Best practices for building a hybrid cloud architecture with AWS services

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Best practices for building a hybrid cloud architecture with AWS services

Table of Contents

Introduction **1**
Overview **3**
Hybrid cloud workshops 3
PoCs 3
Pillars 4
Prerequisites and limitations **5**
Prerequisites 5
AWS Outposts 5
AWS Local Zones 5
Limitations 6
AWS Outposts 6
AWS Local Zones 7
Hybrid cloud adoption process **8**
Networking at the edge 8
VPC architecture 8
Edge to Region traffic 9
Edge to on-premises traffic 12
Security at the edge 16
Data protection 16
Identity and access management 20
Infrastructure security 21
Internet access 22
Infrastructure governance 25
Resiliency at the edge 27
Infrastructure considerations 27
Networking considerations 29
Distributing instances across Outposts and Local Zones 33
Amazon RDS Multi-AZ in AWS Outposts 34
Failover mechanisms 35
Capacity planning at the edge 39
Capacity planning on Outposts 40
Capacity planning for Local Zones 40
Edge infrastructure management 41
Deploying services at the edge 41

Outposts-specific CLI and SDK 43
Resources 45
AWS references 45
AWS blog posts 45
Contributors 46
Authoring 46
Reviewing 46
Technical writing 46
Document history 47

Best practices for building a hybrid cloud architecture with AWS services

Leonardo Alberto Solano Alvarado, Tom Gadomski, Len Gomes, Obed Gutierrez, Dionysios Kakaletris, Matt Price, and Vamsi Krishna Yamavarapu, Amazon Web Services

Many businesses and organizations have adopted cloud computing as a key aspect of their technology strategy. They typically migrate their workloads to the AWS Cloud to increase agility, cost savings, performance, availability, resilience, and scalability. Most applications can be easily migrated, but some applications must remain on premises to take advantage of the low latency and local data processing of the on-premises environment, to avoid high data transfer costs, or for regulatory compliance. Furthermore, a subset of applications might need to be re-architected or modernized before they can be moved to the cloud. This leads many organizations to seek hybrid cloud architectures to integrate their on-premises and cloud operations to support a broad spectrum of use cases. This hybrid approach can provide the benefits of both on-premises and cloud-based computing, and can be particularly useful for edge computing scenarios.

When you build a hybrid cloud with AWS, we recommend that you determine your hybrid cloud strategy and your technical strategy:

- A **hybrid cloud strategy** provides guidelines that govern the consumption of cloud and on-premises resources to support your business objectives. This guidance describes common use cases for building a hybrid cloud, such as supporting ongoing migration to the cloud, ensuring business continuity during disasters, extending cloud infrastructure to the on-premises environment to support low-latency applications, or expanding your international presence on AWS. Defining this strategy helps you identify and define your business objectives for building a hybrid cloud, and provides guidelines for workload placements on the hybrid cloud.
- A **technical strategy** for the hybrid cloud identifies the guiding tenets of the hybrid cloud architecture and defines an implementation framework. This guidance outlines common requirements for a consistently deployed and managed hybrid cloud architecture to help you define principles for a planned hybrid cloud implementation. These requirements include standardized interfaces for resource provisioning and management across your cloud infrastructure.

This guide describes an operations and management framework to help solutions architects and operators identify the building blocks, best practices, and AWS hybrid cloud and in-Region services to implement a hybrid cloud with AWS.

Many organizations have used the solutions described in this guide to successfully deploy hybrid cloud environments that take advantage of the scale, agility, innovation, and global footprint provided by the AWS Cloud. (See case studies.) AWS hybrid cloud services deliver a consistent AWS experience from the cloud to on premises, and at the edge. Services such as AWS Outposts and AWS Local Zones place compute, storage, database, and other select AWS services close to large population and industry centers when you need low latency between end-user devices or existing on-premises data centers and workload servers.

Overview

This guide classifies AWS recommendations for the hybrid cloud into five pillars: networking, security, resiliency, capacity planning, and infrastructure management. It provides guidelines to help you improve your readiness and to develop a migration strategy by using an AWS hybrid edge service such as AWS Outposts or AWS Local Zones. We strongly recommend that you work with your AWS account team or AWS Partner to ensure that an AWS hybrid cloud specialist is available to assist you as you follow this guide and develop your process.

> **Note**
>
> Although AWS Outposts and Local Zones address similar problems, we recommend that you review use cases as well as the services and features available to decide which offering best suits your needs. For more information, see the AWS blog post AWS Local Zones and AWS Outposts, choosing the right technology for your edge workload.

Hybrid cloud workshops

With the assistance of an AWS hybrid cloud subject matter expert (SME), you can run a hybrid cloud workshop to assess your company's maturity level in relation to the five pillars discussed in this guide.

The workshop focuses on internal areas within your organization, such as networking, security, compliance, DevOps, virtualization, and business units. It helps you design a hybrid cloud architecture that meets your organization's requirements and defines implementation details, following the steps in the Hybrid cloud adoption process section of this guide.

PoCs

If you have specific requirements, you can use proofs of concept (PoCs) to validate functionality in Local Zones and AWS Outposts against those requirements.

AWS uses PoCs to help you test the workloads you want to move to an Outpost or Local Zone, to determine whether the workloads will be functional under the test architectures. To access a Local Zone for testing, follow the instructions in the Local Zones documentation. To test your

workload on AWS Outposts, work with your AWS account team or AWS Partner to access an AWS Outposts test laboratory and receive guidance from AWS solutions architects. In all scenarios, the development of a PoC requires you to generate a test document that contains:

- AWS services to use, such as Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Block Store (Amazon EBS), Amazon Virtual Private Cloud (Amazon VPC), and Amazon Elastic Kubernetes Service (Amazon EKS)
- Size and number of instances to consume (for example, `m5.xlarge` or `c5.2xlarge`)
- Test architecture diagram
- Test success criteria
- Details and objectives of each test to runIf you have specific requirements, you can use proofs of concept (PoCs) to validate functionality in Local Zones and AWS Outposts against those requirements.

Pillars

The next section covers prerequisites and limitations for using the architectures discussed in this guide. The sections after that cover the details of each pillar so that the recommendations document that you create during the hybrid cloud workshop can reflect the design details required for implementation.

- Networking at the edge
- Security at the edge
- Resiliency at the edge
- Capacity planning at the edge
- Edge infrastructure management

Prerequisites and limitations

Before you follow this guide, work with your AWS account team or AWS Partner to review the prerequisites and limitations for implementing edge architectures with AWS Outposts and Local Zones.

Prerequisites

AWS Outposts

- Your existing data center must meet the AWS Outposts requirements for facilities, networking, and power. AWS Outposts is designed to operate in a data center environment that has 5-15 kVA redundant power inputs, 145.8 times the kVA of cubic feet per minute (CFM) airflow, and an ambient temperature between 41° F (5° C) and 95° F (35° C), among other requirements.
- Confirm that the AWS Outposts service is available in your country by consulting the AWS Outposts rack FAQs. See the question: *In which countries and territories is Outposts rack available?*
- If your organization requires four or more AWS Outposts racks, your data center must meet the Aggregation, Core, Edge (ACE) rack requirements.
- An internet or AWS Direct Connect link of at least 500 Mbps (1 Gbps is better) must be provided and sustained to connect AWS Outposts to the AWS Region, with appropriate backup connectivity if your use case requires it. The round-trip time latency from AWS Outposts to the Region must be 175 milliseconds at the maximum.
- You must have an active contract for an AWS Enterprise Support or an AWS Unified Operations plan.

AWS Local Zones

- An AWS Local Zone must be available close to your data centers or users. See AWS Local Zones locations.
- Confirm that you have network connectivity from your on-premises infrastructure to the Local Zone:

- Option 1: An Direct Connect link from your data center to the Direct Connect point of presence (PoP) that's closest to the Local Zone. For more information, see Direct Connect in the Local Zones documentation.
- Option 2: An internet link in addition to an on-premises virtual private network (VPN) appliance and the necessary licensing to launch a software-based VPN appliance on Amazon EC2 in the Local Zone. For more information, see VPN connection in the Local Zones documentation.

For additional connectivity options, see the Local Zones documentation.

Limitations

AWS Outposts

- Amazon Relational Database Service (Amazon RDS) on AWS Outposts Multi-AZ deployments require customer-owned IP (CoIP) address pools. For more information, see Customer-owned IP addresses for Amazon RDS on AWS Outposts.
- Multi-AZ on AWS Outposts is available for all supported versions of MySQL and PostgreSQL on Amazon RDS on AWS Outposts. For more information, see Amazon RDS on AWS Outposts support for Amazon RDS features. Amazon RDS on AWS Outposts supports SQL Server, Amazon RDS for MySQL, and Amazon RDS for PostgreSQL databases.
- AWS Outposts isn't designed to operate when it's disconnected from an AWS Region. For more information, see the Thinking in terms of failure modes section in the AWS whitepaper *AWS Outposts High Availability Design and Architecture Considerations*.
- Amazon Simple Storage Service (Amazon S3) on AWS Outposts has some limitations. These are discussed in the How is Amazon S3 on Outposts different from Amazon S3? section of the *Amazon S3 on Outposts User Guide*.
- Application Load Balancers on AWS Outposts don't support mutual TLS (mTLS) or sticky sessions.
- The ACE racks aren't fully enclosed and don't include front or rear doors.
- The instance capacity tool is applicable only for new orders.

AWS Local Zones

- Local Zones don't have an AWS Site-to-Site VPN endpoint. Instead, use a software-based VPN on Amazon EC2.
- Local Zones don't support AWS Transit Gateway. Instead, connect to the Local Zone by using a Direct Connect Private virtual interface (VIF).
- Not all Local Zones support services such as Amazon RDS, Amazon FSx, Amazon EMR, or Amazon ElastiCache, or NAT gateways. For more information, see AWS Local Zones features.
- Application Load Balancers in Local Zones don't support mTLS or sticky sessions.

Hybrid cloud adoption process

The following sections discuss architectures and design details for each pillar of the AWS hybrid cloud:

- Networking at the edge
- Security at the edge
- Resiliency at the edge
- Capacity planning at the edge
- Edge infrastructure management

Networking at the edge

When you design solutions that use AWS edge infrastructure, such as AWS Outposts or Local Zones, you must carefully consider the network design. The network forms the foundation of connectivity for reaching workloads that are deployed in these edge locations, and is critical for ensuring low latency. This section outlines various aspects of hybrid edge connectivity.

VPC architecture

A virtual private cloud (VPC) spans all Availability Zones in its AWS Region. You can seamlessly extend any VPC in the Region to Outposts or Local Zones by using the AWS console or the AWS Command Line Interface (AWS CLI) to add an Outpost or Local Zone subnet. The following examples show how to create subnets in AWS Outposts and Local Zones by using the AWS CLI:

- **AWS Outposts**: To add an Outpost subnet to a VPC, specify the Amazon Resource Name (ARN) of the Outpost.

```
aws ec2 create-subnet --vpc-id vpc-081ec835f3EXAMPLE \
--cidr-block 10.0.0.0/24 \
--outpost-arn arn:aws:outposts:us-west-2:11111111111:outpost/op-0e32example1 \
--tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-subnet}]
```

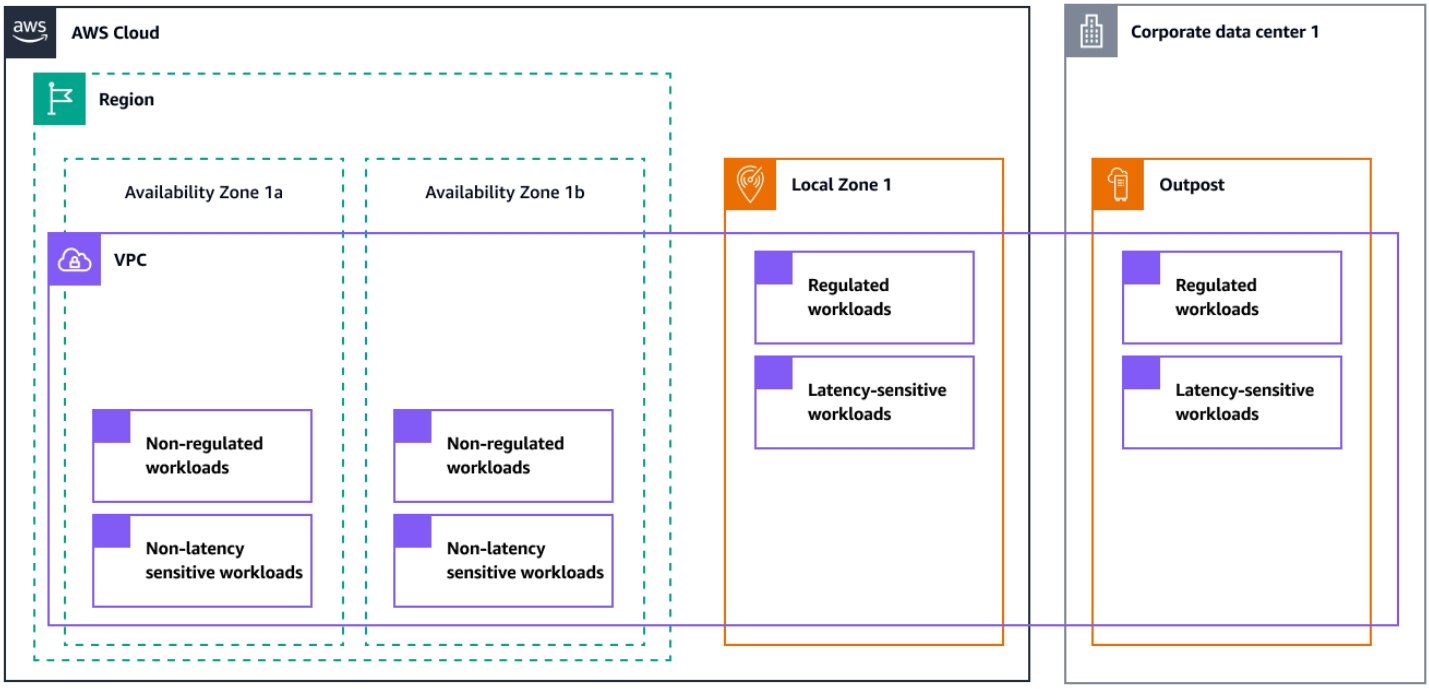
  For more information, see the AWS Outposts documentation.

