# Threat Modeling Report

Created on 4/18/2017 11:04:09 AM

**Threat Model Name:** ACE Threat Model - Pairing

**Owner:**

**Reviewer:**

**Contributors:**

**Description:**

**Assumptions:**

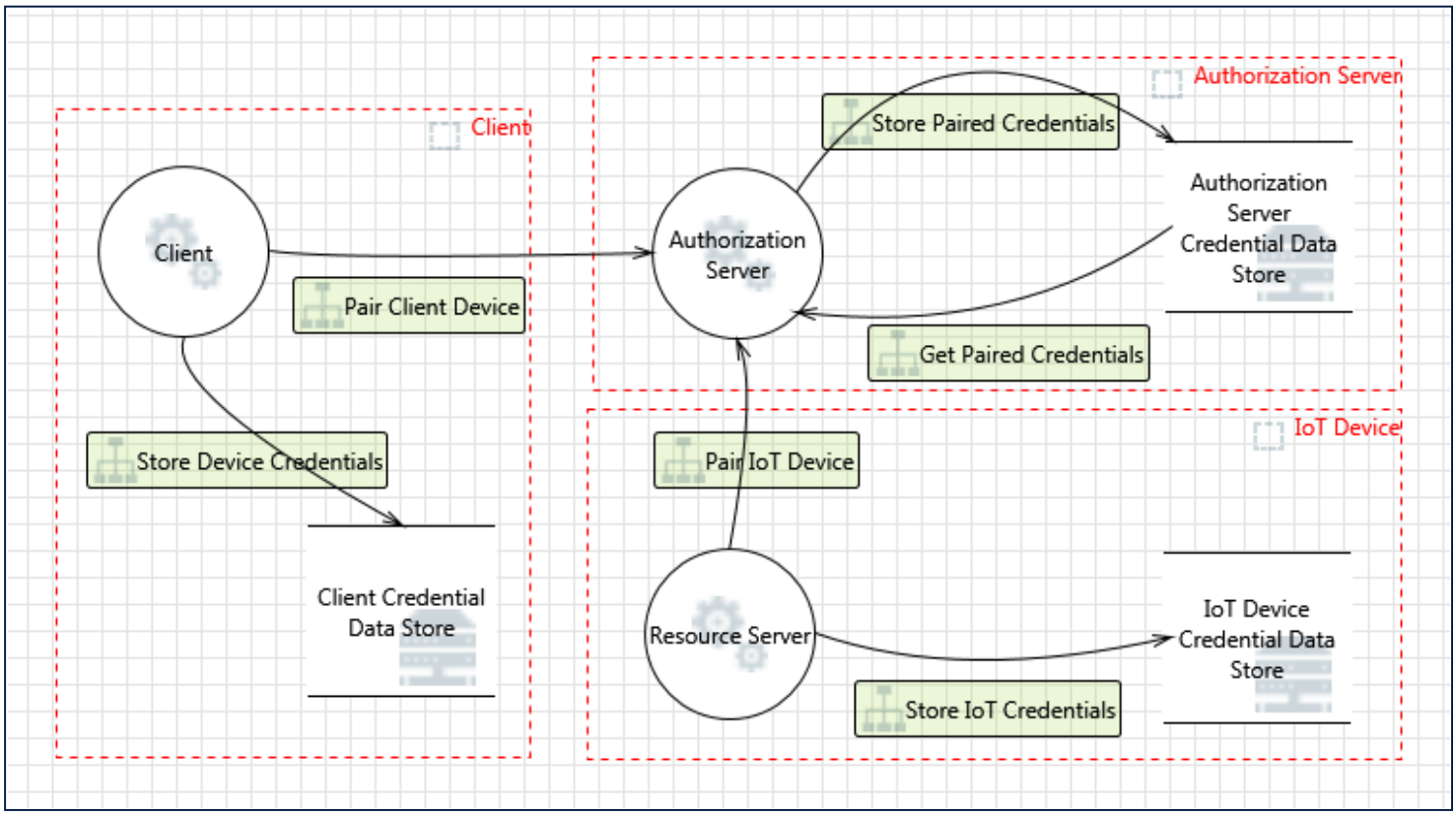
**External Dependencies:**

## Threat Model Summary:

| | |
|---|---|
| Not Started | 0 |
| Not Applicable | 17 |
| Needs Investigation | 15 |
| Mitigation Implemented | 0 |
| Total | 32 |
| Total Migrated | 0 |

