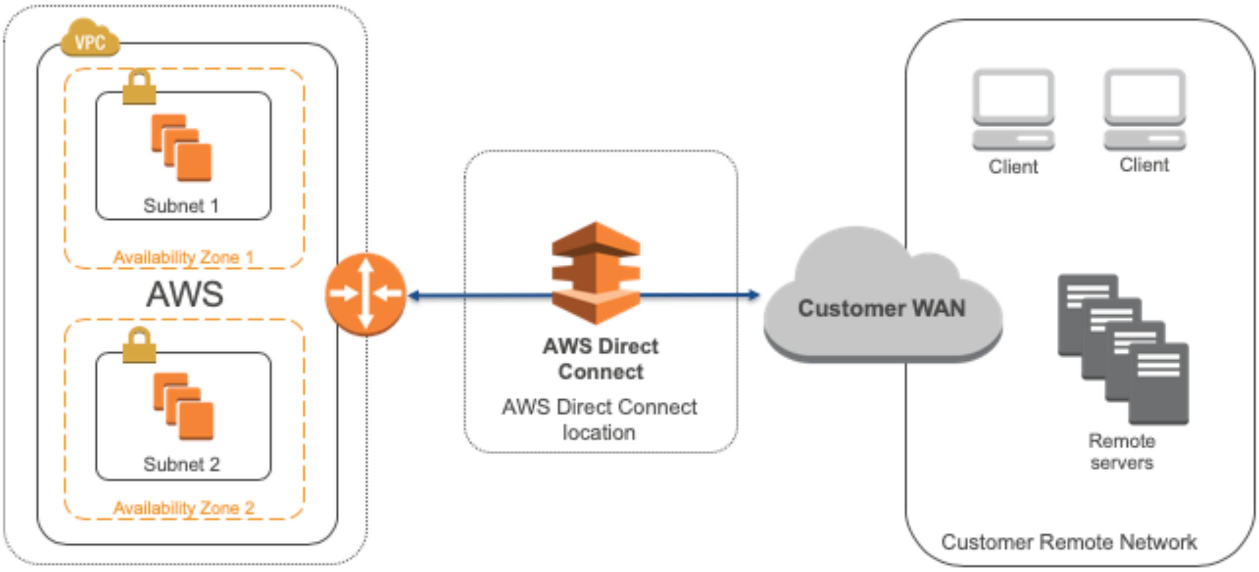


main: see
omain.html

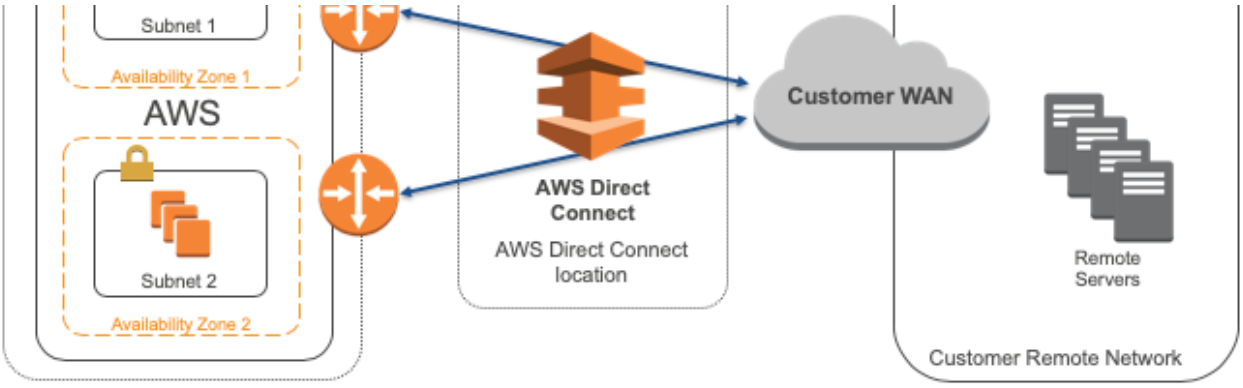
see
in.html

ts-choosing-alias-non-alias.html


VPC
Subnet 1
Availability Zone 1
Subnet 2
Availability Zone 2
VPN gateway
Router 1
Router 2
Tunnel 1
VPN connection 1
Tunnel 2
VPN connection 2
Customer gateway
Customer Remote Network

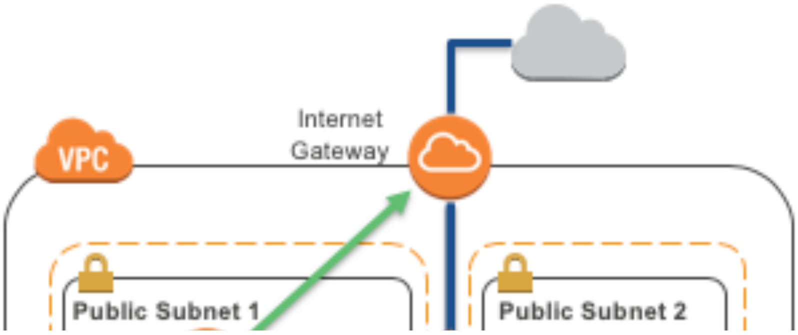

VPC
Subnet 1
Availability Zone 1
AWS
Subnet 2
Availability Zone 2
AWS Direct Connect
AWS Direct Connect location
Customer WAN
Client
Client
Remote servers
Customer Remote Network

VPC
Client
Client



Lab :

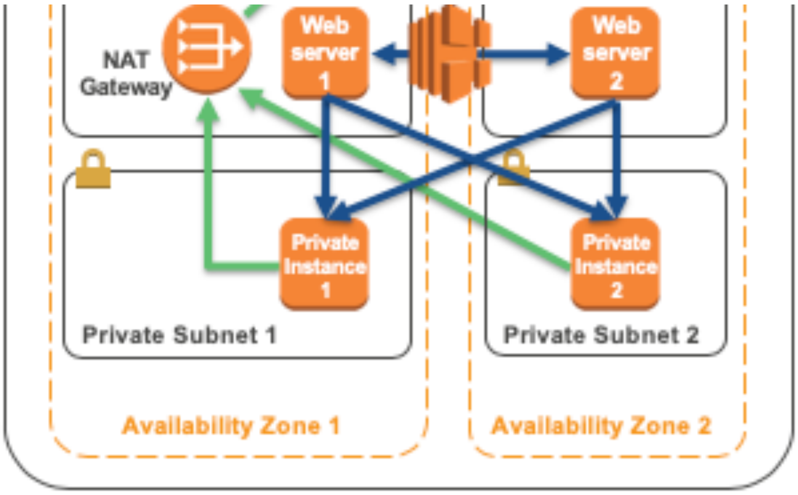
1. *Inspect the resources created automatically for yo*
2. *Start your web server's PHP application*
3. *Create an AMI of your web server.*
4. *Create a second public subnet in a second AZ an server in that subnet.*
5. *Create a load balancer and attach it to both subne*
6. *Create two private subnets with two instances ins*
7. *Create a NAT Gateway and route traffic from priv*
8. *Test the private instances' connections to the NA*



our lab.

nd launch a new web

ets.
side of them.
vate instances to it.
T Gateway.


NAT Gateway
Web server 1
Web server 2
Private Instance 1
Private Instance 2
Private Subnet 1
Private Subnet 2
Availability Zone 1
Availability Zone 2