- **Local Zones**: To add a Local Zone subnet to a VPC, follow the same procedure that you use with Availability Zones, but specify the Local Zone ID (`<local-zone-name>` in the following example).

```
aws ec2 create-subnet --vpc-id vpc-081ec835f3EXAMPLE \
--cidr-block 10.0.1.0/24 \
--availability-zone <local-zone-name> \
--tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-subnet}]
```

  For more information, see the Local Zones documentation.

The following diagram shows an AWS architecture that includes Outpost and Local Zone subnets.

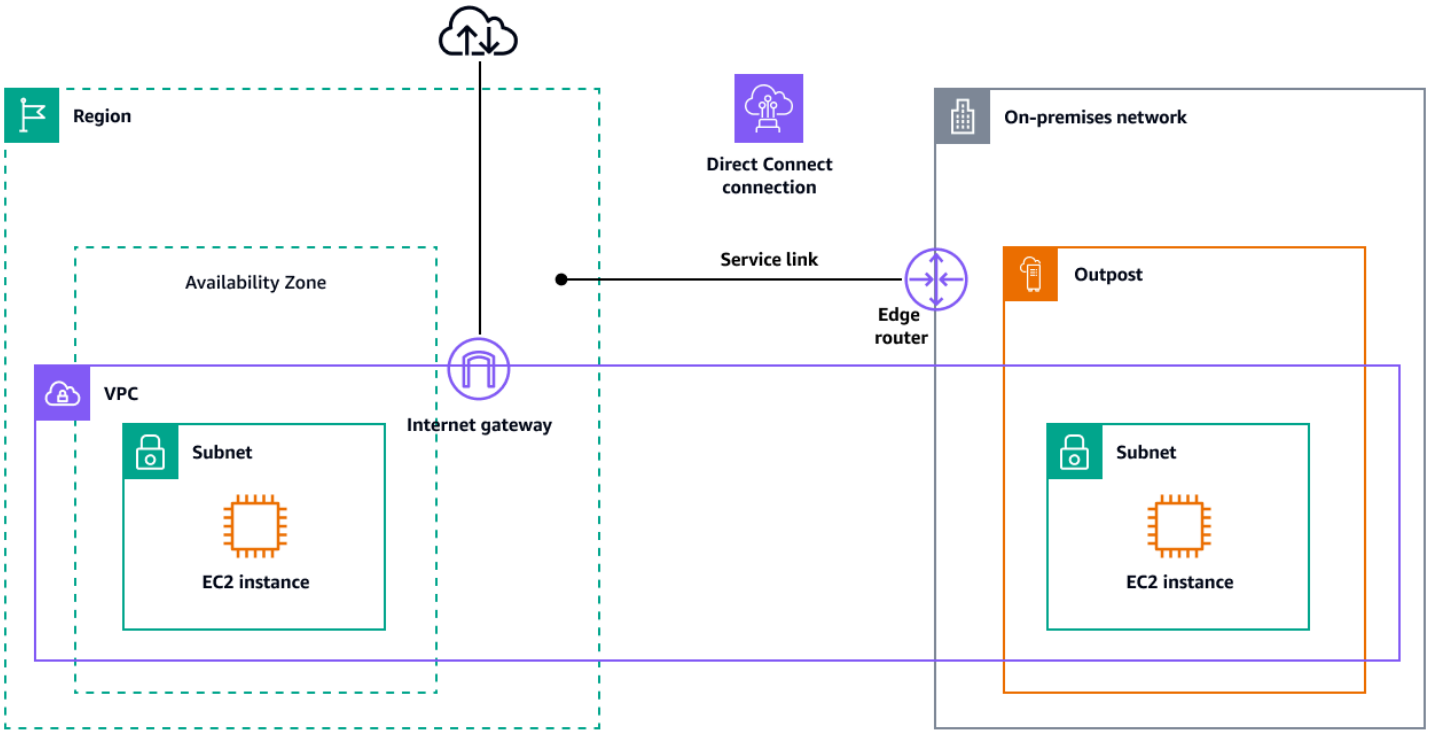


Edge to Region traffic

When you design a hybrid architecture by using services such as Local Zones and AWS Outposts, consider both control flows and data traffic flows between the edge infrastructures and AWS Regions. Depending on the type of edge infrastructure, your responsibility might vary: Some infrastructures require you to manage the connection to the parent Region, whereas others handle this through the AWS global infrastructure. This section explores the control plane and data plane connectivity implications for Local Zones and AWS Outposts.

AWS Outposts control plane

AWS Outposts provides a networking construct called a *service link*. The service link is a required connection between AWS Outposts and the selected AWS Region or parent Region (also referred to as *home Region*). It enables the management of the Outpost and the exchange of traffic between the Outpost and AWS Region. The service link uses an encrypted set of VPN connections to communicate with the home Region. You must provide connectivity between AWS Outposts and the AWS Region either through an internet link or an Direct Connect public virtual interface (public VIF), or through an Direct Connect private virtual interface (private VIF). For an optimal experience and resiliency, AWS recommends that you use redundant connectivity of at least 500 Mbps (1 Gbps is better) for the service link connection to the AWS Region. The minimum 500 Mbps service link connection allows you to launch Amazon EC2 instances, attach Amazon EBS volumes, and access AWS services such as Amazon EKS, Amazon EMR, and Amazon CloudWatch metrics. The network must support a maximum transmission unit (MTU) of 1,500 bytes between the Outpost and the service link endpoints in the parent AWS Region. For more information, see AWS Outposts connectivity to AWS Regions in the Outposts documentation.



For information about creating resilient architectures for service links that use Direct Connect and the public internet, see the Anchor connectivity section in the AWS whitepaper *AWS Outposts High Availability Design and Architecture Considerations*.

AWS Outposts data plane

The data plane between AWS Outposts and the AWS Region is supported by the same service link architecture that is used by the control plane. The bandwidth of the data plane service link between AWS Outposts and the AWS Region should correlate with the amount of data that must be exchanged: The greater the data dependence, the greater the link bandwidth should be.

The bandwidth requirements vary depending on the following characteristics:

- The number of AWS Outposts racks and capacity configurations
- Workload characteristics such as AMI size, application elasticity, and burst speed needs
- VPC traffic to the Region

The traffic between EC2 instances in AWS Outposts and EC2 instances in the AWS Region has an MTU of 1,300 bytes. We recommend that you discuss these requirements with an AWS hybrid cloud specialist before you propose an architecture that has co-dependencies between the Region and AWS Outposts.

Local Zones data plane

The data plane between Local Zones and the AWS Region is supported through the AWS global infrastructure. The data plane is extended through a VPC from the AWS Region to a Local Zone. Local Zones also provide a high-bandwidth, secure connection to the AWS Region, and enables you to seamlessly connect to the full range of Regional services through the same APIs and tool sets.

The following table shows the connection options and associated MTUs.

From	To	MTU
Amazon EC2 in Region	Amazon EC2 in Local Zones	1,300 bytes
Direct Connect	Local Zones	1,468 bytes
Internet gateway	Local Zones	1,500 bytes
Amazon EC2 in Local Zones	Amazon EC2 in Local Zones	9,001 bytes

Local Zones use the AWS global infrastructure to connect with AWS Regions. The infrastructure is fully managed by AWS, so you don't have to set up this connectivity. We recommend that you discuss your Local Zones requirements and considerations with an AWS hybrid cloud specialist before you design any architecture that has co-dependencies between the Region and Local Zones.

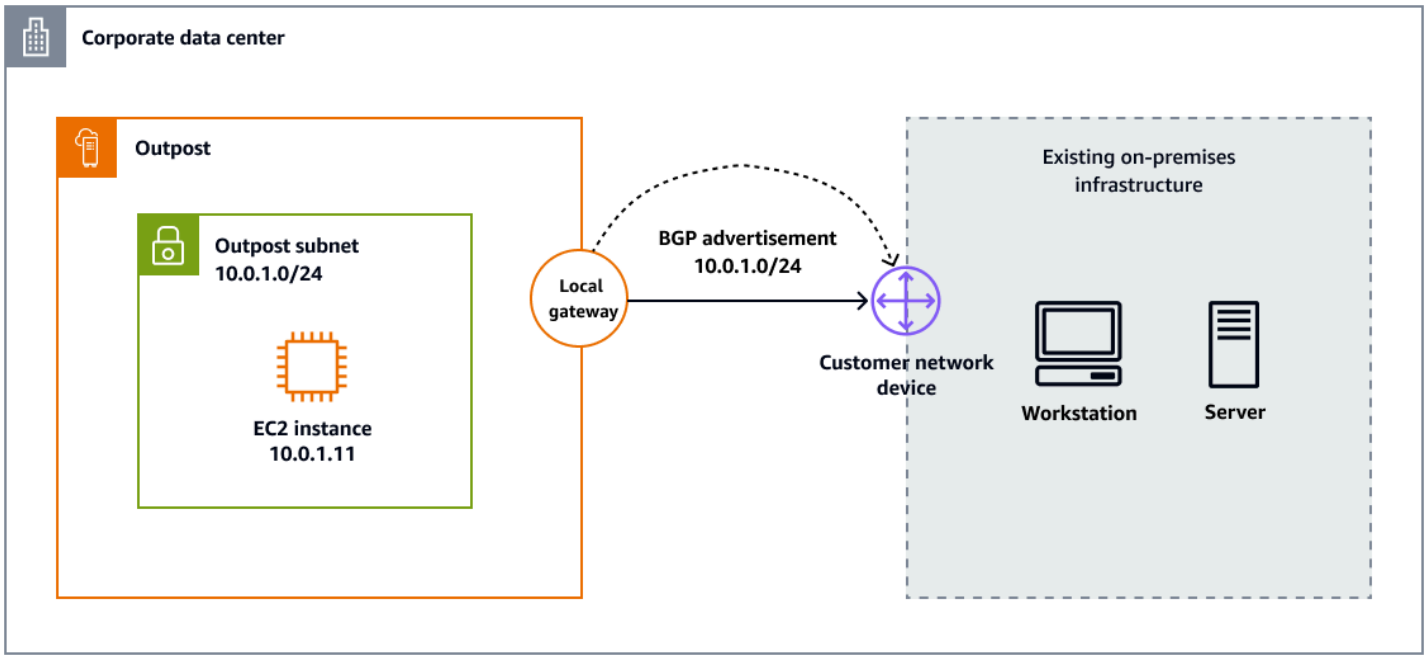
Edge to on-premises traffic

AWS hybrid cloud services are designed to address use cases that require low latency, local data processing, or data residency compliance. The network architecture for accessing this data is important, and it depends on whether your workload is running in AWS Outposts or Local Zones. Local connectivity also requires a well-defined scope, as discussed in the following sections.

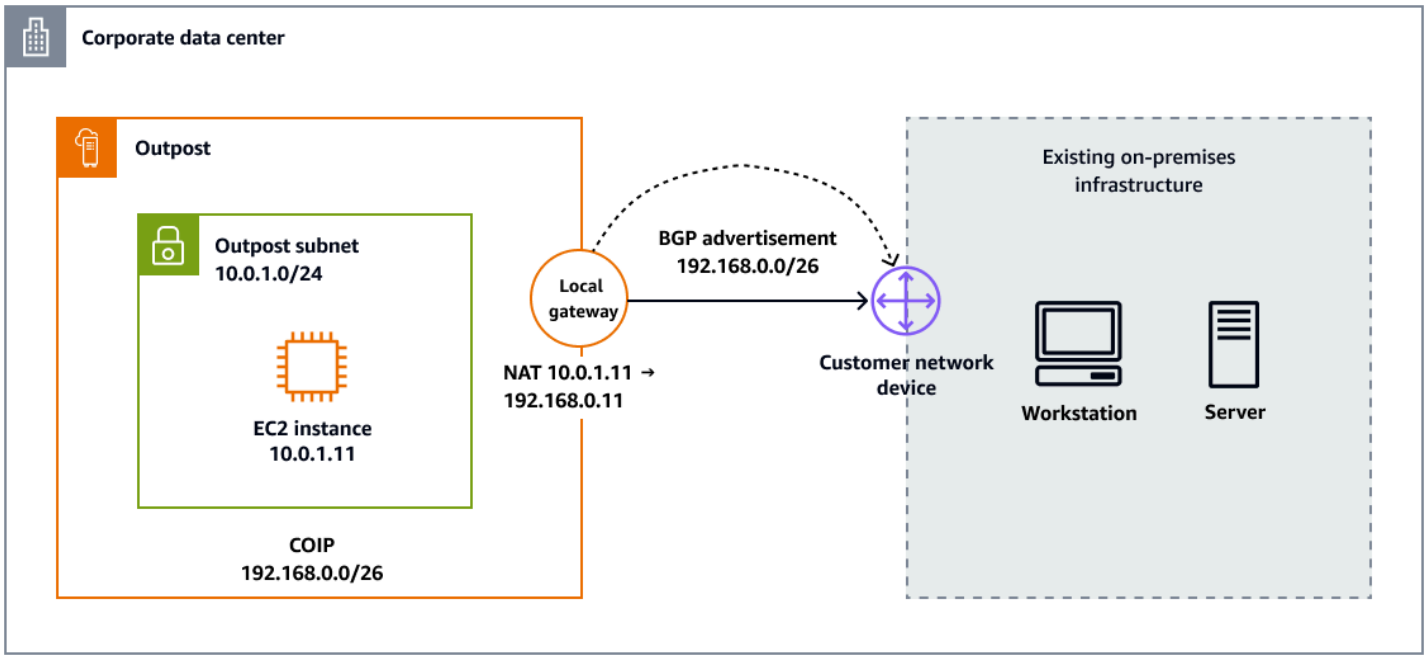
AWS Outposts local gateway

The local gateway (LGW) is a core component of the AWS Outposts architecture. The local gateway enables connectivity between your Outpost subnets and your on-premises network. The primary role of an LGW is to provide connectivity from an Outpost to your local on-premises network. It also provides connectivity to the internet through your on-premises network through either direct VPC routing or customer-owned IP addresses.

