Threat Modeling Report

Created on 11/15/2021 6:54:55 PM

Threat Model Name:

Owner:

Reviewer:

Contributors:

Description:

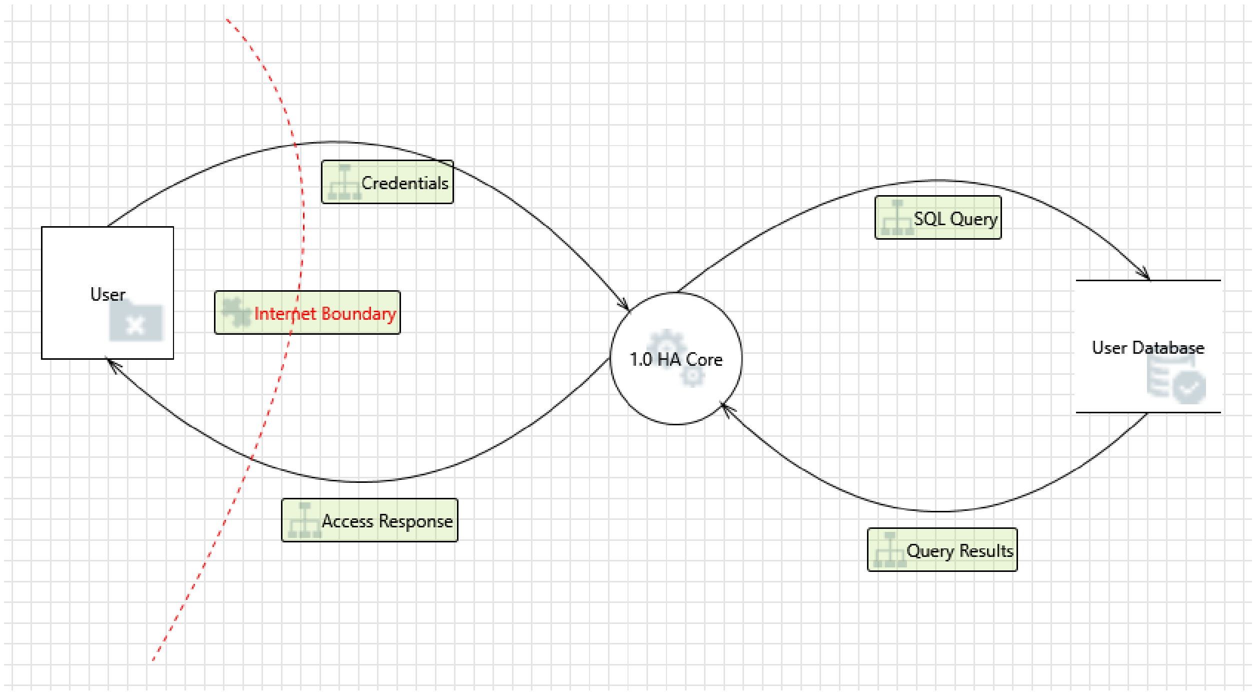
Assumptions:

External Dependencies:

Threat Model Summary:

Not Started	0
Not Applicable	9
Needs Investigation	3
Mitigation Implemented	7
Total	19
Total Migrated	0

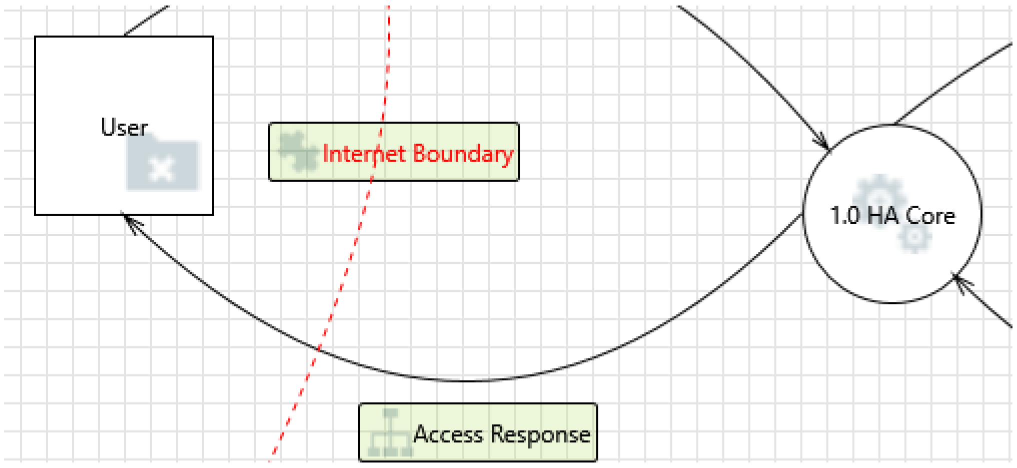
Diagram: DFD Level 1.2 Authentication



DFD Level 1.2 Authentication Diagram Summary:

Not Started	0
Not Applicable	9
Needs Investigation	3
Mitigation Implemented	7
Total	19
Total Migrated	0

Interaction: Access Response



1. Spoofing of the User External Destination Entity [State: Mitigation Implemented] [Priority: High]

Category:	Spoofing
Description:	User may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of User. Consider using a standard authentication mechanism to identify the external entity.
Justification:	User is authenticated using a standard username/password combination. Timeout counter. HA has authentication and MFA for access outside local network, https://www.home-assistant.io/docs/authentication.

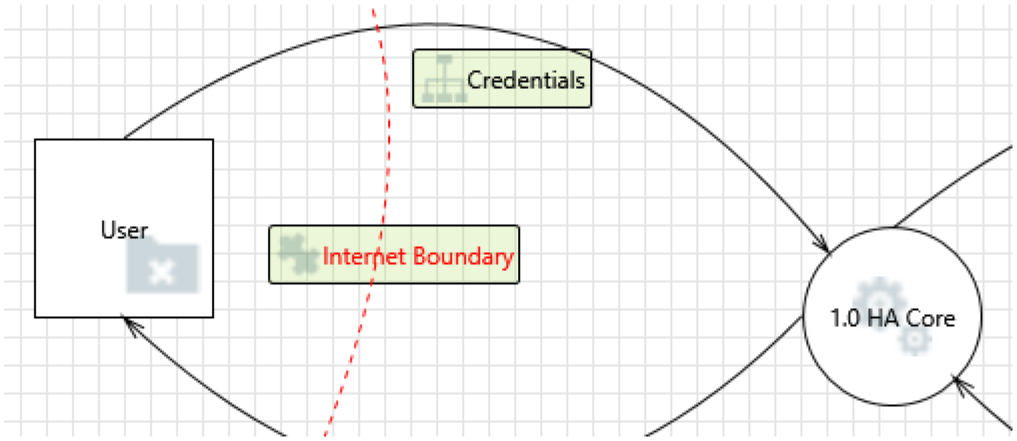
2. External Entity User Potentially Denies Receiving Data [State: Needs Investigation] [Priority: High]

Category:	Repudiation
Description:	User claims that it did not receive data from a process on the other side of the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
Justification:	Does the app that User logs in with have an authentication mechanism to check if the login attempt was heard by HA Core? A timeout counter for the request?

3. Data Flow Access Response Is Potentially Interrupted [State: Not Applicable] [Priority: High]

Category:	Denial Of Service
Description:	An external agent interrupts data flowing across a trust boundary in either direction.
Justification:	User would just not log in to HA Core and would have to attempt log-in again. Not Applicable.

Interaction: Credentials



4. Spoofing the User External Entity [State: Mitigation Implemented] [Priority: High]

Category:	Spoofing
Description:	User may be spoofed by an attacker and this may lead to unauthorized access to 1.0 HA Core. Consider using a standard authentication mechanism to identify the external entity.
Justification:	User is authenticated using a standard username/password combination. Timeout counter. Authentication is available, https://www.home-assistant.io/docs/authentication.

5. Elevation Using Impersonation [State: Not Applicable] [Priority: High]

Category:	Elevation Of Privilege
Description:	1.0 HA Core may be able to impersonate the context of User in order to gain additional privilege.
Justification:	<no mitigation provided>

6. Potential Data Repudiation by 1.0 HA Core [State: Mitigation Implemented] [Priority: High]

Category:	Repudiation
Description:	1.0 HA Core claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
Justification:	Check for logging and audit. Data logging from both the core and IoT devices is available with the logger integration, https://www.home-assistant.io/integrations/logger.