---

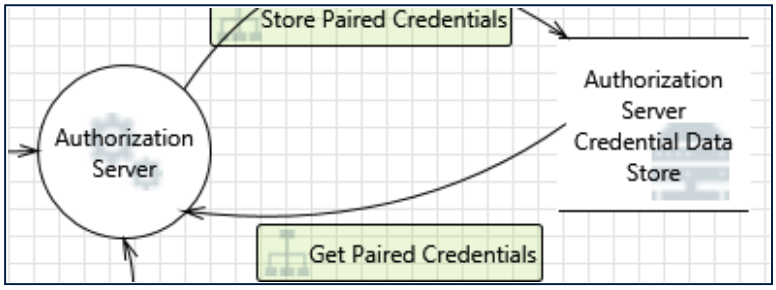
# Diagram: ACE Threat Model - Pairing




## ACE Threat Model - Pairing Diagram Summary:

| | |
|---|---|
| Not Started | 0 |
| Not Applicable | 17 |
| Needs Investigation | 15 |
| Mitigation Implemented | 0 |
| Total | 32 |
| Total Migrated | 0 |

## Interaction: Get Paired Credentials




### 1. Spoofing of Source Data Store Authorization Server Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Spoofing

**Description:** Authorization Server Credential Data Store may be spoofed by an attacker and this may lead to incorrect data delivered to Authorization Server. Consider using a standard authentication mechanism to identify the source data store.

**Justification:** This is not applicable because the data store resides on the same node.

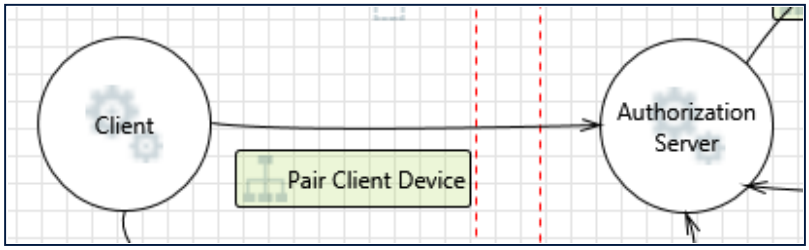
### 2. Weak Access Control for a Resource [State: Needs Investigation] [Priority: High]

**Category:** Information Disclosure

**Description:** Improper data protection of Authorization Server Credential Data Store can allow an attacker to read information not intended for disclosure. Review authorization settings.

**Justification:** Credentials need to be protected especially if resource server manages sensitive data.

## Interaction: Pair Client Device




### 3. Cross Site Request Forgery [State: Not Applicable] [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:** Not applicable because we do not expect this step to be a web-based system.

### 4. Elevation by Changing the Execution Flow in Authorization Server [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** An attacker may pass data into Authorization Server in order to change the flow of program execution within Authorization Server to the attacker's choosing.

**Justification:** Malicious client could obtain access to run other code on the AS.

### 5. Authorization Server May be Subject to Elevation of Privilege Using Remote Code Execution [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege
**Description:** IoT Client may be able to remotely execute code for Authorization Server.
**Justification:** Malicious client could obtain access to run other code on the AS.

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

**Category:** Elevation Of Privilege
**Description:** Authorization Server may be able to impersonate the context of IoT Client in order to gain additional privilege.
**Justification:** It is not applicable in this direction because all AS clients are equal.

### 7. Data Flow Generic Data Flow 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 this flow is interrupted it is indicative of a potential attack and therefore the system should not continue operating.

### 8. Potential Process Crash or Stop for Authorization Server [State: Not Applicable] [Priority: High]

**Category:** Denial Of Service
**Description:** Authorization Server crashes, halts, stops or runs slowly; in all cases violating an availability metric.
**Justification:** If the AS crashes due to DoS is it indicative of a potential attack and should not continue operating.

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

**Category:** Information Disclosure
**Description:** Data flowing across Pair Client Device 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:** Communications eavesdropping is likely in tactical environments.

### 10. Potential Data Repudiation by Authorization Server [State: Not Applicable] [Priority: High]

**Category:** Repudiation
**Description:** Authorization Server 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:** This is not applicable to IoT tactical scenarios.

### 11. Potential Lack of Input Validation for Authorization Server [State: Needs Investigation] [Priority: High]

**Category:** Tampering

**Description:** Data flowing across Pair Client Device may be tampered with by an attacker. This may lead to a denial of service attack against Authorization Server or an elevation of privilege attack against Authorization Server or an information disclosure by Authorization Server. 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:** Tampering with communications is likely in tactical environments.