- **Direct VPC routing** uses the private IP address of the instances in your VPC to facilitate communication with your on-premises network. These addresses are advertised to your on-premises network with Border Gateway Protocol (BGP). Advertisement to BGP is only for the private IP addresses that belong to the subnets on your Outpost rack. This type of routing is the default mode for AWS Outposts. In this mode, the local gateway doesn't perform NAT for instances, and you do not need to assign Elastic IP addresses to your EC2 instances. The following diagram shows an AWS Outposts local gateway that uses direct VPC routing.



- With **customer-owned IP** addresses, you can provide an address range, known as a *customer-owned IP (CoIP) address pool*, which supports overlapping CIDR ranges and other network topologies. When you choose a CoIP, you must create an address pool, assign it to the local gateway route table, and advertise these addresses back to your network through BGP. CoIP addresses provide local or external connectivity to resources in your on-premises network. You can assign these IP addresses to resources on your Outpost, such as EC2 instances, by allocating a new Elastic IP address from the CoIP, and then assigning it to your resource. The following diagram shows an AWS Outposts local gateway that uses CoIP mode.



Local connectivity from AWS Outposts to a local network requires some parameter configurations, such as enabling the BGP routing protocol and advertising prefixes between the BGP peers. The MTU that can be supported between your Outpost and local gateway is 1,500 bytes. For more information, contact an AWS hybrid cloud specialist or review the AWS Outposts documentation.

Local Zones and the internet

Industries that require low-latency or local data residency (examples include gaming, live streaming, financial services, and the government) can use Local Zones to deploy and provide their applications to end users over the internet. During the deployment of a Local Zone, you must allocate public IP addresses for use in a Local Zone. When you allocate Elastic IP addresses, you can specify the location from which the IP address is advertised. This location is called a *network border group*. A network border group is a collection of Availability Zones, Local Zones, or AWS Wavelength Zones from which AWS advertises a public IP address. This helps ensure minimum latency or physical distance between the AWS network and the users who access the resources in these Zones.

To expose an Amazon EC2-hosted workload in a Local Zone to the internet, you can enable the **Auto-assign Public IP** option when you launch the EC2 instance. If you use an Application Load Balancer, you can define it as internet-facing so that public IP addresses assigned to the Local Zone can be propagated by the border network that's associated with the Local Zone. In addition, when you use Elastic IP addresses, you can associate one of these resources with an EC2 instance after

its launch. When you send traffic through an internet gateway in Local Zones, the same instance bandwidth specifications used by the Region are applied. Local Zone network traffic goes directly to the internet or to points of presence (PoPs) without traversing the Local Zone's parent Region, to enable access to low-latency computing.

Local Zones provide the following connectivity options over the internet:

- Public access: Connects workloads or virtual appliances to the internet by using Elastic IP addresses through an internet gateway.
- Outbound internet access: Enables resources to reach public endpoints through network address translation (NAT) instances or virtual appliances with associated Elastic IP addresses, without direct internet exposure.
- VPN connectivity: Establishes private connections by using Internet Protocol Security (IPsec) VPN through virtual appliances with associated Elastic IP addresses.

For more information, see Connectivity options for Local Zones in the Local Zones documentation.

Local Zones and Direct Connect

Local Zones also support Direct Connect, which lets you route your traffic over a private network connection. For more information, see Direct Connect in Local Zones in the Local Zones documentation.

Local Zones and transit gateways

AWS Transit Gateway doesn't support direct VPC attachments to Local Zone subnets. However, you can connect to Local Zone workloads by creating Transit Gateway attachments in the parent Availability Zone subnets of the same VPC. This configuration enables interconnectivity between multiple VPCs and your Local Zone workloads. For more information, see Transit gateway connection between Local Zones in the Local Zones documentation.

Local Zones and VPC peering

You can extend any VPC from a parent Region into a Local Zone by creating a new subnet and assigning it to the Local Zone. VPC peering can be established between VPCs that are extended to Local Zones. When the peered VPCs are in the same Local Zone, traffic stays within the Local Zone and does not hairpin through the parent Region.

Security at the edge

In the AWS Cloud, security is the top priority. As organizations adopt the scalability and flexibility of the cloud, AWS helps them adopt security, identity, and compliance as key business factors. AWS integrates security into its core infrastructure and offers services to help you meet your unique cloud security requirements. When you expand the scope of your architecture into the AWS Cloud, you benefit from the integration of infrastructures such as Local Zones and Outposts into AWS Regions. This integration enables AWS to extend a select group of core security services to the edge.

Security is a shared responsibility between AWS and you. The AWS shared responsibility model differentiates between the security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of AWS security as part of AWS compliance programs.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors, including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

Data protection

The AWS shared responsibility model applies to data protection in AWS Outposts and AWS Local Zones. As described in this model, AWS is responsible for protecting the global infrastructure that runs the AWS Cloud (*security of the cloud*). You are responsible for maintaining control over your content that is hosted on this infrastructure (*security in the cloud*). This content includes the security configuration and management tasks for the AWS services that you use.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS Identity and Access Management (IAM) or AWS IAM Identity Center. This gives each user only the permissions necessary to fulfill their job duties.

Encryption at rest

Encryption in EBS volumes

With AWS Outposts, all data is encrypted at rest. The key material is wrapped with an external key, the Nitro Security Key (NSK), which is stored in a removable device. The NSK is required to decrypt the data on your Outpost rack. You can use Amazon EBS encryption for your EBS volumes and snapshots. Amazon EBS encryption uses AWS Key Management Service (AWS KMS) and KMS keys.

In the case of Local Zones, all EBS volumes are encrypted by default in all Local Zones, except to for the list documented in the AWS Local Zones FAQ (see the question: *What's the default encryption behavior of EBS volumes in Local Zones?*), unless encryption is enabled for the account.

Encryption in Amazon S3 on Outposts

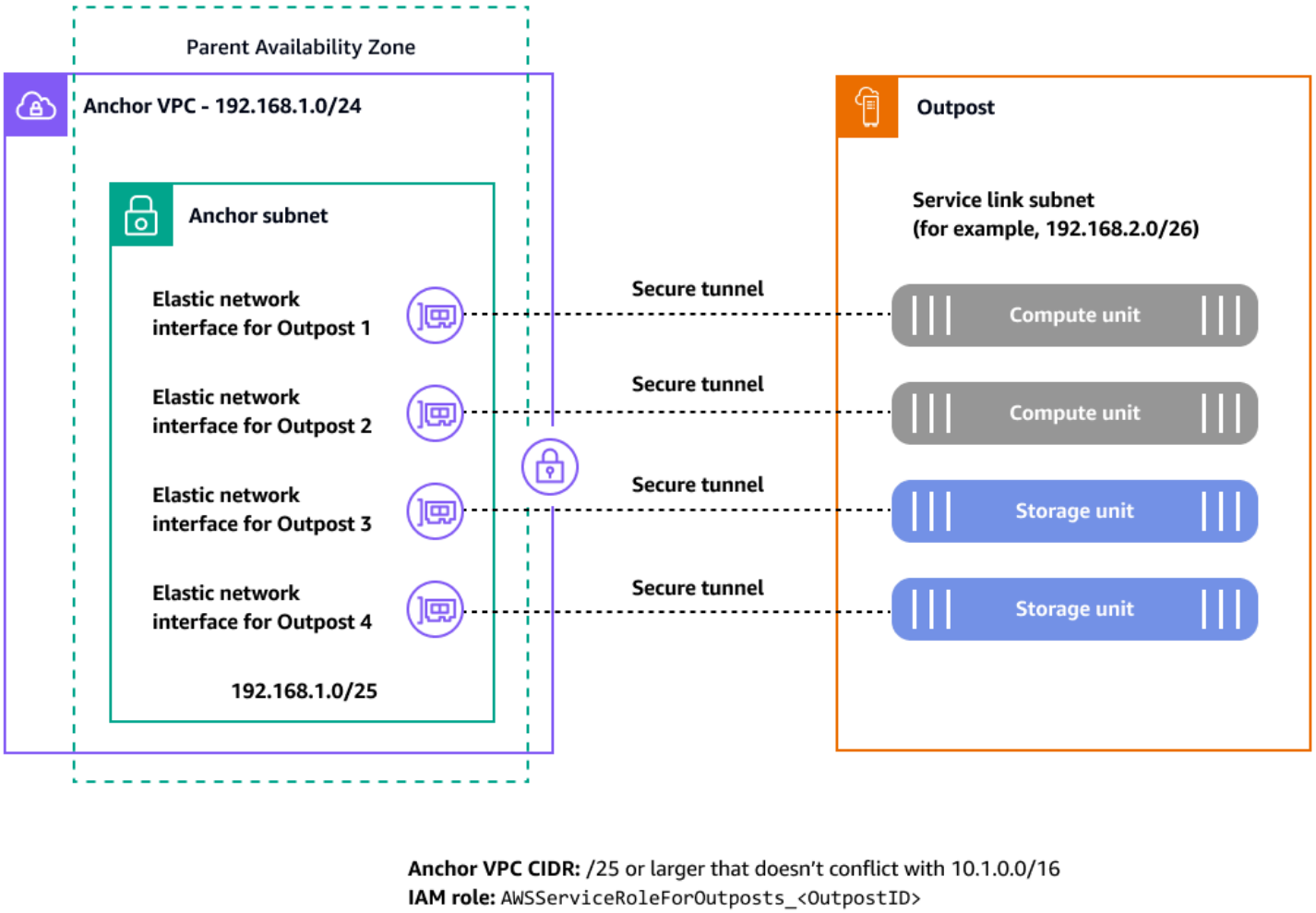
By default, all data stored in Amazon S3 on Outposts is encrypted by using server-side encryption with Amazon S3 managed encryption keys (SSE-S3). You can optionally use server-side encryption with customer-provided encryption keys (SSE-C). To use SSE-C, specify an encryption key as part of your object API requests. Server-side encryption encrypts only the object data, not the object metadata.

> **Note**
>
> Amazon S3 on Outposts doesn't support server-side encryption with KMS keys (SSE-KMS).

Encryption in transit

For AWS Outposts, the service link is a necessary connection between your Outposts server and your chosen AWS Region (or home Region) and allows for the management of the Outpost and the exchange of traffic to and from the AWS Region. The service link uses an AWS managed VPN to communicate with the home Region. Each host inside AWS Outposts creates a set of VPN tunnels to split control plane traffic and VPC traffic. Depending on the service link connectivity (internet or Direct Connect) for AWS Outposts, those tunnels require firewall ports to be opened for the service link to create the overlay on top of it. For detailed technical information about the security of AWS Outposts and the service link, see Connectivity through service link and Infrastructure security in AWS Outposts in the AWS Outposts documentation.

The AWS Outposts service link creates encrypted tunnels that establish control plane and data plane connectivity to the parent AWS Region, as illustrated in the following diagram.



Each AWS Outposts host (compute and storage) requires these encrypted tunnels over well-known TCP and UDP ports to communicate with its parent Region. The following table shows the source and destination ports and addresses for the UDP and TCP protocols.

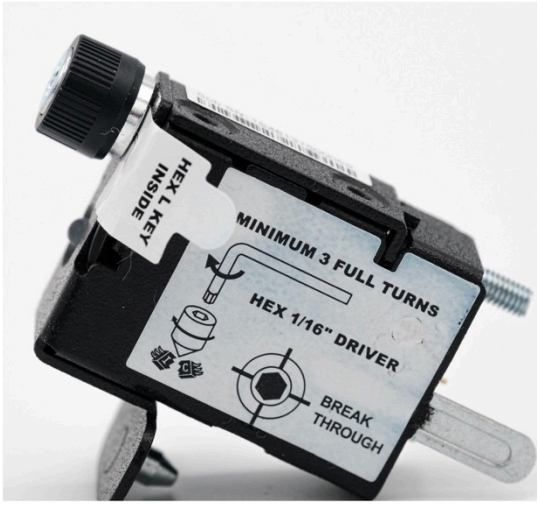
Protocol	Source port	Source address	Destination port	Destination address
UDP	443	AWS Outposts service link /26	443	AWS Outposts Region's public routes or anchor VPC CIDR
TCP	1025-65535	AWS Outposts service link /26	443	AWS Outposts Region's public

routes or anchor VPC CIDR

Local Zones are also connected to the parent Region through the redundant and very high-bandwidth global private backbone of Amazon. This connection gives applications that are running in Local Zones fast, secure, and seamless access to other AWS services. As long as Local Zones are part of the AWS global infrastructure, all data flowing over the AWS global network is automatically encrypted at the physical layer before it leaves AWS secured facilities. If you have specific requirements to encrypt the data in transit between your on-premises locations and Direct Connect PoPs to access a Local Zone, you can enable MAC Security (MACsec) between your on-premises router or switch and the Direct Connect endpoint. For more information, see the AWS blog post Adding MACsec security to Direct Connect connections.

