ManageEngine
PAM360

Bolstering your digital frontier with effective certificate management

Sean Wright

www.manageengine.com/pam360

Certificates have become an important aspect of the operation of an organization's technology infrastructure. This certainly can be attributed to the move to have a secure form of data transmission, primarily HTTPS, as the default. While this is fantastic from a security and privacy perspective (data is now less prone to tampering and eavesdropping), it does however increase the importance of having appropriate certificate management in place.

An area of concern when handling certificates in an enterprise environment is having to deal with certificates amongst numerous teams and departments. This becomes a problem in several ways, most notably, how to securely request, transfer, and store the private keys of these certificates. Private keys are a crucial part of certificates. If a private key is ever compromised, the certificate becomes compromised, and any entity with access to this private key can use this certificate.

When certificates are stored on a system, they are often stored with default write permissions. This allows anyone with local access to the system to be able to view the private key. When multiple teams have access to a single system, this becomes a problem since it is challenging to identify who has had access to this private key. Appropriate read (and write) permissions should be set on private keys to ensure only users with a suitable need have access to the private key. Also, having an appropriate means to monitor the permissions on certificates could be helpful in identifying private keys that have permissions such as world readable permissions set on them.

In a related problem, we have seen instances where the private key for a certificate has been accidently shared. This can be via something such as a public code repository or embedded within released software. Much like the problem with having incorrect permissions on the private key, having the private key for the certificate can carry severe implications. At times, teams may decide to use wildcard certificates. When the private key is

leaked for these certificates, this results in the compromise for all sub-domains that the wildcard certificate belongs to. This not only includes current systems, but it could also include future systems running on subdomains supported by the wildcard certificate. This may even include malicious systems set up by an attacker using subdomains which can spoof.

Another problem relating to private keys is the generation of private keys. As systems become more powerful, the need to use longer, newer algorithms for generating private keys is required. It is vital to stay up to date and follow best practices. This should either be clearly documented or, even better, built into a tool to do this automatically.

Additionally, it's important to ensure that any key lengths and algorithms used to generate the private key are in alignment with the organization's security policy. An ideal solution would involve having a centralized location or service to generate and store these keys. This system would also include fine-grained access controls to keys stored and managed by the service as well as sufficient auditing. However, this may not always be possible or feasible, and private keys may still need to be generated and stored on the individual server or system using the certificate that is associated with the private key.

Coming back to the risks posed by wildcard certificates, one approach to limiting this type of risk is to avoid the use of wildcard certificates entirely. This can be done by restricting certificates to a small or even single domain; in this case, a compromise of a certificate will likely only result in a risk for that single domain. This does, however, have a trade off in terms of security and the extra overhead of additional certificates to manage. It should also be noted that even though a wildcard certificate is a single certificate, it will likely be installed in many locations. So, when it is required to replace this single certificate, it will need to be replaced in all locations where it is installed.

erhaps the most common issue that occurs when attempting to manage many certificates is the expiration of certificates. During recent years, we have seen several outages attributed to expired certificates. An example of this is an outage that affected Spotify in 2020. This impacted customers on the platform, no doubt with some questioning whether they should look to another service instead. We have also seen instances where expired certificates also had a large, negative impact on security related systems. A good example of this was with the infamous Equifax breach of 2017. It turns out that Equifax could have detected the breach a lot sooner, but it didn't due to an expired certificate on its monitoring system.

Unfortunately, it is likely that this problem is only going to get worse. As companies set up new systems and services, they require new certificates for those systems. This means more and more certificates to manage, and, as highlighted above, it only takes a single certificate to cause a severe outage for an organization. This has been made even more relevant with the recent changes from vendors like Apple, Mozilla, and Google who have recently enforced that newly issued server certificates can only have a lifespan of no more than 398 days (the equivalent of 1 year with a grace period).

While this move does carry security benefits, it introduces a burden from an operational point of view. Organizations will not only have an increase of certificates to deal with, but the occurrence of having to renew these certificates has now increased due to the shorter lifespans of these certificates. This includes overheads such as tracking expiring certificates, renewing identified certificates that are close to expiration, and installation of these certificates. The only feasible and scalable solution is automation. When managing hundreds or thousands of certificates, doing this manually is simply not going to scale well.