7. Potential Lack of Input Validation for 1.0 HA Core [State: Mitigation Implemented] [Priority: High]

Category:	Tampering
Description:	Data flowing across Credentials may be tampered with by an attacker. This may lead to a denial of service attack against 1.0 HA Core or an elevation of privilege attack against 1.0 HA Core or an information disclosure by 1.0 HA Core. Failure to verify that input is as expected is a root cause of a very large number of exploitable issues. Consider all paths and the way they handle data. Verify that all input is verified for correctness using an approved list input validation approach.
Justification:	Is there input validation by HA Core for user creds? Is there an account lock out after so many failed log-in attempts? It looks like the failed log-in attempt is part of the authentication though one person added a personalized detector https://community.home-assistant.io/t/failed-login-detection/4280.

8. Spoofing the 1.0 HA Core Process [State: Not Applicable] [Priority: High]

Category:	Spoofing
Description:	1.0 HA Core may be spoofed by an attacker and this may lead to information disclosure by User. Consider using a standard authentication mechanism to identify the destination process.
Justification:	To spoof HA Core and access an installed application the attacker would already need admin rights on the machine that HA Core is running on. Not applicable.

9. Data Flow Sniffing [State: Needs Investigation] [Priority: High]

Category:	Information Disclosure
Description:	Data flowing across Credentials may be sniffed by an attacker. Depending on what type of data an attacker can read, it may be used to attack other parts of the system or simply be a disclosure of information leading to compliance violations. Consider encrypting the data flow.
Justification:	What kind of connection does user use to enter in Log-in creds to HA Core?

10. Potential Process Crash or Stop for 1.0 HA Core [State: Not Applicable] [Priority: High]

Category:	Denial Of Service
Description:	1.0 HA Core crashes, halts, stops or runs slowly; in all cases violating an availability metric.
Justification:	Attacker would have to already know the machine that HA Core is running on to create a DoS attack. Not Applicable.

11. Data Flow Credentials Is Potentially Interrupted [State: Not Applicable] [Priority: High]

Category:	Denial Of Service

Description:	An external agent interrupts data flowing across a trust boundary in either direction.
Justification:	If data flow is interupted, login will just not happen and user will have to try again. Not Applicable.

12. 1.0 HA Core May be Subject to Elevation of Privilege Using Remote Code Execution [State: Not Applicable] [Priority: High]

Category:	Elevation Of Privilege
Description:	User may be able to remotely execute code for 1.0 HA Core.
Justification:	Attacker would already need admin priviliges to do this on HA Core, therfore they would not need to collect the credentials from the user login in. Not Applicable.

13. Elevation by Changing the Execution Flow in 1.0 HA Core [State: Not Applicable] [Priority: High]

Category:	Elevation Of Privilege
Description:	An attacker may pass data into 1.0 HA Core in order to change the flow of program execution within 1.0 HA Core to the attacker's choosing.
Justification:	Attacker would just not allow user to login. User would have to reenter thier creds. Not Applicable.

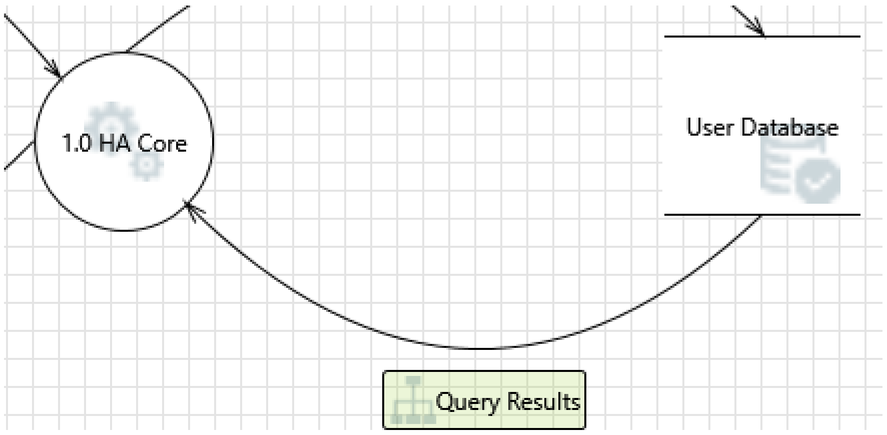
14. Cross Site Request Forgery [State: Mitigation Implemented] [Priority: High]