Data deletion

When you stop or terminate an EC2 instance in AWS Outposts, the memory allocated to it is scrubbed (set to zero) by the hypervisor before it is allocated to a new instance, and every block of storage is reset. Deleting data from the Outpost hardware involves the use of specialized hardware. The NSK is a small device, illustrated in the following photograph, that attaches to the front of every compute or storage unit in an Outpost. It is designed to provide a mechanism to prevent your data from being exposed from your data center or colocation site. Data on the Outpost device is protected by wrapping keying material used to encrypt the device and storing the wrapped material on the NSK. When you return an Outpost host, you destroy the NSK by turning a small screw on the chip that crushes the NSK and physically destroys the chip. Destroying the NSK shreds the data cryptographically on your Outpost.



Identity and access management

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use AWS Outposts resources. If you have an AWS account, you can use IAM at no additional charge.

The following table lists the IAM features that you can use with AWS Outposts.

IAM feature	AWS Outposts support
Identity-based policies	Yes
Resource-based policies	Yes*
Policy actions	Yes
Policy resources	Yes
Policy condition keys (service-specific)	Yes
Access control lists (ACLs)	No
Attribute-based access control (ABAC) (tags in policies)	Yes
Temporary credentials	Yes

Principal permissions	Yes
Service roles	No
Service-linked roles	Yes

*In addition to IAM identity-based policies, Amazon S3 on Outposts supports both bucket and access point policies. These are resource-based policies that are attached to the Amazon S3 on Outposts resource.

For more information about how these features are supported in AWS Outposts, see the AWS Outposts user guide.

Infrastructure security

Infrastructure protection is a key part of an information security program. It ensures that workload systems and services are protected against unintended and unauthorized access, and potential vulnerabilities. For example, you define trust boundaries (for example, network and account boundaries), system security configuration and maintenance (for example, hardening, minimization, and patching), operating system authentication and authorizations (for example, users, keys, and access levels), and other appropriate policy-enforcement points (for example, web application firewalls or API gateways).

AWS provides a number of approaches to infrastructure protection, as discussed in the following sections.

Protecting networks

Your users might be part of your workforce or your customers, and can be located anywhere. For this reason, you can't trust everyone who has access to your network. When you follow the principle of applying security at all layers, you employ a zero trust approach. In the zero trust security model, application components or microservices are considered discrete, and no component or microservice trusts any other component or microservice. To achieve zero trust security, follow these recommendations:

- Create network layers. Layered networks help logically group similar networking components. They also shrink the potential scope of impact of unauthorized network access.

- Control traffic layers. Apply multiple controls with a defense-in-depth approach for both inbound and outbound traffic. This includes the use of security groups (stateful inspection firewalls), network ACLs, subnets, and route tables.
- Implement inspection and protection. Inspect and filter your traffic at each layer. You can inspect your VPC configurations for potential unintended access by using Network Access Analyzer. You can specify your network access requirements and identify potential network paths that do not meet them.

Protecting compute resources

Compute resources include EC2 instances, containers, AWS Lambda functions, database services, IoT devices, and more. Each compute resource type requires a different approach to security. However, these resources do share common strategies that you need to consider: *defense in depth*, *vulnerability management*, *reduction in attack surface*, *automation of configuration and operation*, and *performing actions at a distance*.

Here's general guidance for protecting your compute resources for key services:

- Create and maintain a vulnerability management program. Regularly scan and patch resources such as EC2 instances, Amazon Elastic Container Service (Amazon ECS) containers, and Amazon Elastic Kubernetes Service (Amazon EKS) workloads.
- Automate compute protection. Automate your protective compute mechanisms, including vulnerability management, reduction in attack surface, and management of resources. This automation frees up time that you can use to secure other aspects of your workload, and helps reduce the risk of human error.
- Reduce the attack surface. Reduce your exposure to unintended access by hardening your operating systems and minimizing the components, libraries, and externally consumable services that you use.

In addition, for each AWS service that you use, check the specific security recommendations in the service documentation.

Internet access

Both AWS Outposts and Local Zones provide architectural patterns that give your workloads access to and from the internet. When you use these patterns, consider internet consumption from the

Region a viable option only if you use it for patching, updating, accessing Git repositories that are external to AWS, and similar scenarios. For this architectural pattern, the concepts of centralized inbound inspection and centralized internet egress apply. These access patterns use AWS Transit Gateway, NAT gateways, network firewalls, and other components that reside in AWS Regions, but are connected to AWS Outposts or Local Zones through the data path between the Region and the edge.

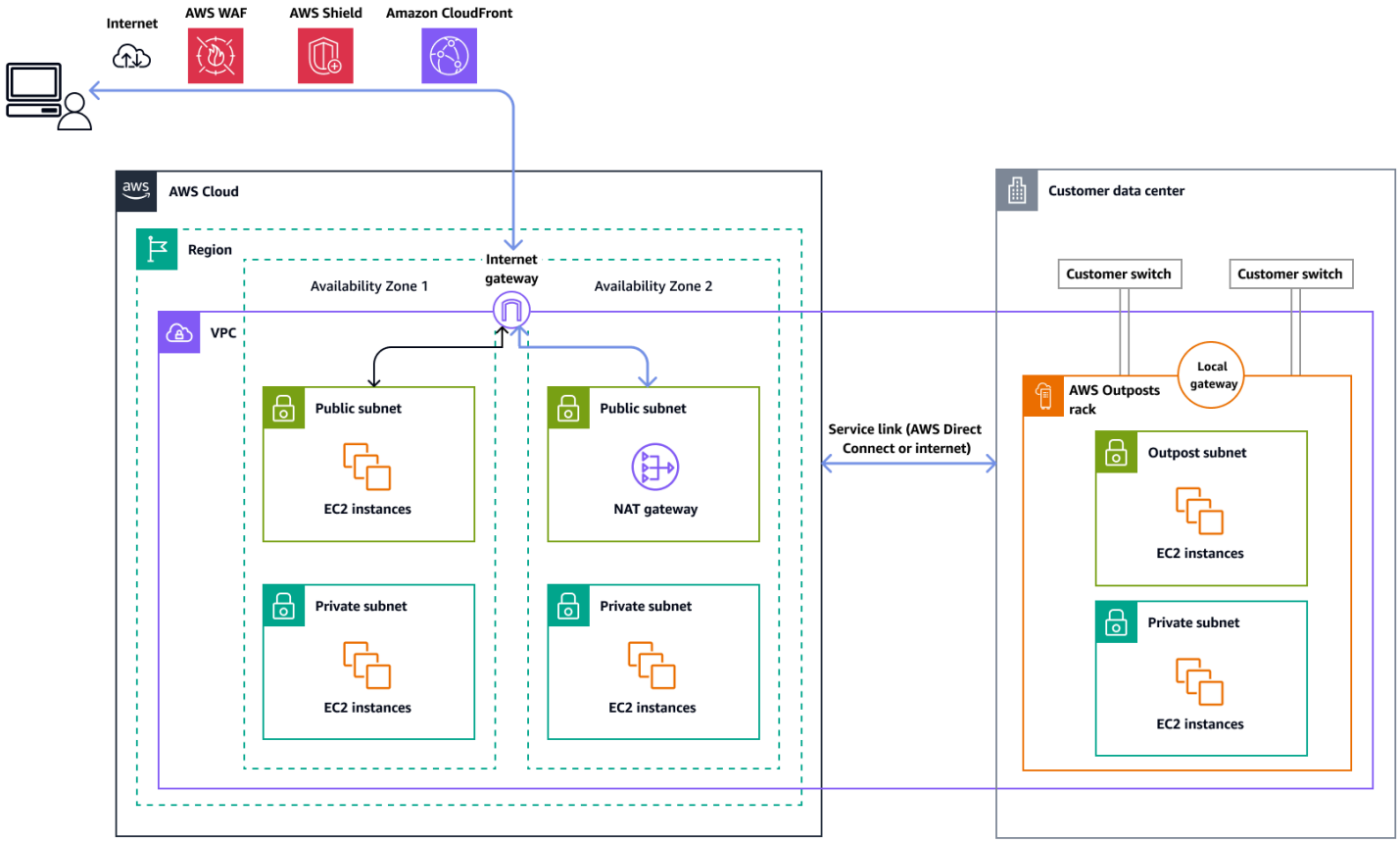
Local Zones adopts a network construct called a *network border group*, which is used in AWS Regions. AWS advertises public IP addresses from these unique groups. A network border group consists of Availability Zones, Local Zones, or Wavelength Zones. You can explicitly allocate a pool of public IP addresses for use in a network border group. You can use a network border group to extend the internet gateway to Local Zones by allowing Elastic IP addresses to be served from the group. This option requires that you deploy other components to complement the core services available in Local Zones. Those components might come from ISVs and help you build inspection layers in your Local Zone, as described in the AWS blog post Hybrid inspection architectures with AWS Local Zones.

In AWS Outposts, if you want to use the local gateway (LGW) to reach the internet from your network, you must modify the custom route table that's associated with the AWS Outposts subnet. The route table must have a default route entry (0.0.0.0/0) that uses the LGW as the next hop. You are responsible for implementing the remaining security controls in your local network, including perimeter defenses such as firewalls and intrusion prevention systems or intrusion detection systems (IPS/IDS). This aligns with the shared responsibility model, which divides security duties between you and the cloud provider.

Internet access through the parent AWS Region

In this option, the workloads in the Outpost access the internet through the service link and the internet gateway in the parent AWS Region. Outbound traffic to the internet can be routed through the NAT gateway that's instantiated in your VPC. For additional security for your ingress and egress traffic, you can use AWS security services such as AWS WAF, AWS Shield, and Amazon CloudFront in the AWS Region.

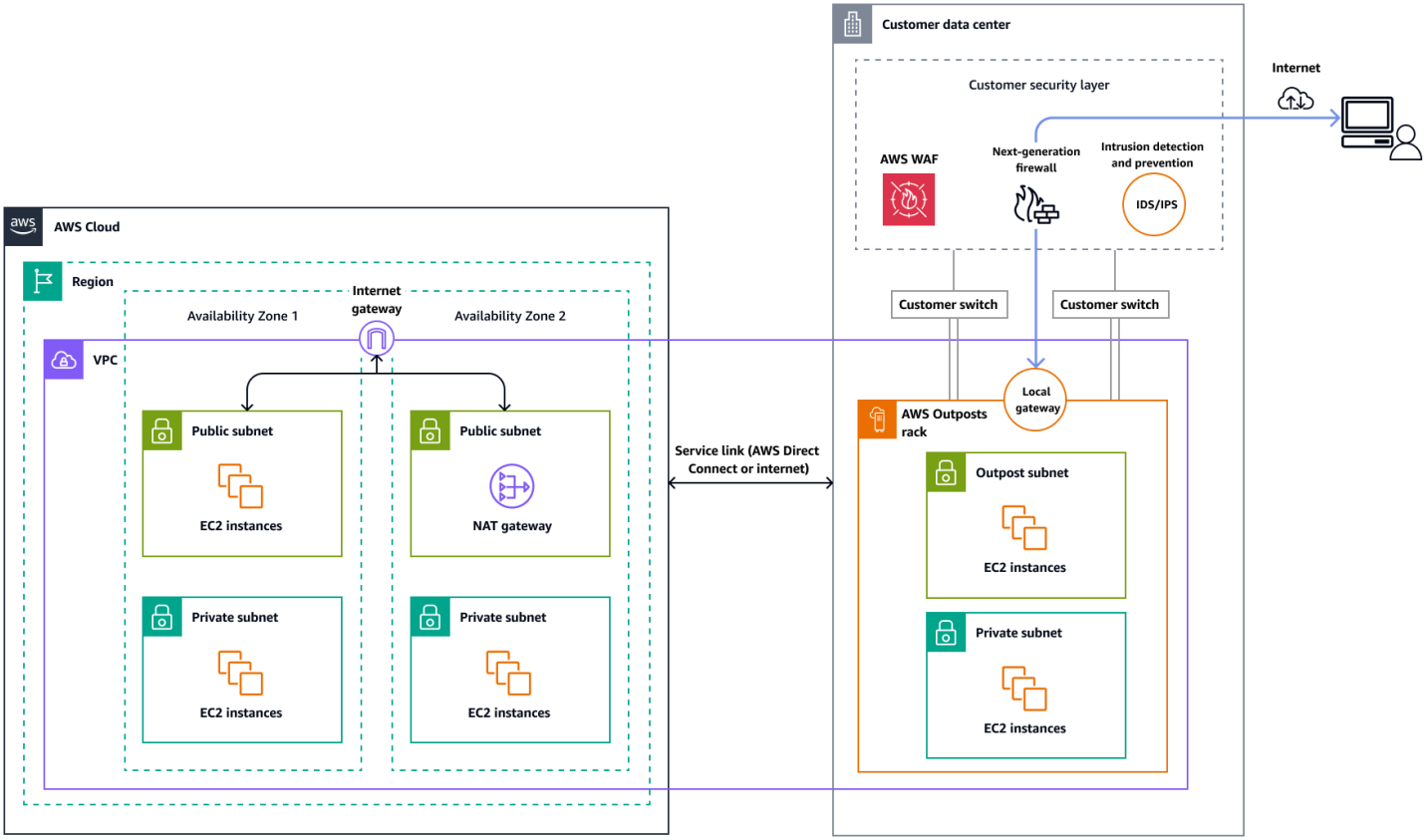
The following diagram shows traffic between the workload in the AWS Outposts instance and the internet going through the parent AWS Region.



Internet access through your local data center's network

In this option, the workloads in the Outpost access the internet through your local data center. The workload traffic that accesses the internet traverses through your local internet point of presence and egresses locally. In this case, your local data center's network security infrastructure is responsible for securing the AWS Outposts workload traffic.