### 12. Spoofing the Authorization Server Process [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Authorization Server may be spoofed by an attacker and this may lead to information disclosure by IoT Client. Consider using a standard authentication mechanism to identify the destination process.

**Justification:** Node impersonation/theft is likely in tactical environments.

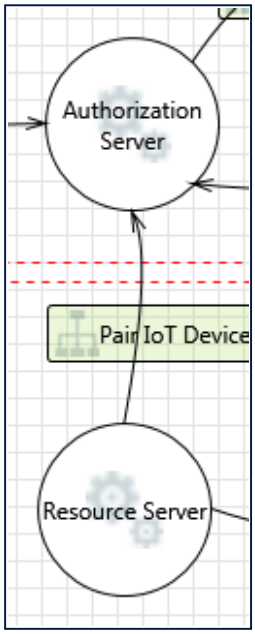
### 13. Spoofing the IoT Client Process [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** IoT Client may be spoofed by an attacker and this may lead to unauthorized access to Authorization Server. Consider using a standard authentication mechanism to identify the source process.

**Justification:** Node impersonation/theft is likely in tactical environments.

## Interaction: Pair IoT Device




### 14. Spoofing the Resource Server Process [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Resource Server may be spoofed by an attacker and this may lead to unauthorized

access to Authorization Server. Consider using a standard authentication mechanism to identify the source process.

**Justification:** Device impersonation/threat is likely in tactical environments.

### 15. Spoofing the Authorization Server Process [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Authorization Server may be spoofed by an attacker and this may lead to information disclosure by Resource Server. Consider using a standard authentication mechanism to identify the destination process.

**Justification:** Device impersonation/threat is likely in tactical environments.

### 16. Potential Lack of Input Validation for Authorization Server [State: Needs Investigation] [Priority: High]

**Category:** Tampering

**Description:** Data flowing across Pair IoT Device may be tampered with by an attacker. This may lead to a denial of service attack against Authorization Server or an elevation of privilege attack against Authorization Server or an information disclosure by Authorization Server. 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:** Tampering with communications is likely in tactical environments.

### 17. Potential Data Repudiation by Authorization Server [State: Not Applicable] [Priority: High]

**Category:** Repudiation

**Description:** Authorization Server 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:** This is not applicable to IoT tactical scenarios.

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

**Category:** Information Disclosure

**Description:** Data flowing across Pair IoT Device 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:** Communications eavesdropping is likely in tactical environments.

### 19. Potential Process Crash or Stop for Authorization Server [State: Not Applicable] [Priority: High]

**Category:** Denial Of Service

**Description:** Authorization Server crashes, halts, stops or runs slowly; in all cases violating an availability metric.

**Justification:** If the AS crashes due to DoS is it indicative of a potential attack and should not continue operating.

### 20. Data Flow Generic Data Flow 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 flow is interrupted it is indicative of a potential attack and should not continue operating.

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

**Category:** Elevation Of Privilege

**Description:** Authorization Server may be able to impersonate the context of Resource Server in order to gain additional privilege.

**Justification:** It is not applicable in this direction because all AS clients are equal.

### 22. Authorization Server May be Subject to Elevation of Privilege Using Remote Code Execution [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** Resource Server may be able to remotely execute code for Authorization Server.

**Justification:** Malicious client could obtain access to run other code on the AS.

### 23. Elevation by Changing the Execution Flow in Authorization Server [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** An attacker may pass data into Authorization Server in order to change the flow of program execution within Authorization Server to the attacker's choosing.

**Justification:** Malicious client could obtain access to run other code on the AS.

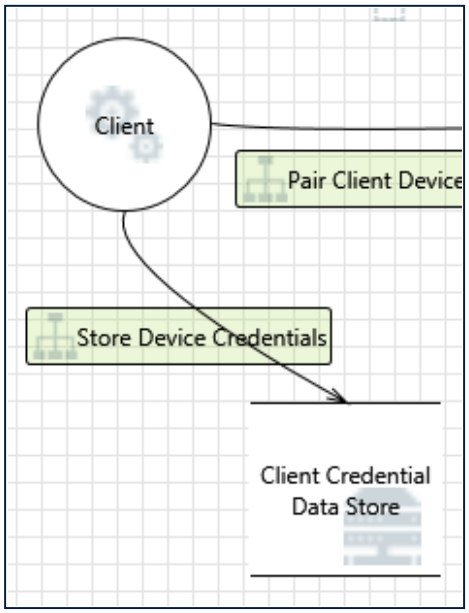
### 24. Cross Site Request Forgery [State: Not Applicable] [Priority: High]

**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:** Not applicable because we do not expect this step to be a web-based system.

