

The background features a large, semi-transparent Bitcoin coin centered over a blue-toned line graph. The coin's details, including the 'BITCOIN' and 'Satoshi Nakamoto' inscriptions, are visible. The graph shows an upward trend with some fluctuations. Decorative elements include a vertical column of dots on the left and a small crosshair with a triangle on the top right.

# Group 27 - Based Bitcoin Core Proposed Feature Enhancement

Amy Cui, David Curtis, Jagrit Rai, John Alajaji, Logan Cantin, Matthew Vandergrift

Video: <https://youtu.be/iRwjqRWI0cw>



# Team

## John Alajaji

(Member)

- Report: Abstract, Feature Motivation, Use Case 1
- Presentation:


## David Courtis

(Presenter)

- Report: Enhancement 1, SAAM NFR Analysis, Use Case 2, Currency Adjustment
- Presentation: Much script writing, slides structure, Use Case, Enhancement 1, Lessons and Limitations.

## Jagrit Rai

(Presenter)

- Report: Risks, Lessons Learned and Limitations, Conclusion
  - Presentation:
- 

## Logan Cantin

(Member)

- Report: Introduction, Enhancement 2
- Presentation: Enhancement 2, Introduction, Concurrency


## Amy Cui

(Team Leader)

- Report: SAAM Analysis, Impacts on Stakeholders, NFR Analysis, Risks, Chosen Enhancement
- Presentation: General formatting, Risks and Limitations (script and pres.), SAAM Analysis - Stakeholders, NFR, Choice

## Matt Vandergrift

(Member)

- Report: Feature Motivation, updated conceptual architecture, impacted files and directories, testing of enchantment and conclusion
  - Presentation: Testing, Enhancement 1, Impacted Files and directories
- 



**01** Introduction  
**02** Feature Overview

**03** Enhancement 1  
**04** Enhancement 2

**05** SAAM Analysis  
-> SAAM Analysis - Stakeholders  
-> SAAM Analysis - NFR Attributes  
-> SAAM Analysis - Choice

**06** High Level Architectural Analysis  
-> Introduced Components + Dependencies  
-> Impacted Files  
-> Important Risks and Limitations

**07** Plans for Testing

**08** Concurrencies

**09** Use Case - Currency Adjustment

**10** Limitations and Lessons Learnt

**11** Conclusion





01.

# Introduction





# Introduction



- There will only ever be 21 Million Bitcoin
- Additionally when a user loses their bitcoin wallet their bitcoin is lost forever
- This leads to deflation of the bitcoin currency, which then:
  - Disincentivizes users from spending their bitcoin
  - Can eventually lead to the collapse of the bitcoin system





02.

# Feature Overview

# Feature Overview

- Control the bitcoin inflation rate to stimulate spending





03.

# Enhancement 1





# Enhancement 1

In the report we include a full and comprehensive rundown here is a short selection of some which are instructive for understanding the updated conceptual architecture.

## Transaction Measurement:

Measuring the metrics of transactions processed within a time frame, such as a moving average or a sliding window.

## Reward Adjustment:

Determines how the block reward should be adjusted based on the measured transaction metrics.

## Monitoring and Reporting:

Reporting on the impact of the adjustments, and providing visibility to network participants.

## Governance and Updates:

Updating DDRA system's parameters and algorithms.