The following image shows traffic between a workload in the AWS Outposts subnet and the internet going through a data center.



Infrastructure governance

Regardless of whether your workloads are deployed in an AWS Region, Local Zone, or Outpost, you can use AWS Control Tower for infrastructure governance. AWS Control Tower offers a straightforward way to set up and govern an AWS multi-account environment, following prescriptive best practices. AWS Control Tower orchestrates the capabilities of several other AWS services, including AWS Organizations, AWS Service Catalog, and IAM Identity Center(see all integrated services) to build a landing zone in less than an hour. Resources are set up and managed on your behalf.

AWS Control Tower provides unified governance across all AWS environments, including Regions, Local Zones (low-latency extensions), and Outposts (on-premises infrastructure). This helps ensure consistent security and compliance across your entire hybrid cloud architecture. For more information, see the AWS Control Tower documentation.

You can configure AWS Control Tower and capabilities such as guardrails to comply with data residency requirements in governments and regulated industries such as Financial Services

Institutions (FSIs). To understand how to deploy guardrails for data residency at the edge, see the following:

- Best practices for managing data residency in AWS Local Zones using landing zone controls (AWS blog post)
- Architecting for data residency with AWS Outposts rack and landing zone guardrails (AWS blog post)
- Data Residency with Hybrid Cloud Services Lens (AWS Well-Architected Framework documentation)

Sharing Outposts resources

Because an Outpost is a finite infrastructure that lives in your data center or in a co-location space, for centralized governance of AWS Outposts, you need to centrally control which accounts AWS Outposts resources are shared with.

With Outpost sharing, Outpost owners can share their Outposts and Outpost resources, including Outpost sites and subnets, with other AWS accounts that are in the same organization in AWS Organizations. As an Outpost owner, you can create and manage Outpost resources from a central location, and share the resources across multiple AWS accounts within your AWS organization. This allows other consumers to use Outpost sites, configure VPCs, and launch and run instances on the shared Outpost.

Shareable resources in AWS Outposts are:

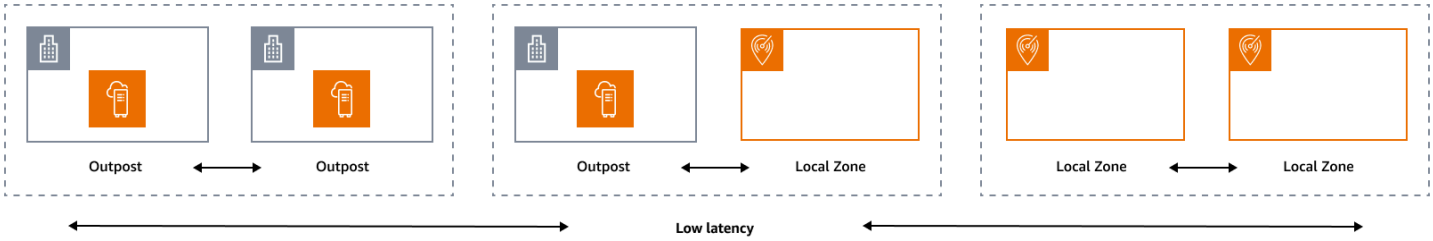
- Allocated dedicated hosts
- Capacity reservations
- Customer-owned IP (CoIP) address pools
- Local gateway route tables
- Outposts
- Amazon S3 on Outposts
- Sites
- Subnets

To follow the best practices for sharing Outposts resources in a multi-account environment, see the following AWS blog posts:

- Sharing AWS Outposts in a multi-account AWS environment: Part 1
- Sharing AWS Outposts in a multi-account AWS environment: Part 2

Resiliency at the edge

The reliability pillar encompasses the ability of a workload to perform its intended function correctly and consistently when it is expected to. This includes the ability to operate and test the workload through its lifecycle. In this sense, when you design a resilient architecture at the edge, you must first consider which infrastructures you will use to deploy that architecture. There are three possible combinations to implement by using AWS Local Zones and AWS Outposts: *Outpost to Outpost*, *Outpost to Local Zone*, and *Local Zone to Local Zone*, as illustrated in the following diagram. Although there are other possibilities for resilient architectures, such as combining AWS edge services with traditional on-premises infrastructure or AWS Regions, this guide focuses on these three combinations that apply to the design of hybrid cloud services



Infrastructure considerations

At AWS, one of the core principles of service design is to avoid single points of failure in the underlying physical infrastructure. Because of this principle, AWS software and systems use multiple Availability Zones and are resilient to the failure of a single zone. At the edge, AWS offers infrastructures that are based on Local Zones and Outposts. Therefore, a critical factor in ensuring resilience in infrastructure design is defining where an application's resources are deployed.

Local Zones

Local Zones act similarly to Availability Zones within their AWS Region, because they can be selected as a placement location for zonal AWS resources such as subnets and EC2 instances. However, they aren't located in an AWS Region, but near large population, industrial, and IT centers where no AWS Region exists today. Despite this, they still retain high-bandwidth, secure connections between local workloads in the Local Zone and workloads that are running in the AWS

Region. Therefore, you should use Local Zones to deploy workloads closer to your users for low-latency requirements.

Outposts

AWS Outposts is a fully managed service that extends AWS infrastructure, AWS services, APIs, and tools to your data center. The same hardware infrastructure that's used in the AWS Cloud is installed in your data center. Outposts are then connected to the nearest AWS Region. You can use Outposts to support your workloads that have low latency or local data processing requirements.

Parent Availability Zones

Each Local Zone or Outpost has a parent Region (also referred to as *home Region*). The parent Region is where the control plane of the AWS edge infrastructure (Outpost or Local Zone) is anchored. In the case of Local Zones, the parent Region is a fundamental architectural component of a Local Zone and cannot be modified by customers. AWS Outposts extends the AWS Cloud to your on-premises environment, so you must select a specific Region and Availability Zone during the ordering process. This selection anchors the control plane of your Outposts deployment to the chosen AWSinfrastructure.

When you develop high availability architectures in the edge, the parent Region of these infrastructures, such as Outposts or Local Zones, must be the same, so that a VPC can be extended between them. This extended VPC is the basis for creating these high-availability architectures. When you define a highly resilient architecture, this is why you must validate the parent Region and the Availability Zone of the Region where the service will be (or is) anchored. As illustrated in the following diagram, if you want to deploy a high availability solution between two Outposts, you must choose two different Availability Zones to anchor the Outposts. This allows for a Multi-AZ architecture from a control plane perspective. If you want to deploy a highly available solution that includes one or more Local Zones, you must first validate the parent Availability Zone where the infrastructure is anchored. For this purpose, use the following AWS CLI command:

```
aws ec2 describe-availability-zones --zone-ids use1-mia1-az1
```

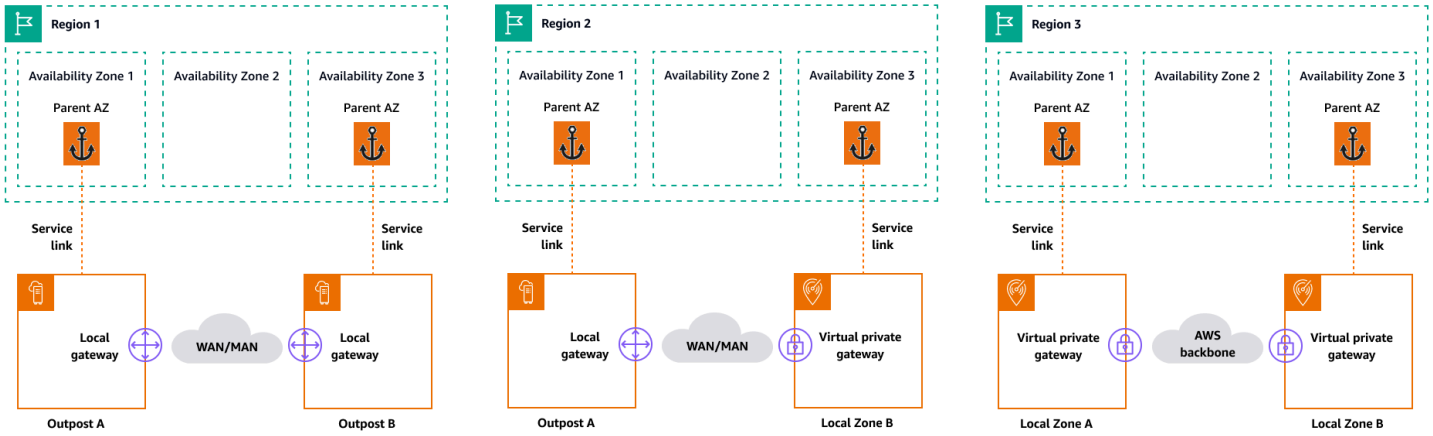
Output of the previous command:

```
{     "AvailabilityZones": [
          {
             "State": "available",
             "OptInStatus": "opted-in",
```

```
            "Messages": [],
            "RegionName": "us-east-1",
            "ZoneName": "us-east-1-mia-1a",
            "ZoneId": "use1-mia1-az1",
            "GroupName": "us-east-1-mia-1",
            "NetworkBorderGroup": "us-east-1-mia-1",
            "ZoneType": "local-zone",
            "ParentZoneName": "us-east-1d",
            "ParentZoneId": "use1-az2"
        }
    ]
}
```

In this example, the Miami Local Zone (`us-east-1d-mia-1a1`) is anchored in the `us-east-1d-az2` Availability Zone. Therefore, if you need to create a resilient architecture at the edge, you must ensure that the secondary infrastructure (either Outposts or Local Zones) is anchored to an Availability Zone other than `us-east-1d-az2`**.** For example, `us-east-1d-az1` would be valid.

The following diagram provides examples of highly available edge infrastructures.



Networking considerations

This section discusses initial considerations for networking at the edge, mainly for connections to access the edge infrastructure. It reviews valid architectures that provide a resilient network for the service link.

Resiliency networking for Local Zones

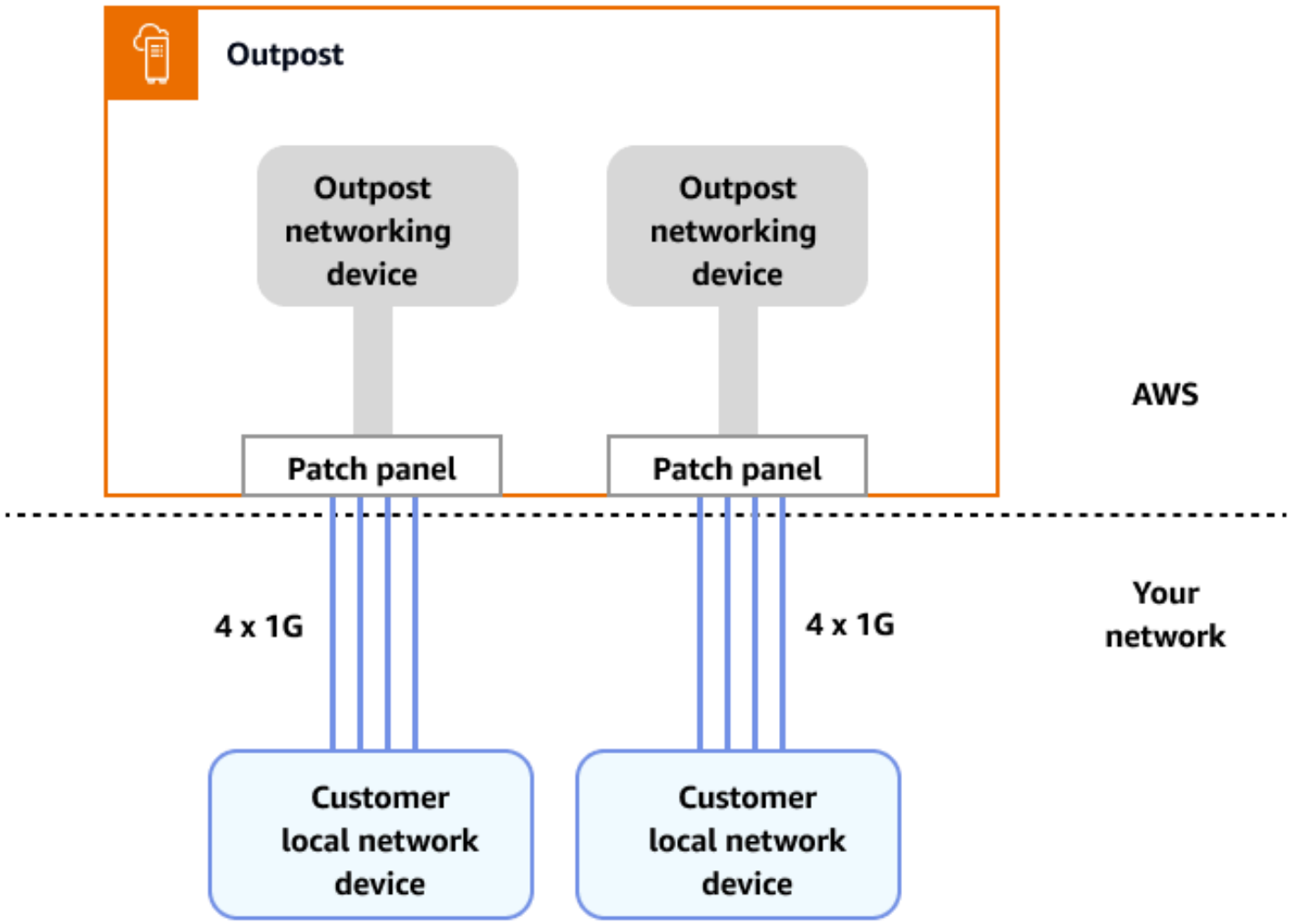
Local Zones are connected to the parent Region with multiple, redundant, secure, high-speed links that enable you to consume any Regional service, such as Amazon S3 and Amazon RDS, seamlessly.

