DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

Assignment 4

In
The Class of

CSCI 5411: ADVANCED CLOUD ARCHITECTING

by

Shreya Kapoor (B00957587)

Table of Contents

Ques 1. To handle outbound internet connectivity from the private subnets, I propose deploying a NAT Gateway in every subnet in all VPCs globally

Deploying a NAT Gateway in every subnet across all Virtual Private Clouds (VPCs) globally for managing outbound internet connectivity from private subnets presents several challenges and considerations. While NAT Gateways are essential for facilitating connectivity, their universal deployment can lead to increased costs and operational complexities. This analysis examines the drawbacks of this approach and proposes a more efficient solution.

1. **Issue with Universal Deployment of NAT Gateways:** Deploying a NAT Gateway in every subnet across all VPCs globally to manage outbound internet connectivity from private subnets may not be the most efficient choice. While NAT Gateways facilitate connectivity, this approach can lead to higher costs and operational complexities.

2. **Cost and Efficiency Considerations:** Implementing a NAT Gateway in every subnet would significantly increase expenses due to associated fees. It's more cost-effective to deploy NAT Gateways selectively in subnets that specifically need outbound internet access. This targeted deployment reduces costs and simplifies tasks like setup and troubleshooting.

3. **Complexity and Scalability Issues:** Having multiple NAT Gateways in various subnets complicates network management and could potentially impact performance by creating bottlenecks. Additionally, each VPC has limits on the number of NAT Gateways, restricting scalability.

4. **Recommended Approach - Centralized Solution:** A better approach is to adopt a centralized NAT Gateway solution. This means deploying NAT Gateways only where necessary, optimizing costs, simplifying management tasks, and ensuring scalability. Consideration of broader network strategies like VPC peering or transit gateways can further enhance secure and efficient communication between Cupid's on-premises systems and AWS Cloud across different regions.

By choosing a centralized NAT Gateway approach and carefully planning the network architecture, Cupid can effectively address challenges related to outbound internet connectivity. This approach maintains cost-efficiency, simplifies operations, and supports scalability for future growth [1].

Ques 2. Given that we have multiple AWS regions in use, I suggest creating a fully meshed VPC peering setup for all VPCs in each region.

Proposing a fully meshed VPC peering setup for all VPCs across multiple AWS regions might seem like a good idea, but it can introduce significant challenges related to complexity, scalability, and performance.

1. **Complexity Issues:** Managing a fully meshed VPC peering configuration becomes very complicated as the number of VPCs and regions grows. Each VPC would need

to establish peering connections with every other VPC, creating a significant administrative burden. This can lead to misconfigurations and difficulties in troubleshooting.

2. **Scalability Challenges:** In a fully meshed setup, the number of peering connections increases exponentially, making it hard to maintain and scale the network effectively. Adding new VPCs or regions would become challenging and involve a lot of configuration and management work.

3. **Performance Impact:** A fully meshed VPC peering setup can negatively affect performance. Traffic between VPCs might take longer, less efficient paths, leading to higher latency and potential performance bottlenecks. Direct communication between VPCs may not follow the most optimal route, causing delays in data transfer [2].

4. **Recommended Approach - Hub-and-Spoke Topology:** A better approach is to use a hub-and-spoke network topology. This involves creating a central "hub" VPC in each region and connecting the "spoke" VPCs to this hub. This reduces the number of peering connections to configure and maintain, simplifying management.

5. **Advantages of Hub-and-Spoke Model:**

   - **Scalability:** New VPCs or regions can be easily added by connecting them to the hub VPC, making the network more scalable.
   - **Control and Security:** The hub VPC serves as a central point for implementing network access control and security measures.
   - **Performance Considerations:** While this model might introduce some extra latency because traffic flows through the hub VPC, this can be minimized by using appropriately sized and well-connected instances for the hub.

Using a hub-and-spoke topology is a more efficient and manageable approach compared to a fully meshed VPC peering setup. It simplifies network management, enhances scalability, and maintains better control and security while minimizing performance issues [3].

Ques 3. To ensure the security of our sensitive user data, I propose encrypting all traffic in-transit within our VPCs using IPsec tunnels.

Encrypting all traffic in-transit within VPCs using IPsec tunnels might seem like a good idea for securing sensitive user data. However, considering the strong security features already provided by AWS, this approach may not be the best choice.

1. **AWS Security Features:** AWS offers a range of security measures to protect data in-transit within its VPC infrastructure. These include secure connectivity options like VPN connections, AWS PrivateLink for private communication [4], and TLS termination in services like ELB and CloudFront.