## Reward Adjustment → Transaction Measurement:

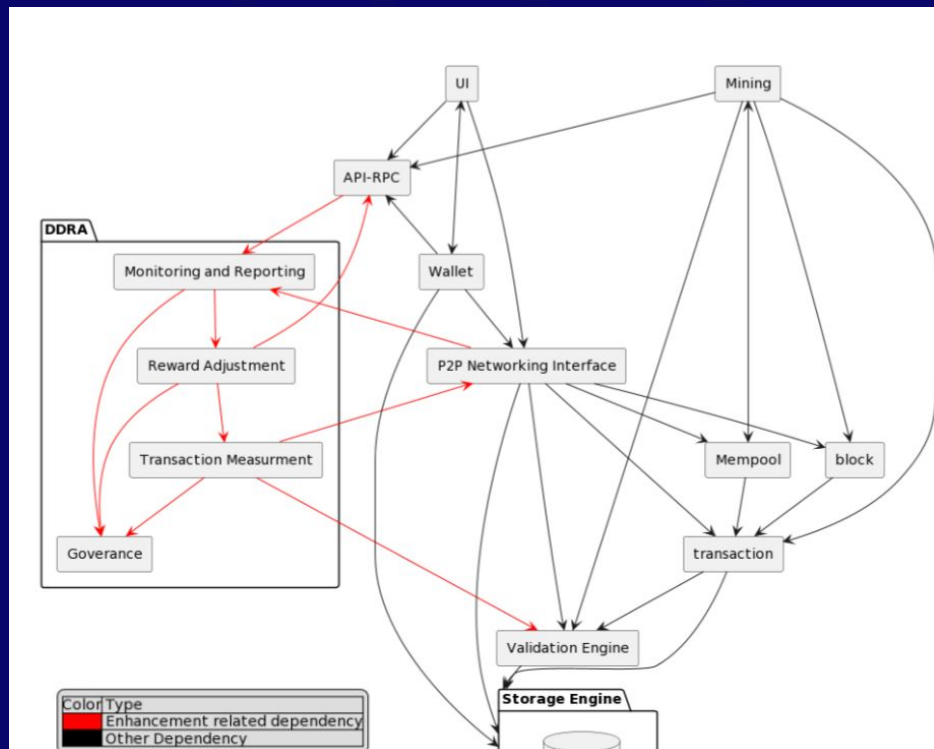
Depends on the metrics attained by the transaction measurement

## Monitoring and reporting → Reward adjustment:

Ensures rewards calculated and other metrics are accessible

## \* → Governance and Updates:

Incorporate future changes within the algorithmic calculations





04.

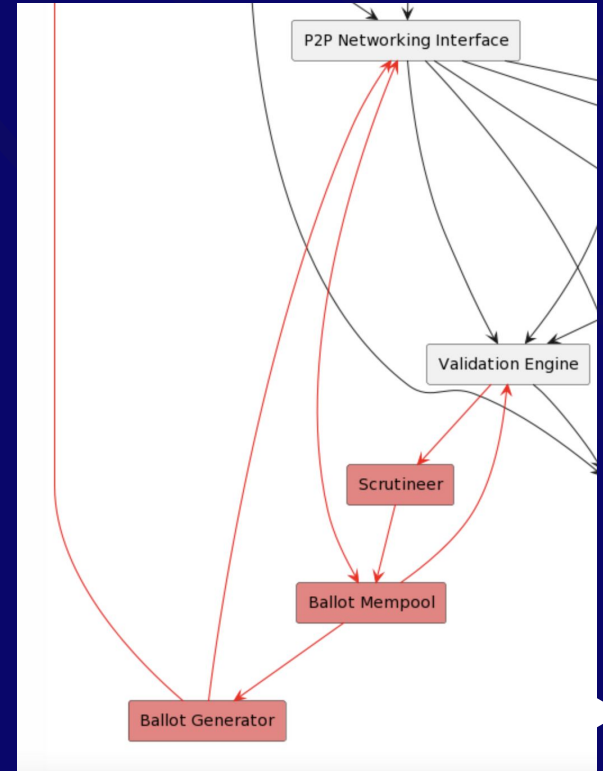
# Enhancement 2

# Enhancement 2

- **Decentralized voting system** to adjust inflation rate using **Proof of Work**
- **Ballots** are the new data structure which represents a vote
- Voting periods are 2 weeks long and at the end, a new block reward is decided upon

## New Components:

- **Ballot Generator**: Does the Proof of Work to generate new ballots
- **Ballot Mempool**: Stores all the ballots corresponding to the current voting period
- **Scrutineer**: Responsible for counting the ballots at the end of the voting period and deciding on the new block reward, which remains in effect until the end of the next voting period





05.