You are responsible for providing connectivity from your on-premises environment or users to the Local Zone. Regardless of the connectivity architecture you choose (for example, VPN or Direct Connect), the latency that must be achieved through the network links must be equivalent to avoid any impact on application performance in the event of a failure in a main link. If you're using Direct Connect, the applicable resilience architectures are the same as those for accessing an AWS Region, as documented in Direct Connect resiliency recommendations. However, there are scenarios that apply mostly to international Local Zones. In the country where the Local Zone is enabled, having only a single Direct Connect PoP makes it impossible to create the architectures recommended for Direct Connect resilience. If you have access to only a single Direct Connect location or require resiliency beyond a single connection, you can create a VPN appliance on Amazon EC2 and Direct Connect, as illustrated and discussed in the AWS blog post Enabling highly available connectivity from on premises to AWS Local Zones.

Resiliency networking for Outposts

In contrast to Local Zones, Outposts have redundant connectivity for accessing workloads deployed in Outposts from your local network. This redundancy is achieved through two Outposts network devices (ONDs). Each OND requires at least two fiber connections at 1 Gbps, 10 Gbps, 40 Gbps, or 100 Gbps to your local network. These connections must be configured as a link aggregation group (LAG) to allow for the scalable addition of more links.

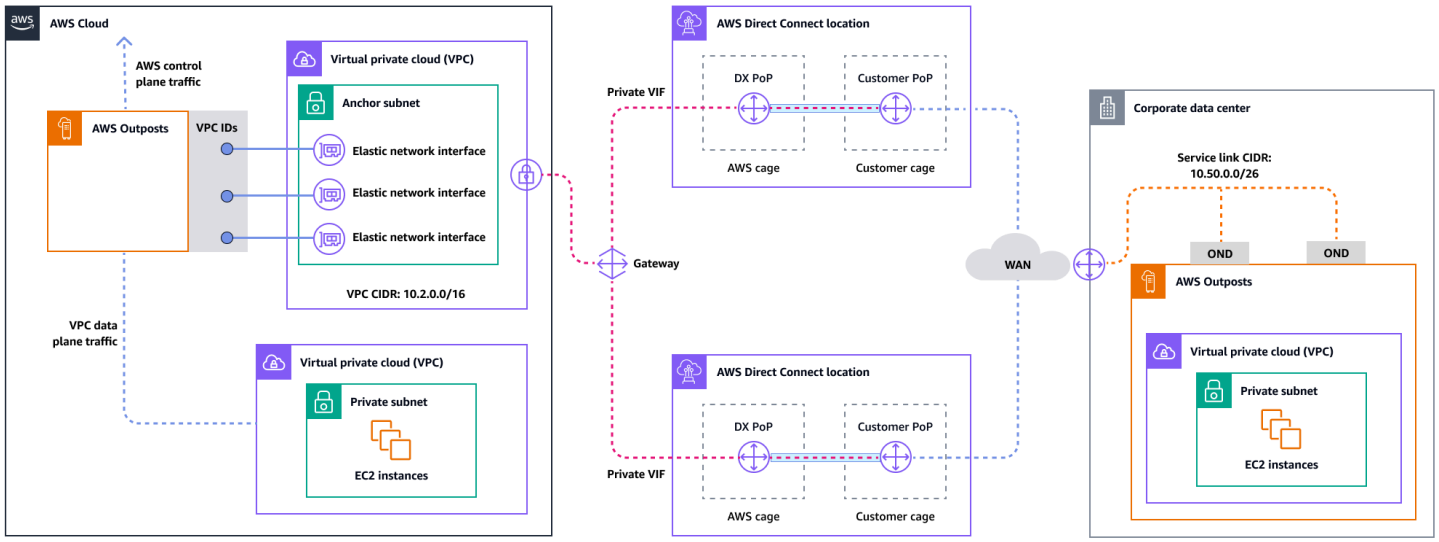
Uplink speed	Number of uplinks
1 Gbps	1, 2, 4, 6, or 8
10 Gbps	1, 2, 4, 8, 12, or 16
40 or 100 Gbps	1, 2, or 4



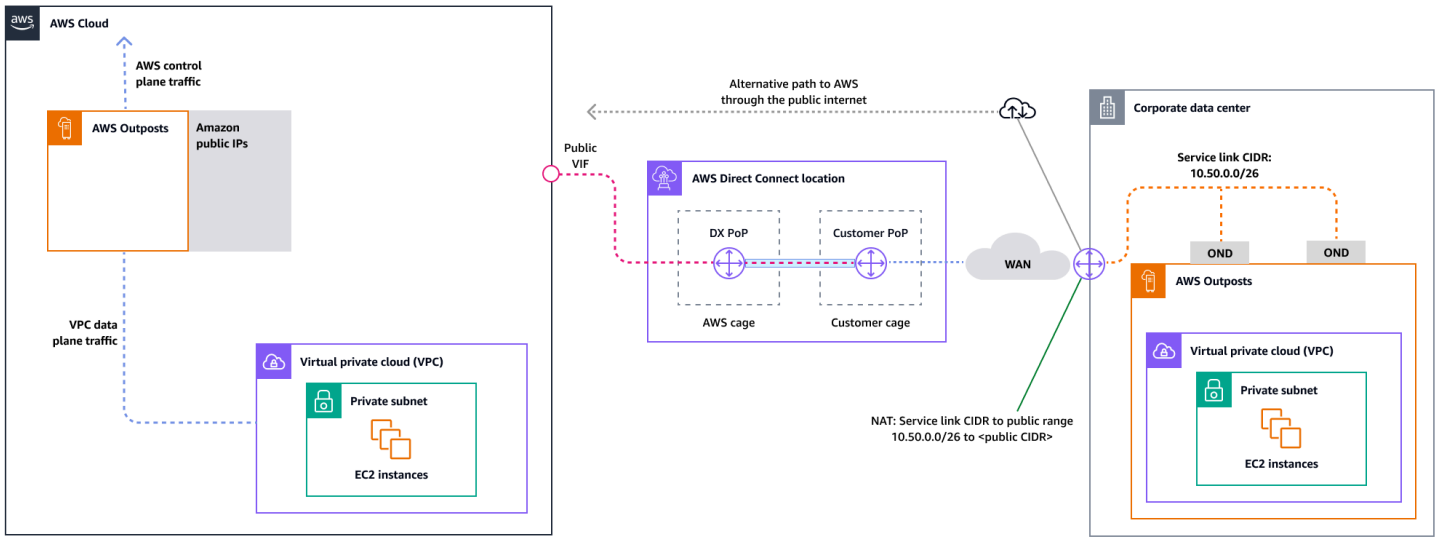
For more information about this connectivity, see Local network connectivity for Outposts Racks in the AWS Outposts documentation.

For an optimal experience and resiliency, AWSrecommends that you use redundant connectivity of at least 500 Mbps (1 Gbps is better) for the service link connection to the AWS Region. You can use Direct Connect or an internet connection for the service link. This minimum enables you to launch EC2 instances, attach EBS volumes, and access AWS services, such as Amazon EKS, Amazon EMR, and CloudWatch metrics.

The following diagram illustrates this architecture for a highly available private connection.



The following diagram illustrates this architecture for a highly available public connection.



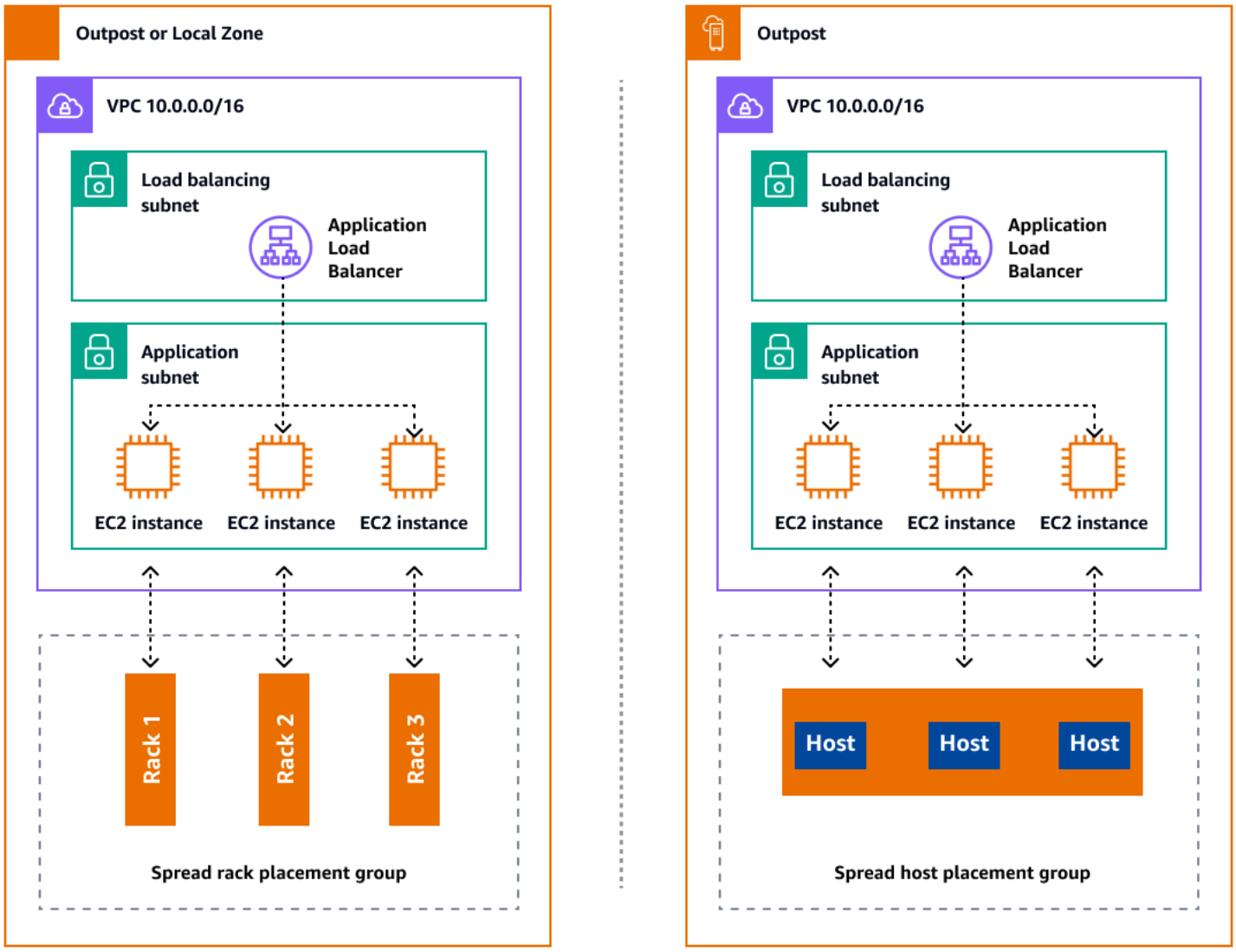
Scaling Outposts rack deployments with ACE racks

The Aggregation, Core, Edge (ACE) rack serves as a critical aggregation point for AWS Outposts multi-rack deployments, and is primarily recommended for installations that exceed three racks or for planning future expansion. Each ACE rack features four routers that support 10 Gbps, 40 Gbps, and 100 Gbps connections (100 Gbps is optimal). Each rack can connect to up to four upstream customer devices for maximum redundancy. ACE racks consume up to 10 kVA of power and weigh up to 705 lbs. Key benefits include reduced physical networking requirements, fewer fiber cabling uplinks, and decreased VLAN virtual interfaces. AWS monitors these racks through telemetry data via VPN tunnels and works closely with customers during installation to ensure proper power availability, network configuration, and optimal placement. The ACE rack architecture provides

increasing value as deployments scale, and effectively simplifies connectivity while reducing complexity and physical port requirements in larger installations. For more information, see the AWS blog post Scaling AWS Outposts rack deployments with ACE Rack.

Distributing instances across Outposts and Local Zones

Outposts and Local Zones have a finite number of compute servers. If your application deploys multiple related instances, these instances might deploy on the same server or on servers in the same rack unless they are configured differently. In addition to the default options, you can distribute instances across servers to mitigate the risk of running related instances on the same infrastructure. You can also distribute instances across multiple racks by using partition placement groups. This is called the *spread rack* distribution model. Use automatic distribution to spread instances across partitions in the group, or deploy instances to selected target partitions. By deploying instances to target partitions, you can deploy selected resources to the same rack while distributing other resources across racks. Outposts also provides another option called *spread host* that lets you distribute your workload at the host level. The following diagram shows the spread rack and spread host distribution options.



