Notes on SSL

Last updated by | Lisa Liu | Nov 6, 2020 at 10:34 AM PST

Notes on SSL

Wednesday, June 19, 2019
4:37 PM

Trusting server certificate is an insecure way of communication because this might make an attacker connect to your application by sending you a cert that your application will trust without validating and start sending data with the attacker certificate that they can eventually decrypt and get the data.

Root certificates from a certain signer can only one, man in the middle will not be able to generate a root certificate for Baltimore (Microsoft's certificate) as example but he will be generating another certificate which your application will not validate or question and will make your application trust the attacker and send data to them(encrypted yes but they can decrypted with their private key as they are the issuer for the certificate). Please note that trusting the root authority of the certificate is a must and you need to use it but it is different than trusting server's certificate. Not trusting server certificate mean that your application will verify the server's identity and make sure it is Microsoft who you are talking to.

1. What is Microsoft's recommendation for implementing same or equivalent security for connecting to azure sql databases?

Microsoft recommendation is not different than any other security expert recommendation please encrypt your connections and don't trust the server certificate, Please fid this documented here:
https://docs.microsoft.com/en-us/azure/sql-database/sql-database-security-overview#transport-layer-security-tls-encryption-in-transit

As a best practice, recommend that in your application's connection string you specify an encrypted connection and not trust the server certificate. This forces your application to verify the server certificate and thus prevents your application from being vulnerable to man in the middle type attacks.
For example when using the ADO.NET driver this is accomplished via Encrypt=True and TrustServerCertificate=False. If you obtain your connection string from the Azure portal, it will have the correct settings.
Please take a look at the following article for more information about SSL for SQL.
https://docs.microsoft.com/en-us/sql/relational-databases/native-client/features/using-encryption-without-validation?view=sql-server-2017

In Short what does TrustServerCertificate means?
This option specifies whether the driver trusts the server certificate when connecting to the server using TLS.
Enabled (yes): The driver trusts the server certificate.
Disabled (no): The driver does not trust the server certificate, and instead uses a CA certificate to verify the server certificate. << recommended

Where to get Baltimore certificate from?
Download the certificate needed to communicate over SSL with your Azure SQL Database server from https://www.digicert.com/CACerts/BaltimoreCyberTrustRoot.crt.pem and save the certificate file to your local drive.

2. trust a server's leaf certificate for ssl connection is to trust the child/leaf certificate that e.g. Microsoft purchased from digicert which has Baltimore Cyber Trust Root as the root/parent. We would import this child certificate to our truststore instead of the Baltimore Cyber Trust Root. Because if we trust the root, ssl will accept any certificate signed by the same root. What is stopping an adversary to purchase a certificate which is also signed by Baltimore Cyber Trust Root? Hence, in our opinion, just trusting the root does not guarantee we are talking to Microsoft.

   Adversary can purchase a certificate that is also signed by Baltimore Cyber Trust Root. But they will not be able to present themselves as <servername>. `database.windows.net` because that domain is reserved for Microsoft. Baltimore Cyber Trust Root can guarantee that every certificate that matches the wildcard *.database.windows.net belongs to Azure SQL database but it can not guarantee that similar ones are not owned by an adversary (e.g. *.azuresqldatabase.windows.net). It's up to the user to make sure that they are specifying proper server name they are connecting to.

3. Secondly, even if we trust the leaf certificate of Microsoft, what guarantee do we have to ensure our apps are connecting to the right database and not another azure sql database owned by an adversary? There might be other measures in place to protect us against those vulnerabilities and this might be a non-issue. We are simply looking for an explanation why/how we are protected.

   If you specify your server name in the connection string, Microsoft will make sure that connection is directed to customer server and not a adversary one. So, trusting Baltimore Cyber Trust Root cert in this case means following: Server presented the certificate that matches *.database.windows.net wildcard. Baltimore cert guarantees that this is Microsoft and Microsoft guarantees that the connection will be directed to customer's server and not an adversary one.

4. Traditionally to fully secure connections between our clients and our databases at bank, we always used mutual TLS to ensure both the client and the server are authorized to talk to each other. Is mutual TLS an option we have with Azure SQL as well? That would eliminate all of the above concerns.
   SQL Azure uses TLS. It supports 1.0, 1.1 and 1.2
   https://docs.microsoft.com/en-us/azure/sql-database/sql-database-security-overview#transport-layer-security-tls-encryption-in-transit

Created with Microsoft OneNote 2016.

How good have you found this content?

😊 ☹