Software Architecture Series

Secure Architecture

Check List

When designing secure architecture, follow the following steps in the secure architecture process to ensure the resulting architecture and system are as secure as possible.

Step	Description
Threat Modeling	Goal: Identify potential threats to the system and discuss ways to mitigate them. Actions: Is there a Threat Modeling process in the organization? - Great! Make sure to join it and be aware of its results. - Make sure the process includes the four questions: 1. What do we build? 2. What can go wrong? 3. How can we mitigate that? 4. Did we succeed? - Make sure the Threat Modeling is summarized is a document If there is no Threat Modeling process in the organization:

	- Talk to the project manager / dev manager and explain the importance of the process - Volunteer to lead the process - Use the four core questions - Assign a team member to summarize the process in a document
Secure Architecture	Goal: Design Secure Architecture based on the threats defined in the Threat Modeling. Actions: - Ask for overview of the network security – where is the firewall? Are there subnets? What's in each one of them? Is there an IPS (Intrusion Prevention System)? - Ask for overview of the platform security – which operating systems are in use? Which versions? Are there outdated / unsupported versions? Is there a well defined patch management policy in place? What is the anti-virus software used? Is it updated regularly? - Design Secure Architecture for each of the following: <u>Authentication</u> 1. Decide on authentication engine. Always prefer a 3rd party product (such as Active Directory)

	2. Decide on the user store. Try to go for a hybrid user store. 3. Design the user store that is part of the software component. 4. Decide on authentication type (user/pwd, text, MFA, etc.). Work with the system analyst on that. 5. Decide on authentication protocol. Modern protocol such as OAuth2 is always preferable. Make sure the development platform supports it (most do) 6. Make sure everyone, including the IT, is in the loop. This is a cross-organization effort. <u>Authorization</u> 1. Decide on roles management – part of the authentication engine or part of the software component? Remember – specific roles are better be in the software, cross-orgs roles are better be in the auth engine. 2. If in the software – design the role component, incl. roles' table, data access 3. Design authorization implementation: a. Action Authorization – recommend the libraries to use, most dev platforms have

	great support. Usually controls access to web API. b. Data Authorization – Decide and design implementation, if needed. Usually built-in DB authorization engines are difficult to use. 4. Work closely with the dev team and system analyst to define the roles. <u>Secure Communication</u> 1. Make sure the organization has secure communication infrastructure. 2. Don't use outdated versions (SSL 1-3, TLS 1.0, 1.1) 3. Insist on using secure communication <u>Secure Code</u> 1. Make sure dev team is aware of secure code techniques 2. Make sure there is training in place 3. Introduce team to OWASP <u>Secure Data</u> 1. Define the data that should be encrypted. Not always a "blanket encryption" is a good idea.

	2. Decide on encryption strategy – using the built-in DB mechanism (preferred) or developed in-house (less preferred). Work with the DBA on that. 3. Make sure there is a secure key store in the organization. 4. If not – talk to the IT guys about creating one. Usually you can use the one built into the OS. 5. If possible, you can use a cloud-based key store such as Azure KeyVault. Logging & Monitoring 1. Make sure there are logging and monitoring mechanisms in place 2. Define security events to be logged (log-in data, validation errors, etc.) 3. Define the alerts that should be raised (high-density of requests, a lot of validation errors)
SDLC	Goal: Integrates security and privacy considerations into the development lifecycle Actions: - Make sure the team is trained in SDLC - If there is an active SDLC process – join it - Work with the devs and ensure secure architecture is implemented

Testing	Goal: To make sure the system is actually secure Actions: - Make sure penetration and load testing are planned - Work with the testers to ensure they test the right thing - Help shape the tests – Black Box vs White Box etc. - Look closely at the results - Fine tune architecture as needed
Production	Goal: Ensures the system is still secure even in production Actions: - Participate in Security Reviews. If none are planned – organize them. - Participate in pen tests conclusions. - Plan changes to the architecture as needed. - Work with the dev team to make sure changes are applied.

I Hope you enjoyed the course, and that it expanded your knowledge about secure architecture, helped you design robust & secure systems, and, of course, made you a better Architect!

For any question or comment contact me at:

memi@memilavi.com

Thanks,

Memi