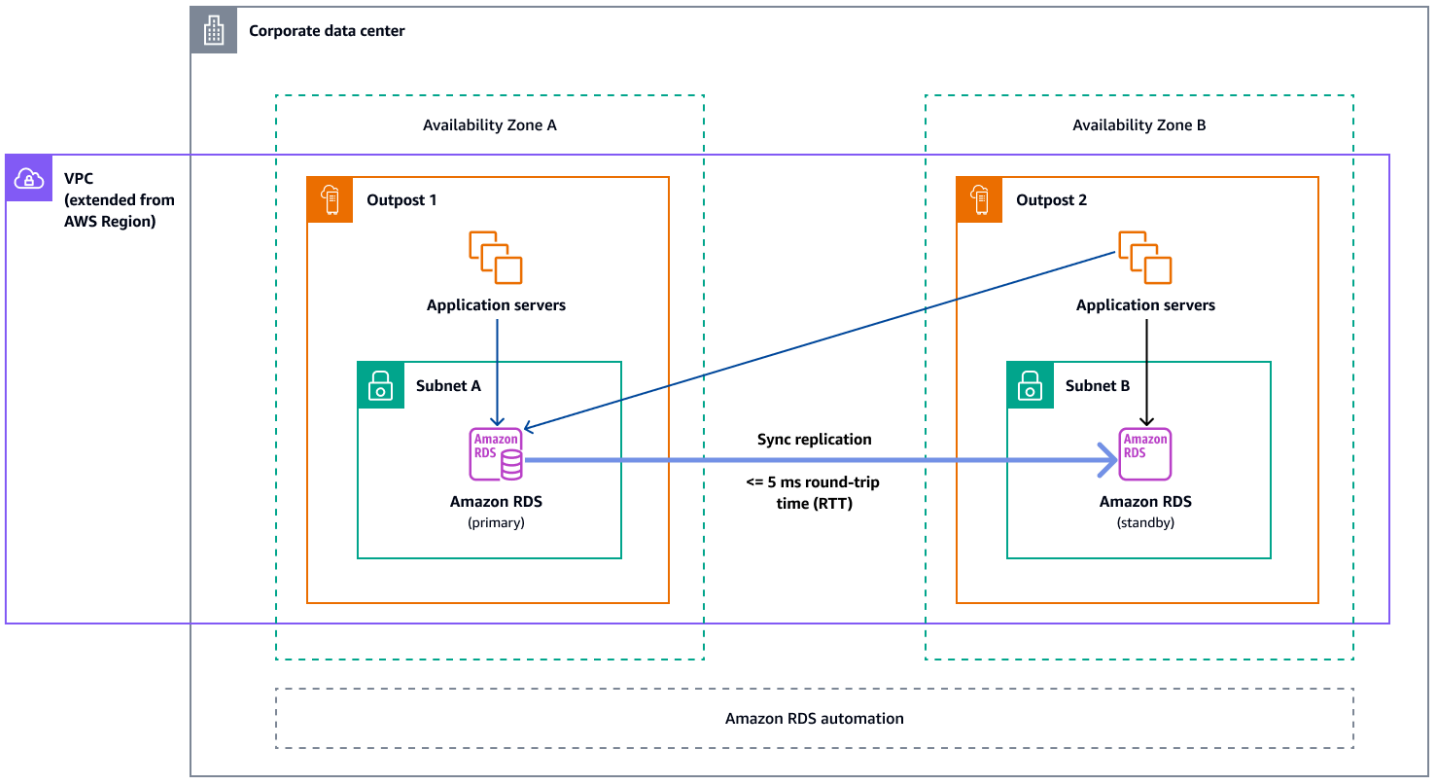
Amazon RDS Multi-AZ in AWS Outposts

When you use Multi-AZ instance deployments on Outposts, Amazon RDS creates two database instances across two Outposts. Each Outpost runs on its own physical infrastructure and connects to different Availability Zones in a Region for high availability. When two Outposts are connected through a customer-managed local connection, Amazon RDS manages synchronous replication between the primary and standby database instances. In case of a software or infrastructure failure, Amazon RDS automatically promotes the standby instance to the primary role and updates the DNS record to point to the new primary instance. For Multi-AZ deployments, Amazon RDS creates a primary DB instance on one Outpost and synchronously replicates the data to a standby DB instance on a different Outpost. Multi-AZ deployments on Outposts operate like Multi-AZ deployments in AWS Regions, with the following differences:

- They require a local connection between two or more Outposts.
- They require customer-owned IP (CoIP) address pools. For more information, see Customer-owned IP addresses for Amazon RDS on AWS Outposts in the Amazon RDS documentation.
- Replication runs on your local network.

Multi-AZ deployments are available for all supported versions of MySQL and PostgreSQL on Amazon RDS on Outposts. Local backups are not supported for Multi-AZ deployments.

The following diagram shows the architecture for Amazon RDS on Outposts Multi-AZ configurations.



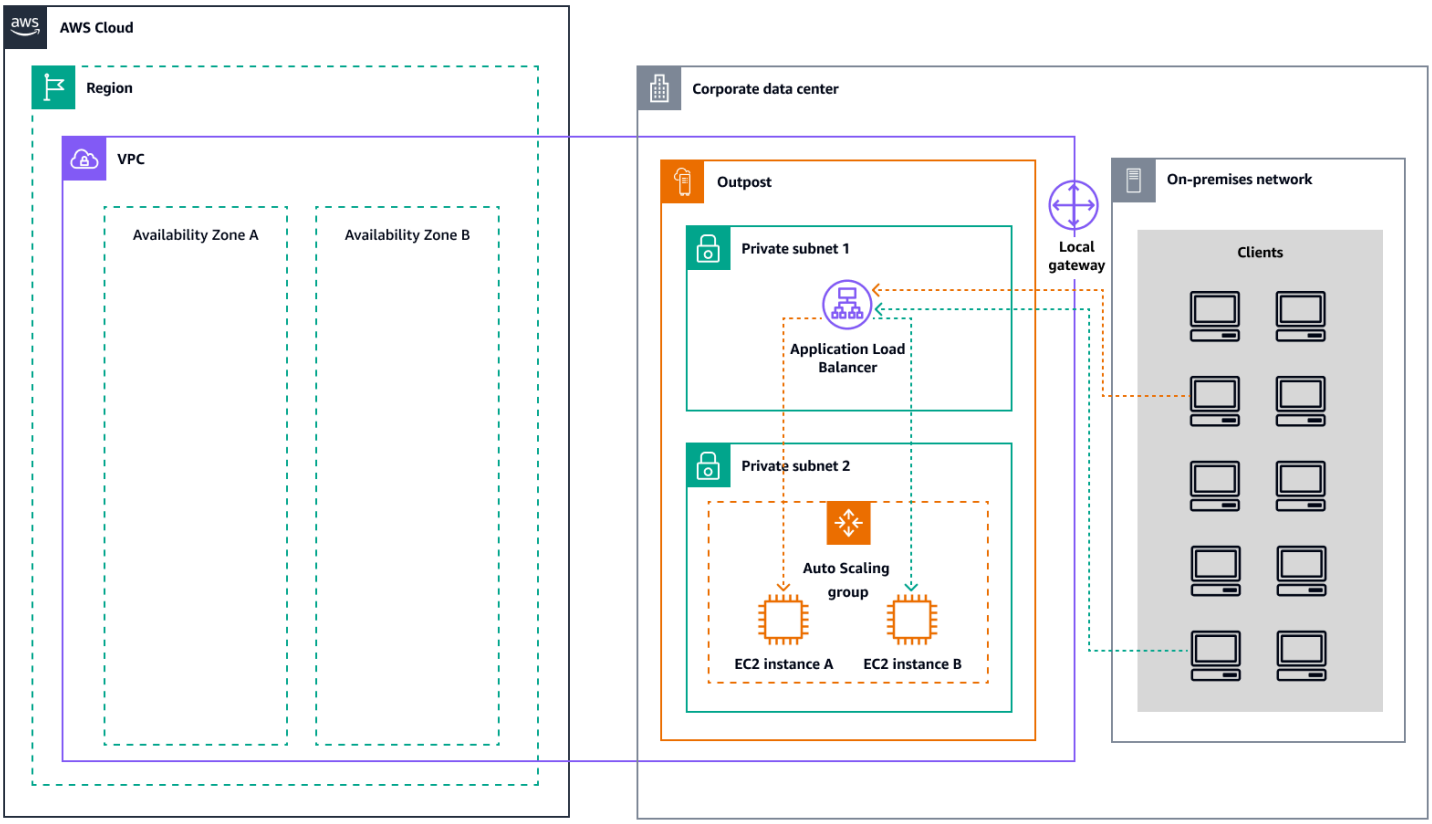
Failover mechanisms

Load balancing and automatic scaling

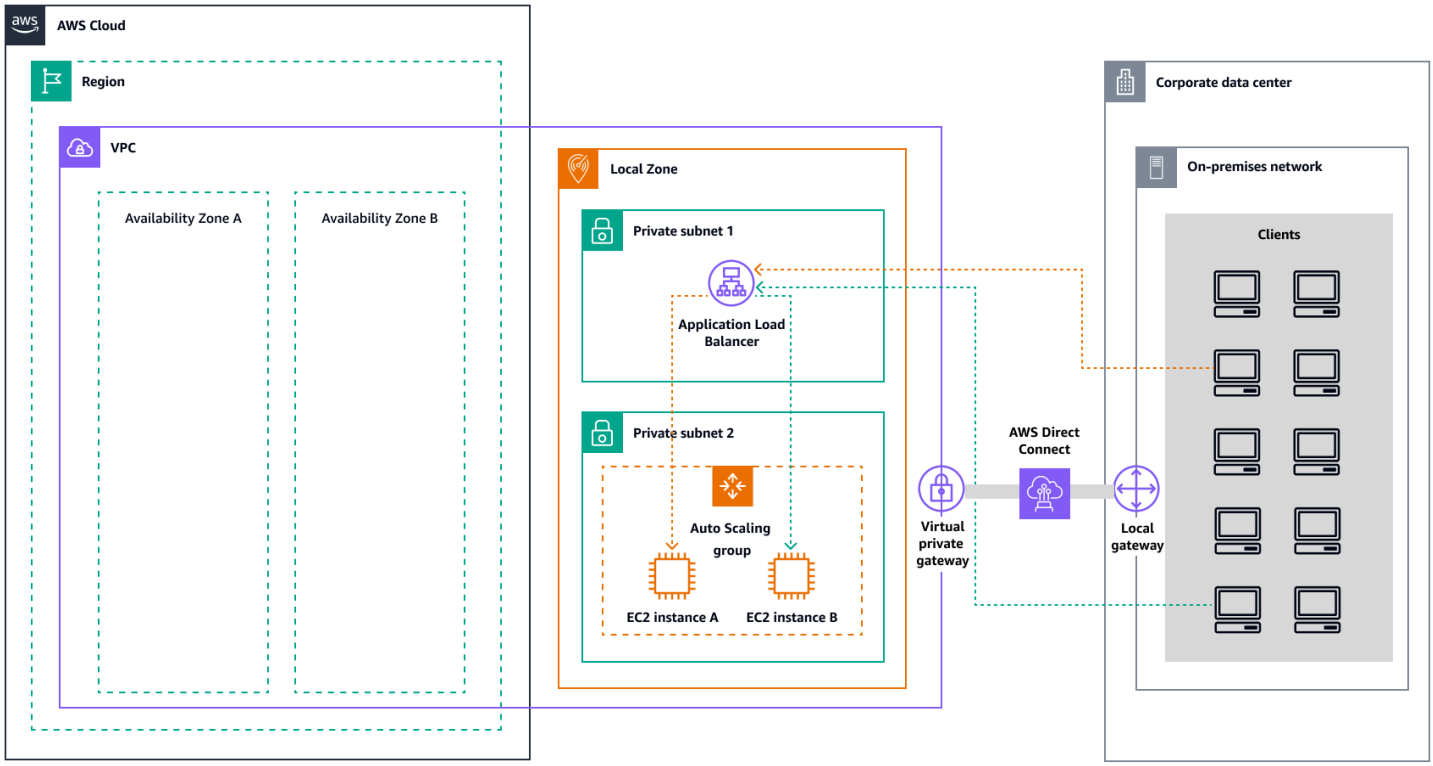
Elastic Load Balancing (ELB) automatically distributes your incoming application traffic across all the EC2 instances that you are running. ELB helps manage incoming requests by optimally routing traffic so that no single instance is overwhelmed. To use ELB with your Amazon EC2 Auto Scaling group, attach the load balancer to your Auto Scaling group. This registers the group with the load

balancer, which acts as a single point of contact for all incoming web traffic to your group. When you use ELB with your Auto Scaling group, it is not necessary to register individual EC2 instances with the load balancer. Instances that are launched by your Auto Scaling group are automatically registered with the load balancer. Similarly, instances that are terminated by your Auto Scaling group are automatically deregistered from the load balancer. After you attach a load balancer to your Auto Scaling group, you can configure your group to use ELB metrics (such as the Application Load Balancer request count per target) to scale the number of instances in the group as demand fluctuates. Optionally, you can add ELB health checks to your Auto Scaling group so that Amazon EC2 Auto Scaling can identify and replace unhealthy instances based on these health checks. You can also create an Amazon CloudWatch alarm that notifies you if the healthy host count of the target group is lower than allowed.

The following diagram illustrates how an Application Load Balancer manages workloads on Amazon EC2 in AWS Outposts.



The following diagram illustrates a similar architecture for Amazon EC2 in Local Zones.