# SAAM Analysis

# SAAM Analysis - Stakeholders

Stakeholders	Most Important NFRs regarding the enhancement
<b>Developers</b>  Responsible for designing and implementing the adjustment system.	<i>Maintainability:</i> The system must be easy to maintain, especially during high periods of activity (high transaction period and voting period).  <i>Performance:</i> The new enhancements should bring better performance in most cases.  <i>Scalability:</i> The system should have high scalability to handle increases in transaction volume based on enhancement.  <i>Security:</i> The enhancement should not introduce new security vulnerabilities.
<b>Merchants</b>  Ensure that transaction fees are kept low and predictable and that the currency is stable. This makes Bitcoin more attractive to used as a payment method with directly impacts Merchants.	<i>Reliability:</i> Want to ensure that the system is reliable and stable after enhancements are added.  <i>Security:</i> Enhancement introduces higher chance of manipulation which can compromise the trust and reliability of Bitcoin as a form of payment.  <i>Usability:</i> Enhancement should not introduce any usability issues that would make it difficult to use Bitcoin as a payment method.
<b>Miners</b>  Miners are responsible for processing transactions and validating votes.	<i>Performance:</i> Enhancements should not negatively impact ability to earn block rewards  <i>Reliability:</i> Enhancements should not increase the risk of a for in the network  <i>Scalability:</i> Enhancement should be able to handle increases in amount of mining at one time.
<b>Users</b>  A automated adjustment system would make the network more efficient while a voting system would allow users to have a say in the direction of the Bitcoin Core Network.	<i>Latency:</i> System should be optimized to minimize any delays in transaction processing.  <i>Reliability:</i> Enhancement should be reliable and stable during high periods of use.  <i>Security:</i> Enhancement should not introduce new attack opportunities to users.  <i>Usability:</i> There should not be an increase in difficulty for usability of Bitcoin Core due to added enhancement.

- **Developers** (both open-source and the Bitcoin Core development team): *responsible for developing the system*
  - focus on ensuring maintainability for optimized development velocity, performance and scalability for feasible deployment, and security to prevent malicious use.
- **Merchants:** *any businesses that accept Bitcoin as a form of payment for their goods or services*
  - a loss in the prioritized NFRs will directly impact payment utility.
- **Miners:** responsible for validating transactions and adding them to the block chain
  - care about maximising hashrate mining, guarantee the network is consistently minting transactions, and incorporating more computing power.
- **Users:** *individuals or businesses who use Bitcoin as a means of payment or store of value*
  - prioritize NFRs in a similar way to merchants, but also want personal transactions or network decisions to be quickly executed.

NFRs		Enhancement I	Enhancement II
<i>Latency</i>	The time delay between initiating a request and receiving a response.	<i>Low:</i> Latency influences the time it takes for the system to process and adjust the block rewards. A low latency system would be able to adjust the block rewards quicker. This is difficult to achieve as adjustments must be made in real-time, and concurrently between clients to ensure consensus with minimal wasted communication.	<i>Medium:</i> The systems voting latency must be optimized to ensure that users receive prompt feedback on the status of their vote and that voting results are calculated quickly. Adjustments must also be made in real-time, but are made considerably less often.
<i>Maintainability</i>	The ability of a software system to be easily maintained and modified over time.	<i>High:</i> The algorithmic load is low and remains stable over time, as such, no maintenance is required as unexpected situations rarely arise and the system is entirely autonomous and self-preserved.	<i>Medium:</i> The voting system must perform well, ensuring that it can handle a high volume of votes without slowing down or crashing. Maintenance of the voting system is fundamental to ensuring a stable function.
<i>Performance</i>	How well a system performs in terms of speed, responsiveness, and resource utilization.	<i>High:</i> Performance will not be a concern in the automated adjustment system, as the relative computational load is very meagre, and no cyclic network interaction is required.	<i>Low:</i> The system will need to be optimized to handle the increased workload efficiently during voting periods, this will directly impact the network congestion and affect the load of the p2p networking component.
<i>Reliability</i>	The ability of a system to perform its intended function without failure, errors or breakdowns.	<i>High:</i> The system must be reliable to ensure that the block rewards remain stable which is essential for maintaining the incentive structure of the network	<i>High:</i> The voting must be available via multiple channels to ensure high availability to ensure that it operates consistently and accurately over time
<i>Security</i>	Ensures that the system is secure and protected against cyber threats.	<i>High:</i> The automated adjustment system is resilient to attacks as it supports a decentralized and bias-free method of adjusting protocol, and does not accept user input. This makes the system highly secure from attacks.	<i>Low:</i> The voting system could potentially introduce a new attack vector. The voting system must be secure so that votes are not tampered with or manipulated, and that there is a low incentive for collusion. The possibility of collusion is a difficult consideration, as a few large mining pools hold much power over the network.
<i>Scalability</i>	The ability of a system to handle increasing amounts of work or data in a responsive and efficient manner.	<i>High:</i> All clients connected and making transactions generally participate in the calculation of these metrics, thereby, the system must be able to handle a large amount of data and ensure that the data meets consensus and is valid; however, since each client is capable of making independent decisions, scalability is uncomplicated to implement.	<i>Medium:</i> If the number of voters grows significantly then the system must be able to handle a large number of votes while maintaining performance and security. Each client must implement an additional layer of communication to achieve this.
<i>Usability</i>	The ease with which users can interact with the system to achieve goals efficiently and with satisfaction.	<i>High:</i> The dynamic reward adjustment system is only modifiable to developers, thereby, will not require general user interaction. This indicates that the system may not require much user interaction support.	<i>Low:</i> The voting system must be user-friendly, ensuring that it is easy to use and understand for all Bitcoin users that wish to vote with minimal user error. This includes the general public.
<i>Robustness</i>	The ability of a system to continue operations while handling errors and exceptions without crashing.	<i>Low:</i> An automated system will be relatively rigid in decision-making compared to human analysis and voting, thereby, is not as robust to adverse conditions without human updates. The governance and updates module does allow modification to the algorithm but should be rarely updated.	<i>High:</i> Adverse conditions can be dynamically handled through the direct decisions of human agents, thereby, allowing the system to adapt to complex scenarios relatively fast and effectively.

