**Test Model**

for

Team Decided - Raft Consensus Library

## Master Test Plan Review - Stage Version 1.0 “beta ready”

During our previous LCAM submission, we’ve outlines a list of 38 tests which directly relate to use cases, functional requirements, and nonfunctional requirements of the project. A successful completion of all of those 38 tests would be directly related proof that all targets have been delivered.

## 

## Previous Test Result Revision

Of the previous 24 tests ensuring software functionality of which all were successfully passed, only one of those tests required revision from lessons learnt in the current phase. There are no previously failing tests which require reviewing and resolving.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Feature/functionality** | **Testing environment** | **Acceptance criteria** | **Role** | **Planned Stage** |
| 5 | Node authenticates using zero knowledge password proof | Dev. Evidence | Reasonable developer evidence provided | Joshua | Prototype |

Although this was successfully implemented as the previous test result shows, it’s since had to be removed due to the discovery of a security flaw in the protocol used. Since no freely integratable libraries of the ideal SRP6A protocol were available at the time, the used protocol was written by hand through a public key exchange to verify identity. Although much thought was put into the protocol, there was an irreconcilable vulnerability in the initial protocol setup where anyone who captured the first message could man-in-the-middle(MITM) the rest of that point-to-point communication. All this code was immediately removed from the project for fear of “what’s worse than bad crypto? Crypto you don’t know is bad”. It all comes back to the “don’t write your own crypto” lesson, which we’ve learnt from this extends to crypto protocols. So after some failed attempts at implementing SRP6A from the protocol docs themselves, the search begun again to locate open source licensed SRP6A code, which after multiple days of searching was eventually found. However during implementation it became clear that it was incompatible with our P2P architectures without major rework. SRP6A is built around server/client TCP communication, however we’re functioning in a P2P UDP style, and although around 20-30 hours were sunk into trying to get the code to function, it could not do so at the reliability level we’re aiming for with this project. We eventually fell back on simply using the SHA256 hash of the supplied password as a simple key derivation function, and using it as the symmetric key for all inbound/outbound packets. The packets are encrypted with industry standard AES. Nodes are only able to communicate if they’re able to successfully decrypt the received packets, so this still provides security based on the strength of the guessable password however there is no key exchange protocol. Although we do plan to revisit this, a full SRP6A UDP/P2P implementation would prove as difficult as consensus itself, and as such falls outside the reasonable expectations of this subject. When the code is released publically, we will ensure reasonable signage of the security used so developers can make their own informed decision.

## Version 1.0 Test Results

Referring back to the Master Test plan from the LCAM document, the following are the set of tests which we’re expecting completion by this IOCM project milestone. We’ve called that milestone “version 1.0”, which correlates with “beta ready, no known bugs”.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Feature/functionality** | **Testing environment** | **Acceptance criteria** | **Role** | **Planned Stage** |
| 25 | Full project code review | Dev. Evidence | Reasonable developer evidence provided | Joshua | Version 1.0 |
| 26 | Redesigned unit testing suite for extended verification of existing functionality | Dev. Evidence | Reasonable developer evidence provided | Sean | Version 1.0 |
| 27 | Cluster can grow upon new node | Demo | Evidence of demo completing action | Joshua | Version 1.0 |
| 28 | Cluster can shrink upon losing node | Demo | Evidence of demo completing action | Sean | Version 1.0 |
| 29 | UAS can attempt to change leader of cluster | Demo | Evidence of demo completing action | Joshua | Version 1.0 |
| 30 | Persistent Log implementation  (“Log compaction”) | Demo | Evidence of demo completing action | Sean | Version 1.0 |
| 31 | **(Optional feature)** Support for upgrade path | Demo | Evidence of demo completing action | Joshua | Version 1.0 |
| 32 | **(Optional feature)** Completed performance analysis/optimization of code | Dev. Evidence | Reasonable developer evidence provided | Sean | Version 1.0 |

### 

### Test 25 - Full project code review

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | N/A - Full project code review | | | | | | |
| **Test Type** | | Developer Evidence | | | | | | |
| **Test Description** | | Full review of code base, implement a much cleaner approach to the underlying multi-threaded library | | | | | | |
| **Pre-Conditions:** | | Prototype library | | | | | | |
| **Post-Conditions:** | | Production ready library | | | | | | |
| **Notes:** | |  | | | | | | |
| **Results** | | Pass | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 | Code refactor | | | Easier to read code base and logic. Simpler design. Added reliability. Resolve dealocking issues. Drastically reduce complexity of multithreading. | Relevant version control repository commits | 1.1 | Pass | |
| **Evidence ID** | | | **Evidence** | | | | | |
| 1.1 | | |  | | | | | |

### 

### Test 26 - Redesigned unit testing suite for extended verification of existing functionality

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | Redesigned unit testing suite for extended verification of existing functionality | | | | | | |
| **Test Type** | | Developer Evidence | | | | | | |
| **Test Description** | | Review of code test suite | | | | | | |
| **Pre-Conditions:** | | Prototype library | | | | | | |
| **Post-Conditions:** | | Produced a unit testing library which supports using inheritance, and reaching a high level of code coverage | | | | | | |
| **Notes:** | |  | | | | | | |
| **Results** | | Pass | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 | Code refactor | | | Test library supports inheritance, high level code coverage added | Relevant version control repository commits | 1.1 | Pass | |
| **Evidence ID** | | | **Evidence** | | | | | |
| [1.1](https://bitbucket.org/teamdecided/raftconsensuslibrary/commits/eb704cf44f78ffe112195ffca4a6e689a63cf14d) | | |  | | | | | |