2. **Leveraging Native Encryption Capabilities:** By using these built-in encryption features, Cupid can effectively secure data in-transit without the added complexity and overhead of implementing IPsec tunnels for all traffic within VPCs. AWS's existing encryption mechanisms are designed to meet high security standards and

can protect sensitive user data efficiently.

3. **Complexity and Performance Concerns:** Introducing IPsec tunnels for encrypting all traffic within VPCs could lead to unnecessary complexity and potential performance issues. Managing IPsec tunnels for all traffic would add administrative overhead and could slow down the network [5].

4. **Recommended Approach - Using AWS Encryption Capabilities:** Instead of using IPsec tunnels, Cupid should focus on utilizing AWS's encryption capabilities. This includes:

    - **VPN Connections:** For secure connectivity between on-premises infrastructure and VPCs.
    - **SSL/TLS Encryption:** For protecting sensitive data transmitted over the internet [6].
    - **Security Groups and NACLs:** For controlling access and securing the network.

5. **Benefits of Aligning with AWS Best Practices:** By following AWS's best practices and using their robust security features, Cupid can create a secure environment within their VPCs. This approach ensures the protection of sensitive user data during transit and helps comply with data privacy regulations.

Rather than implementing IPsec tunnels for all in-transit traffic, Cupid should leverage AWS's native encryption capabilities. This approach simplifies management, avoids potential performance issues, and ensures robust data protection in line with AWS's high security standards.

Ques 4. I propose using the same NACLs for all our subnets to maintain a consistent security posture.

Using the same Network Access Control Lists (NACLs) for all subnets to maintain a consistent security posture might seem like a good idea. However, this approach may not be the most suitable for Cupid's architecture. Following are the reasons:

1. **Detailed Security Management:** Different subnets within Cupid's architecture have unique security needs and risk levels. Applying the same NACLs to all subnets could either expose sensitive resources due to overly permissive rules or block necessary communications due to overly restrictive rules. It's important to evaluate and tailor NACL rules for each subnet based on its specific requirements.

2. **Adhering to Least Privilege:** Following the principle of least privilege is essential for security. Each subnet should have its own NACLs that only grant necessary permissions and restrict access to sensitive resources. This minimizes the potential attack surface and reduces the impact of any security breaches.

3. **Growth and Maintenance:** As Cupid's architecture grows and changes, using the same NACLs for all subnets can create management challenges. With more subnets and increased network complexity, maintaining and updating security policies consistently across all subnets becomes difficult. Tailored NACLs help manage security more effectively as the network evolves.

Impact on Public and Private Subnets: NACLs play a crucial role in controlling traffic flow to and from subnets. Public subnets, which interact with the internet, need NACLs that can handle a wide range of IP addresses while maintaining security against attacks. On the other hand, private subnets, which are isolated from direct internet access, require stricter NACLs to ensure that only authorized internal traffic is allowed. Using the same NACLs for both public and private subnets could lead to security vulnerabilities or operational issues.

Recommended Approach: Instead of using identical NACLs for all subnets, a better strategy is to assess the security needs of each subnet individually and create specific NACLs tailored to those needs. This allows for fine-grained control over traffic flows, ensuring appropriate security levels while enabling necessary communication between components.

While maintaining consistent security measures is important, it’s crucial to balance this with the unique security requirements of individual subnets within Cupid’s architecture. By tailoring NACLs to the specific needs of each subnet, Cupid can achieve a more secure and manageable network environment.

Ques 5. To enable secure access to user profile from the VPCs, I propose using public APIs which is very convenient.

Using public APIs to access user profiles from VPCs might seem convenient, but it raises significant security and privacy concerns. Public APIs are designed for external access and are exposed to the public internet, which can pose risks to sensitive user data.

1. **Security Risks of Public APIs:** Public APIs bypass internal security controls like firewalls and network access control lists (NACLs) that protect VPCs. This weakens the security posture and could expose sensitive user profile data to unauthorized access or malicious attacks. Public APIs, being exposed to the internet, are more vulnerable to threats such as data breaches, DDoS attacks, and other cyber threats that could compromise the confidentiality and integrity of user data.

2. **Internal APIs as a Secure Alternative:** A safer approach is to use internal APIs specifically designed for accessing user profile data within the VPCs. These internal APIs can be secured with proper authentication methods like API keys or access tokens. Additionally, VPC-level security measures like NACLs and security groups can control access to these APIs, ensuring that only authorized entities within the VPCs can interact with sensitive user data. Internal APIs are not exposed to the public internet, which significantly reduces the attack surface and potential vulnerabilities.