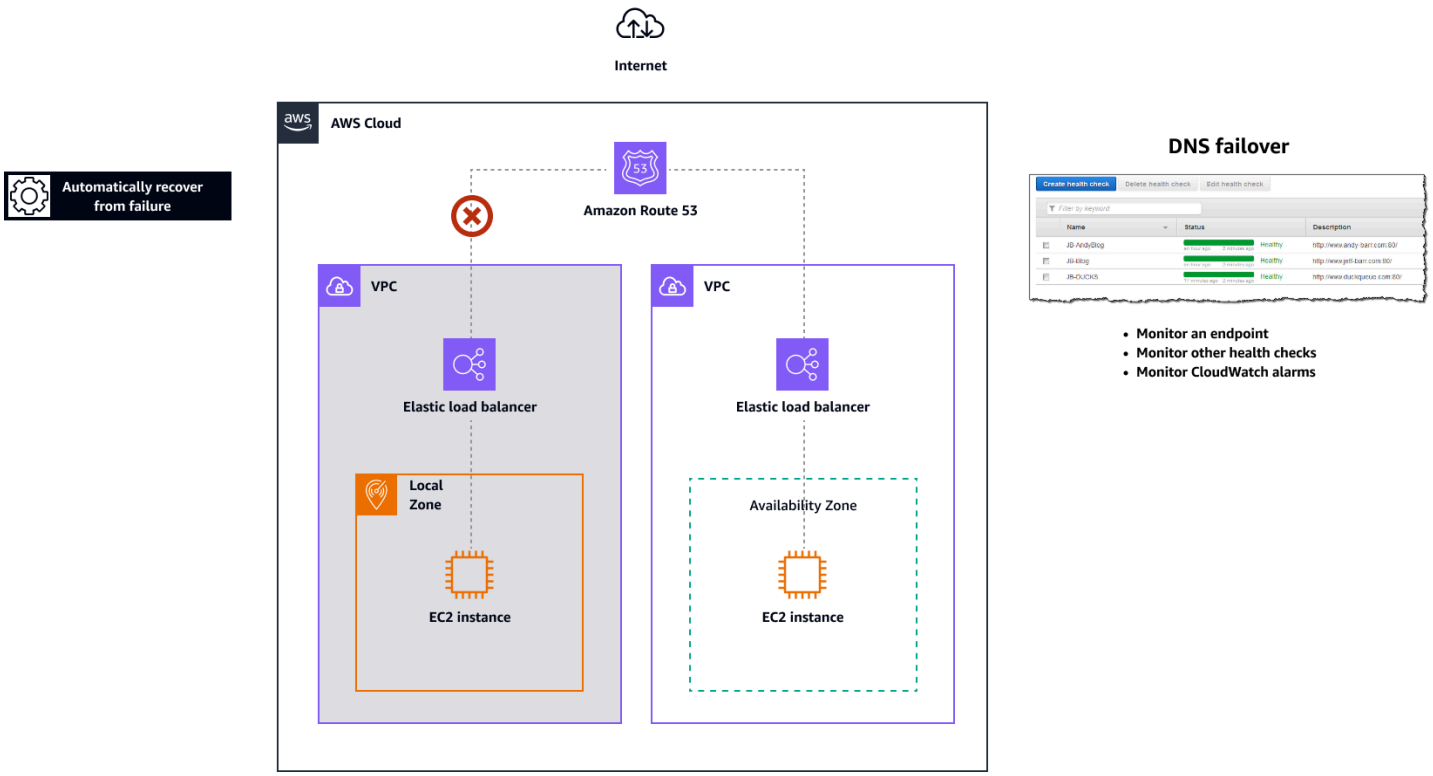
> **Note**
>
> Application Load Balancers are available in both AWS Outposts and Local Zones. However, to use an Application Load Balancer in AWS Outposts, you need to size the Amazon EC2 capacity to provide the scalability that the load balancer requires. For more information about sizing a load balancer in AWS Outposts, see the AWS blog post Configuring an Application Load Balancer on AWS Outposts.

Amazon Route 53 for DNS failover

When you have more than one resource performing the same function—for example, multiple HTTP or mail servers—you can configure Amazon Route 53 to check the health of your resources and respond to DNS queries by using only the healthy resources. For example, let's assume that your website, `example.com`, is hosted on two servers. One server is in a Local Zone and the other server is in an Outpost. You can configure Route 53 to check the health of those servers and to respond to DNS queries for `example.com` by using only the servers that are currently healthy. If you're using alias records to route traffic to selected AWS resources, such as ELB load balancers, you can configure Route 53 to evaluate the health of the resource and route traffic only to resources

that are healthy. When you configure an alias record to evaluate the health of a resource, you don't need to create a health check for that resource.

The following diagram illustrates Route 53 failover mechanisms.



Note

- If you're creating failover records in a private hosted zone, you can create a CloudWatch metric, associate an alarm with the metric, and then create a health check that is based on the data stream for the alarm.
- To make an application publicly accessible in AWS Outposts by using an Application Load Balancer, set up networking configurations that enable Destination Network Address Translation (DNAT) from public IPs to the load balancer's fully qualified domain name (FQDN), and create a Route 53 failover rule with health checks that point to the exposed public IP. This combination ensures reliable public access to your Outposts-hosted application.

Amazon Route 53 Resolver on AWS Outposts

Amazon Route 53 Resolver is available on Outposts racks. It provides your on-premises services and applications with local DNS resolution directly from Outposts. Local Route 53 Resolver endpoints also enable DNS resolution between Outposts and your on-premises DNS server. Route 53 Resolver on Outposts helps improve the availability and performance of your on-premises applications.

One of the typical use cases for Outposts is to deploy applications that require low-latency access to on-premises systems, such as factory equipment, high-frequency trading applications, and medical diagnosis systems.

When you opt in to use local Route 53 Resolvers on Outposts, applications and services will continue to benefit from local DNS resolution to discover other services, even if connectivity to a parent AWS Region is lost. Local Resolvers also help reduce latency for DNS resolutions because query results are cached and served locally from the Outposts, which eliminates unnecessary round-trips to the parent AWS Region. All DNS resolutions for applications in Outposts VPCs that use private DNS are served locally.

In addition to enabling local Resolvers, this launch also enables local Resolver endpoints. Route 53 Resolver outbound endpoints enable Route 53 Resolvers to forward DNS queries to DNS resolvers that you manage—for example, on your on-premises network. In contrast, Route 53 Resolver inbound endpoints forward the DNS queries they receive from outside the VPC to the Resolver that's running on Outposts. It allows you to send DNS queries for services deployed on a private Outposts VPC from outside that VPC. For more information about inbound and outbound endpoints, see Resolving DNS queries between VPCs and your network in the Route 53 documentation.

Capacity planning at the edge

The capacity planning phase involves collecting the vCPU, memory, and storage requirements to deploy your architecture. In the cost optimization pillar of the AWS Well-Architected Framework, right-sizing is an ongoing process that starts with planning. You can use AWS tools to define optimizations based on resource consumption within AWS.

Edge capacity planning in Local Zones is the same as in AWS Regions. You should check to make sure that your instances are available in each Local Zone, because some instance types might differ from the types in AWS Regions. For Outposts, you should plan for capacity based on your workload requirements. Outposts are slotted with fixed numbers of instances per host and can be reslotted

as needed. If your workloads require spare capacity, take that into consideration when you plan your capacity needs.

Capacity planning on Outposts

AWS Outposts capacity planning requires specific inputs for Regional right-sizing, plus edge-specific factors that affect application availability, performance, and growth. For detailed guidance, see Capacity planning in the AWS whitepaper *AWS Outposts High Availability Design and Architecture Considerations*.

Capacity planning for Local Zones

A Local Zone is an extension of an AWS Region that is geographically close to your users. Resources that are created in a Local Zone can serve local users with very low-latency communications. To enable a Local Zone in your AWS account, review Getting started with AWS Local Zones in the AWS documentation. Each Local Zone has different slotting available for families of EC2 instances. Validate the instances available in each Local Zone before you use them. To confirm the available EC2 instances, run the following AWS CLI command:

```
aws ec2 describe-instance-type-offerings \
--location-type "availability-zone" \
--filters Name=location,Values=<local-zone-name>
```

Expected output:

```
{
  "InstanceTypeOfferings": [
      {
          "InstanceType": "m5.2xlarge",
          "LocationType": "availability-zone",
          "Location": "<local-zone-name>"
      },
      {
          "InstanceType": "t3.micro",
          "LocationType": "availability-zone",
          "Location": "local.zone-name"
      },
      ...
  ]
}
```

Edge infrastructure management

AWS provides fully managed services that extend AWS infrastructure, services, APIs, and tools closer to your end users and data centers. The services that are available in Outposts and Local Zones are the same as those available in AWS Regions, so you can manage those services by using the same AWS console, AWS CLI, or AWS APIs. For supported services, see the AWS Outposts feature comparison table and AWS Local Zones features.

Deploying services at the edge

You can configure the available services in Local Zones and Outposts in the same way you configure them in AWS Regions: by using the AWS console, AWS CLI, or AWS APIs. The primary difference between Regional and edge deployments is the subnets where resources will be provisioned. The Networking at the edge section described how subnets are deployed in Outposts and Local Zones. After you identify the edge subnets, you use the edge subnet ID as a parameter to deploy the service in Outposts or Local Zones. The following sections provide examples of deploying edge services.

Amazon EC2 at the edge

The following `run-instances` example launches a single instance of type `m5.2xlarge` into the edge subnet for the current Region. The key pair is optional if you do not plan to connect to your instance by using SSH on Linux or remote desktop protocol (RDP) on Windows.

```
aws ec2 run-instances \
    --image-id ami-id \
    --instance-type m5.2xlarge \
    --subnet-id <subnet-edge-id> \
    --key-name MyKeyPair
```

Application Load Balancers at the edge

The following `create-load-balancer` example creates an internal Application Load Balancer and enables the Local Zones or Outposts for the specified subnets.

```
aws elbv2 create-load-balancer \
    --name my-internal-load-balancer \
    --scheme internal \
```

```
    --subnets <subnet-edge-id>
```

To deploy an internet-facing Application Load Balancer to a subnet on an Outpost, you set the `internet-facing` flag in the `--scheme` option and provide a CoIP pool ID, as shown in this example:

```
aws elbv2 create-load-balancer \
    --name my-internal-load-balancer \
    --scheme internet-facing \
    --customer-owned-ipv4-pool <coip-pool-id>
    --subnets <subnet-edge-id>
```

For information about deploying other services at the edge, follow these links:

Service	AWS Outposts	AWS Local Zones
Amazon EKS	Deploy Amazon EKS on-premises with AWS Outposts	Launch low-latency EKS clusters with AWS Local Zones
Amazon ECS	Amazon ECS on AWS Outposts	Amazon ECS applications in shared subnets, Local Zones, and Wavelength Zones
Amazon RDS	Amazon RDS on AWS Outposts	Select the Local Zone subnet
Amazon S3	Getting started with Amazon S3 on Outposts	Not available
Amazon ElastiCache	Using Outposts with ElastiCache	Using Local Zones with ElastiCache
Amazon EMR	EMR clusters on AWS Outposts	EMR clusters on AWS Local Zones
Amazon FSx	Not available	Select the Local Zone subnet
AWS Elastic Disaster Recovery	Working with AWS DRS and AWS Outposts	Not available

AWS Transform MGN	Not available	Select the Local Zone subnet as the staging subnet

Outposts-specific CLI and SDK

AWS Outposts has two groups of commands and APIs for creating a service order or manipulating the routing tables between the local gateway and your local network.

Outposts ordering process

You can use the AWS CLI or the Outposts APIs to create an Outposts site, to create an Outpost, and to create an Outposts order. We recommend that you work with a hybrid cloud specialist during your AWS Outposts ordering process to ensure proper selection of resource IDs and optimal configuration for your implementation needs. For a complete resource ID list, see the AWS Outposts racks pricing page.

Local gateway management

The management and operation of the local gateway (LGW) in Outposts requires knowledge of the AWS CLI and SDK commands available for this task. You can use the AWS CLI and AWS SDKs to create and modify LGW routes, among other tasks. For more information about managing the LGW, see these resources:

- AWS CLI for Amazon EC2
- EC2.Client in the AWS SDK for Python (Boto)
- Ec2Client in the AWS SDK for Java

CloudWatch metrics and logs

For AWS services that are available in both Outposts and Local Zones, metrics and logs are managed in the same way as in Regions. Amazon CloudWatch provides metrics that are dedicated to monitoring Outposts in the following dimensions:

Dimension	Description
`Account`	The account or service using the capacity

`InstanceFamily`	The instance family
`InstanceType`	The instance type
`OutpostId`	The ID of the Outpost
`VolumeType`	The EBS volume type
`VirtualInterfaceId`	The ID of the local gateway or service link virtual interface (VIF)
`VirtualInterfaceGroupId`	The ID of the VIF group for the local gateway VIF

For more information, see CloudWatch metrics for Outposts racks in the Outposts documentation.

Resources

AWS references

- Hybrid Cloud with AWS
- AWS Outposts User Guide for Outposts racks
- AWS Local Zones User Guide
- AWS Outposts Family
- AWS Local Zones
- Extend a VPC to a Local Zone, Wavelength Zone, or Outpost (Amazon VPC documentation)
- Linux instances in Local Zones (Amazon EC2 documentation)
- Linux instances in Outposts (Amazon EC2 documentation)
- Get Started Deploying Low Latency Applications with AWS Local Zones (tutorial)

AWS blog posts

- Running AWS infrastructure on premises with Amazon EC2
- Building modern applications with Amazon EKS on Amazon EC2
- How to choose between CoIP and direct VPC routing modes on Amazon EC2 rack
- Selecting network switches for your Amazon EC2
- Maintaining a local copy of your data in AWS Local Zones
- Amazon ECS on Amazon EC2
- Managing edge-aware service mesh with Amazon EKS for AWS Local Zones
- Deploying local gateway ingress routing on Amazon EC2
- Automating your workload deployments in AWS Local Zones
- Sharing Amazon EC2 in a multi account AWS environment: Part 1
- Sharing Amazon EC2 in a multi account AWS environment: Part 2
- AWS Direct Connect and AWS Local Zones interoperability patterns
- Deploy Amazon RDS on Amazon EC2 with Multi-AZ high availability

Contributors

The following individuals contributed to this guide.

Authoring

- Leonardo Solano, Principal Hybrid Cloud Solutions Architect, AWS
- Len Gomes, Partner Solutions Architect, AWS
- Matt Price, Senior Enterprise Support Engineer, AWS
- Tom Gadomski, Solutions Architect, AWS
- Obed Gutierrez, Solutions Architect, AWS
- Dionysios Kakaletris, Technical Account Manager, AWS
- Vamsi Krishna, Principal Outposts Specialist, AWS

Reviewing

- David Filiatrault, Delivery Consultant, AWS

Technical writing

- Handan Selamoglu, Sr. Documentation Manager, AWS

Document history

The following table describes significant changes to this guide.

Change	Description	Date
Initial publication	—	June 10, 2025