## Interaction: Store Device Credentials




### 25. Potential Excessive Resource Consumption for IoT Client or Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Denial Of Service

**Description:** Does IoT Client or Client Credential Data Store 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:** This is not applicable because the data store resides on the same node.

### 26. Weak Credential Storage [State: Needs Investigation] [Priority: High]

**Category:** Information Disclosure

**Description:** Credentials held at the server are often disclosed or tampered with and credentials stored on the client are often stolen. For server side, consider storing a salted hash of the credentials instead of storing the credentials themselves. If this is not possible due to business requirements, be sure to encrypt the credentials before storage, using an SDL-approved mechanism. For client side, if storing credentials is required, encrypt them and protect the data store in which they're stored.

**Justification:** Credentials need to be protected especially if resource server manages sensitive data.

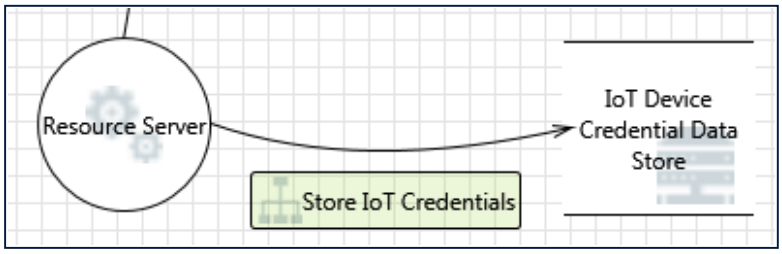
### 27. Spoofing of Destination Data Store Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Spoofing

**Description:** Client Credential Data Store may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Client Credential Data Store. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** This is not applicable because the data store resides on the same node.

## Interaction: Store IoT Credentials




### 28. Spoofing of Destination Data Store IoT Device Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Spoofing

**Description:** IoT Device Credential Data Store may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of IoT Device Credential Data Store. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** This is not applicable because the data store resides on the same node.

### 29. Potential Excessive Resource Consumption for Resource Server or IoT Device Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Denial Of Service

**Description:** Does Resource Server or IoT Device Credential Data Store 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:** This is not applicable because the data store resides on the same node (resource-constrained device)

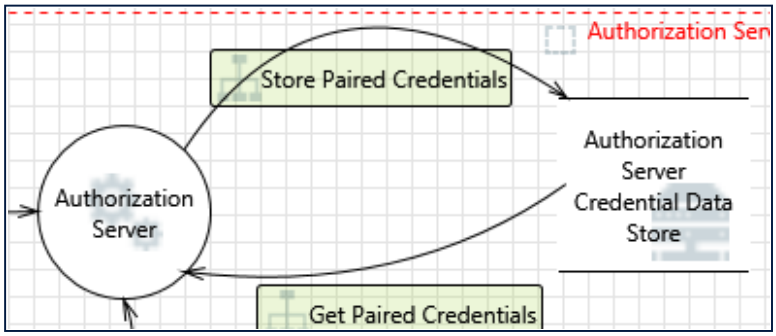
### 30. Weak Credential Storage [State: Needs Investigation] [Priority: High]

**Category:** Information Disclosure

**Description:** Credentials held at the server are often disclosed or tampered with and credentials stored on the client are often stolen. For server side, consider storing a salted hash of the credentials instead of storing the credentials themselves. If this is not possible due to business requirements, be sure to encrypt the credentials before storage, using an SDL-approved mechanism. For client side, if storing credentials is required, encrypt them and protect the data store in which they're stored

**Justification:** Credentials need to be protected especially if resource server manages sensitive data.

## Interaction: Store Paired Credentials




### 31. Spoofing of Destination Data Store Authorization Server Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Spoofing

**Description:** Authorization Server Credential Data Store may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Authorization Server Credential Data Store. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** This is not applicable because the data store resides on the same node.

### 32. Potential Excessive Resource Consumption for Authorization Server or Authorization Server Credential Data Store [State: Not Applicable] [Priority: High]

**Category:** Denial Of Service

**Description:** Does Authorization Server or Authorization Server Credential Data Store 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:** This is not applicable because the data store resides on the same node.