

According to a 2023 Military Transition Survey conducted by the University of Phoenix, approximately 32% of non-active military members reported that they encountered obstacles transferring their military skills to the right civilian job.

If you or someone you know is a veteran interested in building with AWS, this article is for you.

You receive a tasking: conduct a reconnaissance operation. This is unlike any reconnaissance operation you have ever conducted, although the insights gained from this operation will undeniably shape strategic decisions. Similarly to how reconnaissance missions involve simply observing behavior, your new mission is to develop and optimize software that provides key insights on clickstream data in order to better position your best selling products, drive user engagement, and increase total sales.

The client that asked for this software has specific intelligence requirements. The client would like to learn as much as possible about user behavior patterns.

- Where are they coming from?
- What are they clicking on the most?
- How long are they staying on the site?

But here is the constraint that matters: the data acquired is incredibly valuable and as such it must remain secure. The client cannot expose their analytical infrastructure to the open internet.

This is where tactical doctrine translates brilliantly into cloud architecture.

Your first step is to establish a Virtual Private Cloud (VPC). This is your operational area of responsibility. This is the defined space where you control every ingress and egress point. Within it, you create your public subnet (your forward position where traffic initially lands) and your private subnet (your secure command post where the actual intelligence analysis happens).

The client's clickstream collection endpoint is fronted by an Application Programming Interface (API) Gateway endpoint at the public edge. It is still visible but it remains a hardened location designed to receive incoming data. The data analysis and storage, however, does not happen here. The public facing API receives, validates, and immediately passes the intelligence backward through a controlled channel intentionally and strategically directed by your route table and by the integration Lambda that runs inside your private subnet.

Security is an absolute must. As you layer those security cordons from the inside out, you establish security groups around the public subnet with very explicit instructions, "Allow HTTPS inbound on port 443 from anywhere. Deny everything else." Similarly, you position a security group around your private subnet but the instructions differ. These are more stringent given the sensitive nature of the intel analysis happening within. "Only accept traffic originating from the

public subnet's Elastic Network Interfaces (ENIs). Reject all traffic that does not meet this criteria."

Your Network Access Control Lists (ACLs) broaden your security posture. This is a stateless perimeter check that evaluates traffic before it even gets close to your security groups. Its stateless nature is its strength. It does not care that it already checked your credentials when you came in, it will check your credentials again on the way out. No if's, and's, or but's.

Together your route table, Network ACL, and Security Groups form a layered defense consisting of perimeter control, packet filtering, and stateful enforcement.

We finally get to your actual mission. The clickstream data arrives at your API Gateway, which triggers a Lambda function. That Lambda function validates the incoming request, assumes its IAM role, then forwards the authenticated data into an SQS queue. You access the SQS through a VPC endpoint, ensuring your clickstream data never traverses the public internet. The queue remains reachable only through the SQS VPC endpoint from your private subnet. Functionally, this keeps the entire data path within your controlled area of operations, never exposed to routing through public infrastructure.

Lambda functions, akin to troops tasked to run back and forth conducting a specific job, poll from that queue in controlled batches preserving the order of messages when using a FIFO queue. They process each click event, enrich it with geolocation data, session tracking, and detailed behavioral patterns. They take meticulous and organized notes. The longer unstructured notes go to your Data Lake. The detailed metadata gets neatly logged in DynamoDB.

Eventually the client needs to see this intelligence. Perhaps they access it through a different controlled channel like a dashboard running on an EC2 instance in the private subnet, accessible only through AWS Systems Manager Session Manager, which routes through the AWS control plane and not the internet. Or they pull reports through a secured API that sits behind CloudFront and WAF, which applies additional filtering before anything reaches your infrastructure.

Beneath all of this runs CloudTrail. Every API call, every Lambda invocation, every DynamoDB write, every IAM check that succeeded or failed gets logged with a timestamp and the identity of who actioned this. CloudTrail is your operational record. When you need answers, CloudTrail gives you the complete signal trace. If something goes sideways, you have the full record to understand exactly what happened and why. You have accountability.

What is fascinating is of course the technical capability that we've just described but also the fact that as a veteran you already understand this. You know that layered defenses beat single perimeters. You have executed asynchronous operations, and you've planned operations with redundancy built in because single points of failure are unacceptable.

The AWS services are just the terminology but the doctrine is already in your muscle memory.

I hope that this short essay helps at least one veteran realize that what they learned in the service is still applicable and in turn this provides a sense of purpose.

We lose too many veterans every day and now, more than ever, we need builders. We need you.