Category:	Elevation Of Privilege
Description:	Cross-site request forgery (CSRF or XSRF) is a type of attack in which an attacker forces a user's browser to make a forged request to a vulnerable site by exploiting an existing trust relationship between the browser and the vulnerable web site. In a simple scenario, a user is logged in to web site A using a cookie as a credential. The other browses to web site B. Web site B returns a page with a hidden form that posts to web site A. Since the browser will carry the user's cookie to web site A, web site B now can take any action on web site A, for example, adding an admin to an account. The attack can be used to exploit any requests that the browser automatically authenticates, e.g. by session cookie, integrated authentication, IP whitelisting. The attack can be carried out in many ways such as by luring the victim to a site under control of the attacker, getting the user to click a link in a phishing email, or hacking a reputable web site that the victim will visit. The issue can only be resolved on the server side by requiring that all authenticated state-changing requests include an additional piece of secret payload (canary or CSRF token) which is known only to the legitimate web site and the browser and which is protected in transit through

SSL/TLS. See the Forgery Protection property on the flow stencil for a list of mitigations.

Justification: What different ways can a user login to HA Core? If through a browser, then a CSRF could be possible. Remote access can be done using TLS/SSL with the Duck DNS add-on integrating Let&#39;s Encrypt https://www.home-assistant.io/docs/configuration/securing/.

Interaction: Query Results



15. Weak Access Control for a Resource [State: Mitigation Implemented] [Priority: High]

Category: Information Disclosure

Description: Improper data protection of User Database can allow an attacker to read information not intended for disclosure. Review authorization settings.

Justification: What kind of authorization settings are in place for User Database. Database is SQLite and it looks like access to the database is dependent on access to configuration directory, thus making standard authentication apply.

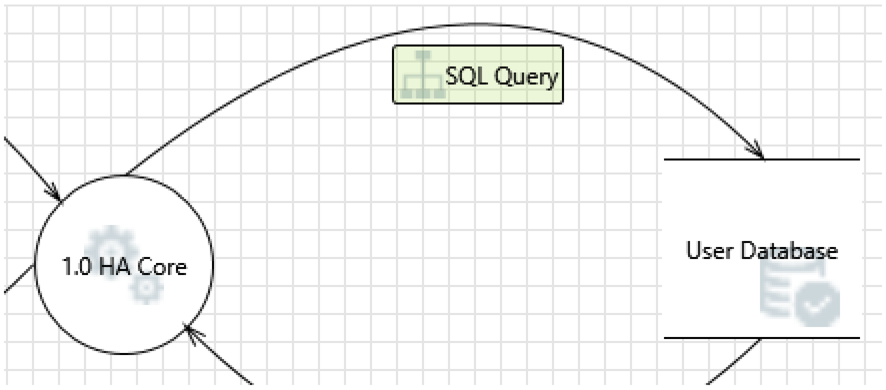
16. Spoofing of Source Data Store User Database [State: Not Applicable] [Priority: High]

Category: Spoofing

Description: User Database may be spoofed by an attacker and this may lead to incorrect data delivered to 1.0 HA Core. Consider using a standard authentication mechanism to identify the source data store.

Justification: To spoof User Database attacker would have to already have administrative access to the machine that HA Core is running on. Not Applicable.

Interaction: SQL Query



17. Potential Excessive Resource Consumption for 1.0 HA Core or User Database [State: Mitigation Implemented] [Priority: High]

Category:	Denial Of Service
Description:	Does 1.0 HA Core or User Database take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.
Justification:	Data update coordinator and System Monitor platform monitors disk usage, memory and CPU and running processes. These are housed in the configuration file.

18. Potential SQL Injection Vulnerability for User Database [State: Needs Investigation] [Priority: High]

Category:	Tampering
Description:	SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker.
Justification:	Review SQL statements for injection vulnerabilities. Database in use is SQLite and it looks to be local.

19. Spoofing of Destination Data Store User Database [State: Not Applicable] [Priority: High]

Category:	Spoofing

Description:	User Database may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of User Database. Consider using a standard authentication mechanism to identify the destination data store.
Justification:	To spoof the User Database the attacker would have to already have administrative access to the machine that HA Core is running on. Not Applicable.