# University of Birmingham

# Software Engineering Project Proposal

| Student Name | Haodong Zuo | Student ID: | 2105646 |
|---|---|---|---|
| Supervisor Name | Vincent Rahli | | |
| Project Category/Topic | Software Engineering | | |
| Project Title | Implementation and evaluation of Byzantine-Resilient Real-Time Reliable Broadcast | | |

**Project Aim:**

- **Description**

  A Byzantine broadcast protocol will be implemented which achieves real-time and reliable. By comparing with real-time reliable protocols and existing Byzantine solutions, it will be evaluated.

- **Significance and Relevance**

  Due to hardware errors, network congestion or disconnection, and malicious attacks, computers and networks may exhibit unpredictable behavior. Existing protocols failed to tolerate attacks or support real-time. RT-ByzCast is the first real-time Byzantine reliable broadcast protocol which meets the demand of cyber-physical systems dealing with the failures and compling reliable and real-time in communication (Kozhaya, Decouchant and Esteves-Verissimo, 2019).

  A distribute system is generally composed of multiple nodes with equal status. However, machines in a distributed system have different configurations, and the services running on them may also be implemented in different languages and architectures; the nodes are connected through the network, and the network bandwidth, Delay and packet loss rate are different which brings uncertainty to the distribute system. As a result, it is necessary to have some strictly defined protocols to be resilient and fast.

  In this thesis, I am going to investigate RT-ByzCast by implementing the protocol and thoroughly evaluating my implementation.

**Related work:**

The implementation is based on the paper RT-ByzCast, and Cachin's book is considered to be a tool book for me to acquire basic knowledge on it. Lamport's work proposed the Byzantine Generals Problem and Amir's article is thought to be a guidance for me to evaluate the implementation.


Kozhaya, D., Decouchant, J. and Esteves-Verissimo, P. (2019). RT-ByzCast: Byzantine-Resilient Real-Time Reliable Broadcast. IEEE Transactions on Computers, 68(3), pp.440–454.

Cachin, C. (2014). Introduction to reliable and secure distributed programming. Springer.

Amir, Y., Coan, B., Kirsch, J. and Lane, J. (2011). Prime: Byzantine Replication under Attack. IEEE Transactions on Dependable and Secure Computing, 8(4), pp.564–577.

Castro, M. and Liskov, B. (2002). Practical byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems, 20(4), pp.398–461.
Lamport, L. (1983). The Weak Byzantine Generals Problem. Journal of the ACM, [online] 30(3), pp.668–676. Available at: https://people.eecs.berkeley.edu/~luca/cs174/byzantine.pdf.

**Project Objectives/Deliverables:**

As primary objectives, I will make myself familiar with OMNet by starting with some examples. RT-ByzCast will be implemented including proof of life mechanism and broadcast part. Evaluation of the protocol will be complied after that.

As secondary objectives, I could potentially compare the protocol with other protocols such as PRIME, try using different kinds of crypto mechanisms, implement a recovery mechanism to bring back self-crashed nodes.

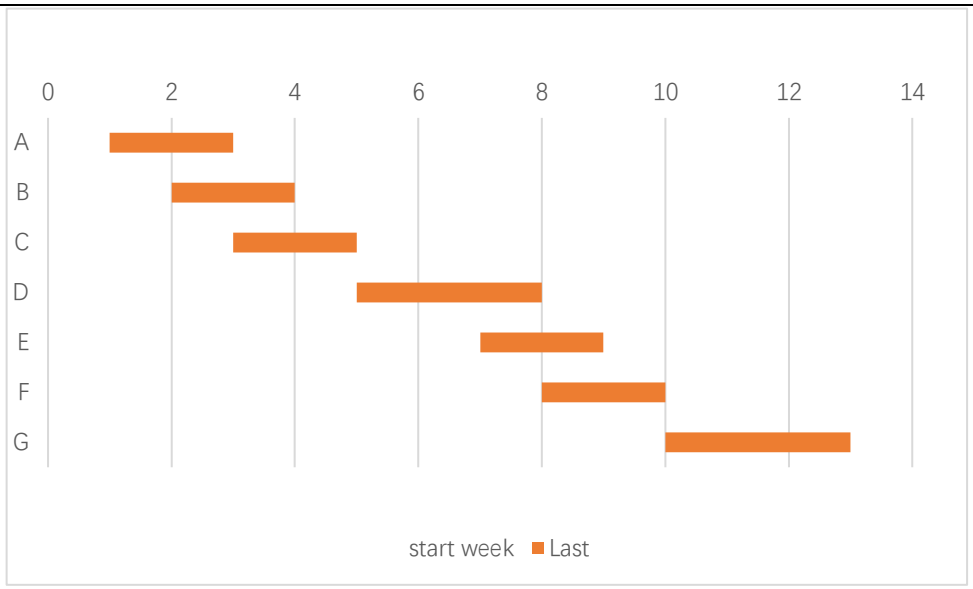
**Methodology:**

With the guidance of the supervisor, there are four parts in methodology to meet the project objectives:
(1)proof of life
Each node in the system judges itself by this function. If it cannot receive enough echoes from other nodes it send heartbeats to, it will crash itself to resist byzantine problem.
(2)Reliable broadcast
The number of non-byzantine nodes and communication rounds will be used as factors to evaluate whether the protocol is reliable or not.
(3)Integrate within OMNet++
The protocol and tests will be integrated with OMNet++ by simulating real-time communication in a distribute system.
(4)Evaluate
Software real time and byzantine resistant will be evaluated by monitoring broadcast and consensus of nodes in the distribute system.
The speed of the broadcast and the number of the transitions by comparing with other protocols.

**Project plan:**

- **Feasibility**

  software, hardware and development environment can be fully guaranteed and references to relevant materials are provided by the supervisor.
  I am familiar with information security which helps me to implement the project.
  I worked with python and have faith in using C++ with guidance of my supervisor.
- **Gantt chart**




- **Explanation of Gantt chart**

Task A: Read related books and documents to understand Byzantine broadcast in distribute system. Plan for the project and finish writing the proposal.

TaskB: Learn and master relevant technical knowledge (Omnet++ and C++), and proceed to start with basic examples.

TaskC: Introduce background knowledge, theoretical basis and technical support; complete the first chapter of the thesis

TaskD: Implement the protocol with four parts of the functions and finish writing the methodology part of the thesis.

TaskE: Evaluate RT-ByzCast with other protocols and further optimize related algorithms;

TaskF: Complete the system and the first draft of the paper.

TaskG: Revise and improve the thesis.

**Risks and contingency plan:**

- **Risks**

Completion of the target date ahead, but the functions of the system does not meet the requirement or expectation.
When implementing related functions, some unfamiliar areas need more time in design and implementation than expected
Some necessary functions cannot be implemented using existing code and libraries, and new libraries or new functions have to be developed by oneself.

- **Contingency plan:**

Discuss the project with the supervisor and make new plan for improvement.
Ask supervisor for more technique supports to deal with the lack of libraries.

**Hardware/Software Resources**

Windows 64 bit operating system with C++ compilation environment

OMNet++ :

It provides a C++ library, which consists of the simulation kernel and some tools used to create simulation components (such as random number generation, statistics collection, topology discovery, etc.), an IDE environment for designing, running and evaluating simulation; extended interface for real-time simulation.

Compared with similar tools like OPNet、NS2, it is open source and highly customized.

**Data**

No datasets are required to be used in this project.