# SAAM Analysis

## NFR Attributes

### Enhancement I

1. Struggles with latency
2. Algorithm is low and remains stable over time. Very little maintenance required.
3. Stable performance
4. Better security with no new attack vectors
5. High scalability due to clients ability to make independent decisions
6. No user interaction required
7. Low robustness due to rigid decision-making

Versus

### Enhancement II

1. Few latency concerns
2. Requires regular maintenance
3. Struggles with performance during voting periods
4. Introduces new attack vector through vote tampering and manipulation
5. Medium scalability with some challenges
6. Need to implement user-friendly system
7. High robustness through human agents

# SAAM Analysis - Choice

The automated adjustment system addresses several key Non-Functional Requirements (NFRs), such as maintainability, performance, reliability, security, scalability, and usability, all of which are highly desired by stakeholders.

## Downsides:

- a need for frequent adjustments to achieve low latency
- robustness of the automated system is limited.

## Advantages:

System has autonomy and self-preservation.

- low algorithmic load - system doesn't require maintenance of the voting system, ensuring stable functionality.
- doesn't need cyclic network interaction, making performance a non-issue.
- more resilient to attacks, due to no user interaction
- scalability advantage since there is no live propagation of a large load





06.

# High Level Architectural Analysis



# Introduced Components + Dependencies

Note that the 2nd level dependencies we covered in section 3.

## Modified Component:

### API/RPC:

Need to be modified to account for the variable coinbase transaction value.

## New High Level Dependencies:

### Transaction Measurement -> P2P Network Interface:

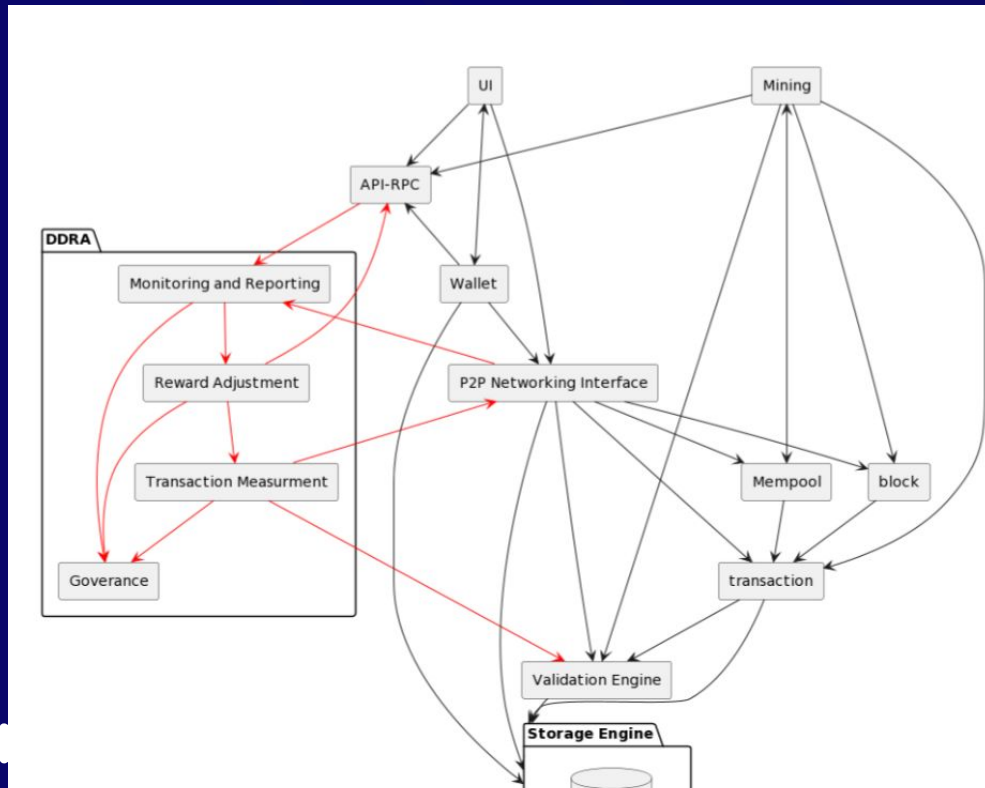
The Transaction measurement subsystem must manage the transactions transmitted via the P2P Network Interface.