### 

### ~~Test 27 - Cluster can grow upon new node~~

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | Cluster can grow upon new node | | | | | | |
| **Test Type** | | Developer evidence | | | | | | |
| **Test Description** | | N/A | | | | | | |
| **Pre-Conditions:** | | N/A | | | | | | |
| **Post-Conditions:** | | N/A | | | | | | |
| **Notes:** | | This was part of the work item “Dynamic cluster membership”. We’ve since made an evidence based decision to not pursue implementing this feature due to the expected time requirements far exceeding time available. | | | | | | |
| **Results** | | No Action required | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 |  | | |  |  |  |  | |
| **Evidence ID** | | | **Evidence** | | | | | |
|  | | |  | | | | | |

### 

### ~~Test 28 - Cluster can shrink upon losing node~~

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | Cluster can shrink upon losing node | | | | | | |
| **Test Type** | | Developer evidence | | | | | | |
| **Test Description** | | N/A | | | | | | |
| **Pre-Conditions:** | | N/A | | | | | | |
| **Post-Conditions:** | | N/A | | | | | | |
| **Notes:** | | This was part of the work item “Dynamic cluster membership”. We’ve since made an evidence based decision to not pursue implementing this feature due to the expected time requirements far exceeding time available. | | | | | | |
| **Results** | | No Action required | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 |  | | |  |  |  |  | |
| **Evidence ID** | | | **Evidence** | | | | | |
|  | | |  | | | | | |

### 

### ~~Test 29 - UAS can attempt to change leader of cluster~~

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | UAS can attempt to change leader of cluster | | | | | | |
| **Test Type** | | Developer evidence | | | | | | |
| **Test Description** | | N/A | | | | | | |
| **Pre-Conditions:** | | N/A | | | | | | |
| **Post-Conditions:** | | N/A | | | | | | |
| **Notes:** | | This was part of the work item “Dynamic cluster membership”. We’ve since made an evidence based decision to not pursue implementing this feature due to the expected time requirements far exceeding time available. | | | | | | |
| **Results** | | No Action required | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 |  | | |  |  |  |  | |
| **Evidence ID** | | | **Evidence** | | | | | |
|  | | |  | | | | | |

### 

### Test 30 - Persistent Log implementation (“Log compaction”)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | Persistent Log implementation (“Log compaction”) | | | | | | |
| **Test Type** | | Developer Evidence | | | | | | |
| **Test Description** | | Unit and Integration test added log compaction feature | | | | | | |
| **Pre-Conditions:** | | Prototype library | | | | | | |
| **Post-Conditions:** | | User is able to persistently store their key-value store information to disk | | | | | | |
| **Notes:** | |  | | | | | | |
| **Results** | | Pass | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 | Run test suite against “Persistent Log” | | | All test pass | Version control commits, unit tests results | 1.1, 1.2 | Pass | |
| **Evidence ID** | | | **Evidence** | | | | | |
| [1.1](https://bitbucket.org/teamdecided/raftconsensuslibrary/commits/84ab9d88ed8bc7f334d109921625527071dd9ed3) | | |  | | | | | |
| 1.2 | | |  | | | | | |

### 

### ~~Test 31 - (Optional feature) Support for upgrade path~~

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | (Optional feature) Support for upgrade path | | | | | | |
| **Test Type** | | Developer evidence | | | | | | |
| **Test Description** | | N/A | | | | | | |
| **Pre-Conditions:** | | N/A | | | | | | |
| **Post-Conditions:** | | N/A | | | | | | |
| **Notes:** | | This was part of the work item “Support for upgrade path”. We’ve since made an evidence based decision to not pursue implementing this feature due to the expected time requirements far exceeding time available. Also in this case there was not a sufficiently reasonable benefit for the feature to be added as well. | | | | | | |
| **Results** | | No Action required | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 |  | | |  |  |  |  | |
| **Evidence ID** | | | **Evidence** | | | | | |
|  | | |  | | | | | |

### 

### Test 32 - (Optional feature) Completed performance analysis/optimization of code

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case** | | (Optional feature) Completed performance analysis/optimization of code | | | | | | |
| **Test Type** | | Developer Evidence | | | | | | |
| **Test Description** | | Developer provides evidence of performance analysis/optimisation of code | | | | | | |
| **Pre-Conditions:** | | Code has some functionality which may be optimised | | | | | | |
| **Post-Conditions:** | | Code which performs some form of functionality faster or more efficiently | | | | | | |
| **Notes:** | |  | | | | | | |
| **Results** | | Pass | | | | | | |
| **Step.** | **Step Description** | | | **Expected Result** | **Evidence Description** | **Evidence ID(s)** | **Result** | |
| 1 | Developers perform reviews of code | | | Optimisations are found and implemented in the code | Link to the code file which contained the optimisations | 1.1 |  | |
| **Evidence ID** | | | **Evidence** | | | | | |
| [1.1](https://bitbucket.org/teamdecided/raftconsensuslibrary/src/2e50b45a8531a4c9072742b2403a2cc813ed6731/RaftConsensus/RaftConsensus/Consensus/RaftConsensus.cs) | | | Description of change: We found an optimisation to our implementation of the protocol. Instead of the leader waiting for the next heartbeat to be sent out before giving new entries to nodes to bring them up to date, the leader will instead directly respond with the next entry upon receiving a commit reply. This brings the total round trip time for an update from (heartbeat interval + (2 \* round trip time)) to just (2 \* round trip time). | | | | | |

### 