Thankfully, one potential solution exists already, and that is in the form of the Automated Certificate Management Environment (ACME) protocol. This is a defined protocol that allows the automation of certificate generation and certificate renewal. Having this in place allows systems to effectively manage their own certificates without the need for human intervention. We have already seen the potential of this with the rise of the CA Let's Encrypt to become the world's largest CA in a matter of years. Using automation such as ACME can also help with the problem of incorrect permissions set on a certificate's private key, since the automation piece will set the appropriate permissions on the private key. Another popular option is to leverage some cloud-based services such as AWS Certificate Manager. The additional benefit of utilizing these services is that they often integrate with other services within the cloud environment, allowing for further automation of certificate issuance and renewals.

Another problem when it comes to managing certificates, especially across multiple teams, is the problem with Certificate Authority (CA) certificates. These certificates, like all other certificates, also expire at some point. When this happens, all current certificates for that CA will need to be replaced. Likewise, if a CA is ever compromised, all certificates signed by that CA will need to be immediately replaced. Having a tool to help track where these certificates live is vital when it comes time to reissue the certificate under a new CA. Not having this information at hand will mean that certificates will likely not be replaced, and one will likely face outages like those experienced by expiring certificates. Also, depending on the number of certificates, it could take significant effort to reissue the new certificates. So, either this would need to be done in advance (giving enough time to reissue all new certificates) or, even better, automated using a protocol such as ACME.

We also have seen past instances were certificates were incorrectly issued for organizations that did not own the domain in the request. A very good example of this was when Symantec issued certificates to a domains that

the requester didn't own. This prompted Google to develop a standard called Certificate Transparency (CT). This enforces all public CAs to publish details publicly about certificates that they sign and issue.

By registering your organization's domains with a CT log monitoring tool, you can get alerts to all certificates created for the registered domains. This helps with two things. First, it helps form an complete list of certificates created for a domain by commercial CAs. This list can then be fed into some form of database or asset management tool. Second, it helps identify and alert on any certificates that may have been created without the appropriate authorization.

Tied to these two issues is the update of certificate trust stores. This becomes a problem, especially when using internal CAs. If a CA is ever compromised, not only will all certificates issued by that CA have to be reissued, the new CA certificate will have to be included into trust stores on systems and clients that connect to systems using these certificates. This can take significant effort, and if it is not performed, systems and users will experience connectivity issues. Again, having some form of automation can help in this aspect, especially leveraging tools and mechanisms for importing new CA certificates.

Lastly, a problem that teams will have to contend with is managing which CAs are being used among teams. Organizations should perform reviews of CAs and select a set that they are comfortable with using and have agreements in place. Thankfully, there are mechanisms in place that can help with this. One such mechanism is to utilize the DNS Certificate Authority Authorization internet security policy mechanism. This policy specifies which CAs are authorized to issue certificates for the domains listed in the policy.

By having a policy like this in place, you can prevent teams requesting and obtaining certificates from CAs that have not been approved by the organization. It can also enable control around central management of certificates—for example, leveraging a single CA platform to issue certificates for your organization. Many commercial CAs now come with some form of certificate management tools, so this is a means of enforcing all certificates to be generated through these tools.

Vendor note

As much as HTTPS and certificates promise web security, these comes with additional responsibilities for organizations' IT teams to enforce a sustainable, end-to-end certificate life cycle management program. While automating certificate life cycle management through tools and solutions is an option, it's also important for the solution to be able to integrate with the overall ITOps to provide holistic visibility and enable quicker follow-up actions. For instance, integrating certificate management with privileged access management helps your IT admins manage key-based authentication to privileged corporate assets from a single console without having to navigate between multiple interfaces. Similarly, integrating certificate management with IT service management helps trigger incidents automatically when a certificate expires, and integrating with enterprise mobile device management helps automate the renewal and reissuing of certificates deployed to corporate mobile devices.

About the author:

Sean Wright

Sean is currently lead application security SME at Immersive Labs, where his responsibility lies in leading the development of content for the application security offering from Immersive Labs. Sean has held a previous role in a large managed security services provider and was responsible for helping provide technical leadership for the team responsible for the application security of systems for organizations. Sean is also the co-leader of the OWASP Scotland chapter. Sean is an active member of the community and gives regular talks at local events and conferences such as BSides.

www.manageengine.com/pam360

4141 Hacienda Drive Pleasanton,
CA 94588, USA
US +1 888 204 3539
UK : +44 (20) 35647890
Australia : +61 2 80662898
www.manageengine.com/pam360

ManageEngine
PAM360