*Note: Talk track explicitly assumes that SE has shown “Day 2” deck and exposed pain points that justify move to automated certificate renewals. I lightly touch on those in the “Introduction” talk track but am intentionally not diving deeper so the SE isn’t saying the same things over and over (too much, anyway).*

*Last reviewed for module v0.1.0-rc.0* [*https://github.com/hashicorp/demo-vault-auto-renewed-certs/tree/0.1.0-rc.0*](https://github.com/hashicorp/demo-vault-auto-renewed-certs/tree/0.1.0-rc.0)

1. Introduction
   1. Talk Track:
      1. “I’m going to showcase how easy it is to automate certificate renewals with Vault Enterprise. As we discussed, automating this process reduces the risk of costly service outages and frees up thousands of hours of your teams’ time spent on manually rotating certificates, so they can focus on the things that differentiate your business.
      2. “In this demo, we’ll focus on a simple case: a group of EC2 virtual machines that all need their own certificate, with Vault Agent installed on each machine and handling certificate rotation.
      3. “While we won’t showcase most of them today, there are a lot of features in Vault Enterprise that are critical to operating a PKI in production and at scale, specifically around **governance** (namespaces, managed keys, control groups), **scalability** (performance standbys, performance replication), and **availability** (DR replication). I’m happy to walk you through them at the end or a separate occasion if you’d like to know more about specific features.”
2. Starting Point – No Certificates
   1. Talk Track: “Let’s start by taking a look at our demo application. We have five virtual machines running, but none of them have a certificate yet.”
   2. *Action: Navigate to public-facing Demo App URL (found in the module’s outputs)*
   3. *Action: Refresh a few times to show multiple EC2 instance IDs, none of them having a certificate*
3. Vault Authentication & Authorization
   1. Talk Track:
      1. “Let’s fix this. All these EC2 instances are authenticating to Vault using their native AWS identity and within a Vault namespace dedicated to the demo business unit.
      2. “Vault namespaces are an administrative construct that enables platform operators to delegate authentication, authorization, and secrets management decisions – effectively supporting the at-scale offering of Vault as an internal service.
      3. “For our demo, we have the *Demo BU* namespace set up.”
   2. *Action: Navigate to Demo BU namespace.*
   3. Talk Track:
      1. “This Demo BU namespace has its own set of Secrets Engines. In our case, it’s only the Demo BU Issuing CA, but normally you would have things like dynamic cloud and database credentials, encryption-as-a-service, etc.
      2. “The namespace also has its own set of trusted identity providers, entities, and groups.”
   4. *Action: Navigate to* ***Access*** *tab.*
   5. Talk Track: “In this case, we have set up AWS itself as a trusted identity provider, so that our EC2 instances can authenticate to Vault with their IAM Instance Profile. This solves the challenge of ‘secret zero’ – you get rid of those super-sensitive bootstrap secrets and instead rely on an assertion by the underlying platform that this machine has this identity.”
   6. *Action: Navigate to* ***Entities***
   7. Talk Track: “In this case, we already have a Vault entity (a security principal) configured. We called it ‘my-app-backend’ and mapped it to an AWS IAM Role that the EC2 instances all share.”
   8. *Action: Navigate to* ***Groups***
   9. Talk Track: “Let’s fix the permissions, so the EC2 instances can get their certificates. We have precreated a group ‘my-app-backend’ that has a policy that grants group members the permissions they need to issue certificates. Let’s add the ‘my-app-backend’ entity to the group.”
   10. *Action: Add entity to group.*
   11. Talk Track: “Let’s also take a quick look at the permissions policy. Vault permissions are default-deny, meaning you have to specifically grant specific permissions for any given action to be permissible.”
   12. *Action: Navigate to the group’s ACL policy.*
   13. Talk Track: “The group has permission to access these two API paths within the Demo BU’s Vault namespace. You’ll note that the specific path includes ‘my-app-backend’ – that’s referencing what we call a *PKI Role*, which is how you define requirements and restrictions like permissible hostnames, maximum certificate lifetimes, etc.”
4. Ta-Dah!
   1. Talk Track: “Let’s now look at our demo application. There’s sometimes a bit of a delay until the Vault Agent retries fetching those certificates, but hopefully enough time has passed.”
   2. *Action: Go to demo app URL, refresh a few times*
   3. Talk Track:
      1. “As you can see, each of our five EC2 instances has its own certificate. Each certificate is expiring every 30 seconds and is automatically renewed by Vault Agent running in the background.
      2. Note that this is for demo purposes only – you’d normally want your certificate periods to be a bit more reasonable, like two weeks or a month.
      3. And just as easy – we can remove the permissions so certificates stop being renewed.”
   4. *Action: Remove entity from group*
   5. *Action: Observe certificates expiring after 30 seconds*