3. **Enhanced Security and Control:** Using internal APIs allows Cupid to maintain secure and controlled access to sensitive user profile data within the VPCs. This ensures that access to the data is limited to authorized entities within the VPCs, providing an additional layer of security. Internal APIs also allow for more granular control over access permissions, enabling the implementation of role-based access control (RBAC) and other security best practices. Furthermore, internal APIs enable the application of encryption measures both in transit and at rest, ensuring the highest level of data protection.

Recommended Approach: Instead of relying on public APIs, Cupid should implement internal APIs for accessing user profile data within the VPCs. This approach aligns with best security practices and provides an additional layer of protection for sensitive information. Internal APIs should be designed with strong authentication and authorization mechanisms, encryption, and logging to monitor and audit access to sensitive data.

Given the sensitivity of user profile data, it's crucial to prioritize security. By adopting internal APIs and robust security measures within the VPCs, Cupid can enhance the security of user profile data and mitigate risks associated with using public APIs. This approach ensures secure and compliant access to sensitive information, protecting it from potential security breaches and aligning with regulatory requirements [7].

Ques 6. I wanted to use transit gateway to centrally route the traffic. However, since transit gateway cannot be connected to the on-premise network, I will have a mix of VPC peering and Direct Connect or VPN connection to the on-premise network.

Due to the limitations of Transit Gateway in connecting directly to on-premise networks, using a mix of VPC peering, Direct Connect, and VPN connections is a suitable and effective solution. Here's why this approach works well for Cupid's architecture.

1. **Role of Transit Gateway:** Transit Gateway is a scalable and flexible service that simplifies network connectivity and routing within AWS. However, it doesn't provide a direct connection to on-premise networks. Therefore, relying solely on Transit Gateway isn't feasible for establishing a hybrid network setup [8].

2. **Benefits of VPC Peering:** VPC peering allows for secure and private communication between VPCs within AWS regions. It enables the exchange of traffic between VPCs using private IP addresses, facilitating efficient and controlled data transfer. By leveraging VPC peering, Cupid can ensure secure communication between different VPCs in the AWS cloud [2].

3. **Connecting to On-Premise Network:** To connect AWS infrastructure with the on-premise network, using Direct Connect or VPN connections is appropriate:

    - **Direct Connect:** Provides a dedicated and private network connection between AWS and the on-premise infrastructure, ensuring reliable and secure communication. This option is ideal for high-throughput and low-latency requirements [9].
    - **VPN Connections:** Utilize encrypted tunnels over the public internet to establish a secure connection between AWS VPCs and the on-premise network. VPN connections are flexible and can be set up quickly, providing a secure way to connect distributed resources.

Combining these connectivity options allows Cupid to establish a hybrid communication setup that ensures secure and efficient data transfer between the on-premise system and the AWS Cloud across different regions. By using a mix of VPC peering, Direct Connect, and VPN connections, Cupid can achieve a robust and scalable network architecture, balancing security, performance, and flexibility.

References

[1] “Centralized egress to internet,” *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/whitepapers/latest/building-scalable-secure-multi-vpc-network-infrastructure/centralized-egress-to-internet.html. [Accessed: July 15, 2024].

[2] “VPC peering configurations with routes to an entire VPC,” *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/peering/peering-configurations-full-access.html. [Accessed: July 15, 2024].

[3] “Hub-and-spoke topology vs mesh topology,” *Ccna-classes.com*. [Online]. Available: https://ccna-classes.com/ccna-study-resources/hub-and-spoke-topology-vs-mesh-topology/. [Accessed: July 15, 2024].

[4] “AWS PrivateLink,” *Amazon.com*. [Online]. Available: https://aws.amazon.com/privatelink/. [Accessed: July 15, 2024].

[5] “What is IPsec? | How IPsec VPNs work,” *Cloudflare.com*. [Online]. Available: https://www.cloudflare.com/learning/network-layer/what-is-ipsec/. [Accessed: July 15, 2024].

[6] “What is an SSL/TLS Certificate?,” *Amazon.com*. [Online]. Available: https://aws.amazon.com/what-is/ssl-certificate/#:~:text=SSL%2FTLS%20stands%20for%20secure,using%20the%20SSL%2FTLS%20protocol. [Accessed: July 15, 2024].

[7] “A complete guide to building and managing internal APIs,” Opslevel.com. [Online]. Available: https://www.opslevel.com/resources/a-complete-guide-to-building-and-managing-internal-apis. [Accessed: July 15, 2024].

[8] “What is a transit gateway?,” *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/tgw/what-is-transit-gateway.html#:~:text=A%20transit%20gateway%20is%20a,using%20the%20AWS%20Global%20Infrastructure. [Accessed: July 15, 2024].

[9] “AWS Direct Connect,” *Amazon.com*. [Online]. Available: https://aws.amazon.com/directconnect/. [Accessed: July 15, 2024].