### P2P Network Interface -> Monitoring and Reporting:

For propagation of important information to the rest of the network.

### API/RPC -> Monitoring and Reporting:

API/RPC holds the getBlockTemplate rpc used in mining, but gets the value of the coinbase transaction from the Monitoring and Reporting subsystem.





# Impacted Files and Directories



- mining.cpp in the rpc folder:
  - the getBlockTemplate rpc is found here, which will need to be modified to account for changing the value of a coinbase transaction
- policy.h file within the policy folder in the validation engine:
  - To account for the fact that the Transaction Measurement component needs to verify that transactions are valid. Hence a method for efficiently verifying this should be added
- Consensus folder within the validation engine:
  - Need for a defined formula for calculating reward change based on transaction rate



# Important Risks and Limitations

## Latency

### Limitation:

- Achieving a low latency system can be resource-intensive, which can increase the cost of operating and maintaining the system.

### Risk:

- Real-time adjustments between clients must be made with minimal wasted communication to ensure consensus
  - If the system is unable to adjust block rewards quickly enough, miners may leave the network
  - If adjustments made too quickly could lead to inconsistencies in the network.

## Robustness

### Limitation:

- The system's decision-making is based on predetermined rules and algorithms, which may not be adaptable to unforeseen circumstances.

### Risk:

- The system may not be as robust to adverse conditions without human intervention, and updates to the algorithm should be kept to a minimum.
  - This could limit the system's ability to respond to changing market conditions or emerging threats.



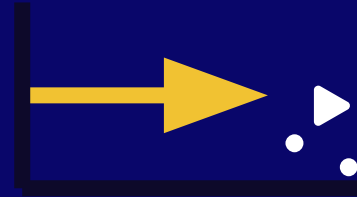
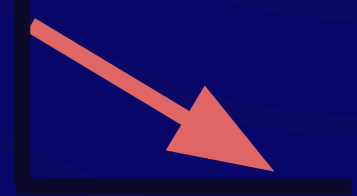
07.

# Plans for Testing

# Plans for Testing



- Transaction Shortage: Decrease transaction rate 90%
  - Test 1: Does transaction measurement record this drop?
  - Test 2: Has getblocktemplate been changed?
  - Test 3: Are increased rewards being propagated?
- Transaction Storm: Increase transaction rate 90%
  - Do Test 1-3 from shortage
  - Determine an equilibrium point between transaction fee and deflation
- Idle System: Record + simulate real word behaviour
  - Test how disruptive the enhancement is to the system





08.

# Concurrencies



# Concurrencies



- The adjustment system must operate on a p2p decentralized network, thereby, must work independently of other hosts to reach consensus.
- Client concurrency is difficult to achieve as there is a sequential method of calculating reward based on each transaction received, and a lockstep is essential for reliable tracking.
- That being said, concurrency is present between mining and calculating reward, as a constructed block will no longer need lockstep updates. The need for concurrency is limited however, as calculations are very fast.





09.

# Currency Adjustment

Use Case Analysis





# Currency Adjustment

This system is triggered by out-of-date data, or by the adjustment detection use case.

## API-RPC Check:

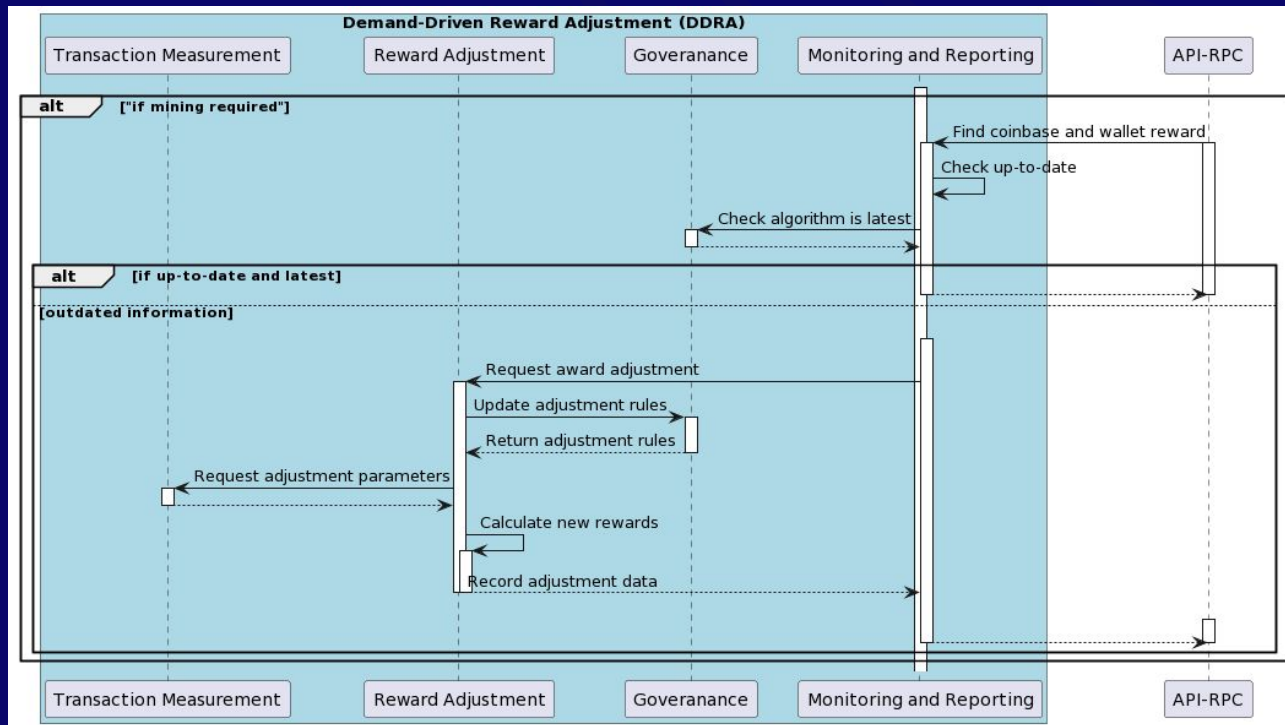
Checks for updates to ensure accurate information. The monitoring and reporting module seeks consensus.

## Generate New Values:

Monitoring and reporting calls the reward adjustment to generate new values. Requests parameters from the transaction measurement module, calculates reward.

## Propagation:

The monitoring and reporting module supplies updated information to API-RPC





10.

# Limitations and Lessons Learnt



# Lessons and Limitations

## Lessons

### Breakdown of Concepts:

Breaking down complicated variables was harder than imagined.

### Interdisciplinary Knowledge:

Discovered the value of interdisciplinary skills amongst computer scientists.

## Limitations

### Knowledge:

Lacking knowledge of fields like economics and finance.

### Theoretical Impact:

Real testing is difficult as the behavior of the network depends on the nodes that are currently active on the network.



11.

Conclusion





# Thank you

Any Questions?

Amy Cui (19ayc1@queensu.ca)  
David Curtis (20dhc@queensu.ca)  
Jagrit Rai (19jr28@queensu.ca)  
John Alajaji (18ja19@queensu.ca)  
Logan Cantin (logan.cantin@queensu.ca)  
Matthew Vandergrift (19mwv1@queensu.ca)