# A Blockchain-based Reputation System for Small Satellite Relay Networks

**Lillian Clark, Yeh-Ching Tung, Matthew Clark, Laurence Zapanta**
**The Aerospace Corporation**
**2310 E. El Segundo Blvd.**
**El Segundo, CA 90245**
**lilliamc@usc.edu, yeh-ching.tung@aero.org, matthew.a.clark@aero.org, laurence.f.zapanta@aero.org**

*Abstract*— **Currently, space communications networks are attempting to move away from large geostationary (GEO) satellites towards large constellations of small satellites. This trend is observed both in government and commercial communications satellites. By relaying data across multiple constellations/networks, we may be able to reduce end-to-end latencies and reduce burden (mass, power, cost) for all users. However, this is only possible if networks across constellations can establish inter-satellite authentication and trust. The key to this trust is based on demonstrated ability of the relay satellites to meet performance requirements, i.e. "reputation." In this work, we propose leveraging distributed ledger technologies (i.e. blockchains) to develop a secure, decentralized reputation system for satellite relay networks. This informs a reputation-aware routing protocol and reduces the average data latency across the network. In this paper, we discuss designing the blockchain-based reputation system and routing protocol. We then analyze the resultant network performance with respect to average latency, computational complexity, and storage considerations for a variety of use cases.**

## 1. Introduction

Advancements in aerospace and computer engineering combined with the growing paradigm of secondary payloads in commercial launch vehicles have allowed for miniaturization in space systems. Communications satellites, for example, can now be as small as a cereal box (e.g. CubeSats) instead of the size of a schoolbus, the previous norm for GEO communication satellites. However, small satellites are still less powerful than their larger counterparts in terms of data collection, processing, and transmission power.

Because of these limitations, multi-satellite systems have grown in popularity to compensate for the limited capabilities of a single small satellite. Multi-satellite systems provide additional advantages in that they are more robust to the loss of a single asset. It is for these reasons that tech giants like OneWeb, SpaceX, Facebook, and Amazon [1] are interested in large constellations of communications satellites in Low Earth Orbit (LEO).

Global access or persistence over a target area, two goals of the proposed LEO constellations, will likely require inter-satellite communication links. This trend toward cross links provides advantages and disadvantages. The increasing number of players lends itself to a paradigm of spacecraft operations where cooperation across agencies is desirable. This in turn could decrease delays, minimize energy consumption, and encourage satellite specialization. However, increased cooperation is dependent on ongoing evaluations of authentication and reputation. Reputation is a measure of how reliably a satellite meets performance requirements, taken as an average of the achieved performance with respect to latency, throughput, capacity, or other metrics.

Incorporating reputation can improve network resiliency to various threats including selfish nodes, blackholes, and minor anomalies. Selfish nodes are nodes in a network which malevolently direct traffic away from themselves in order to save resources. Blackholes are nodes in a network which malevolently direct traffic towards themselves in order to eavesdrop on, redirect, or drop messages. Both of these threats falsify the state of their (or their neighbors') links, which is preventable via a reputation system. Further, if a degradation in performance that is not caught by onboard systems were to occur, a reputation system would enable detection of the anomaly. The secure and distributed qualities of our reputation system ensure that this ledger is only changed by trusted network nodes and will not be lost if any network node is compromised.

In this work we address the challenges of network performance and resiliency via two novel contributions. The first is a secure ledger of reputation data which is tolerant to delays and partitions, and resilient to adversaries. The second is a reputation-aware routing protocol informed by this ledger which improves overall network performance.

## 2. Background

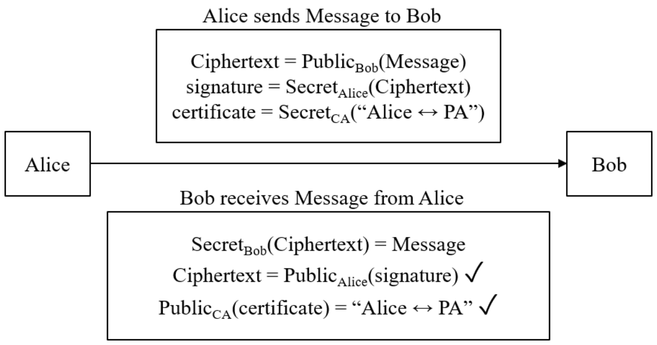
Our work leverages distributed ledger technologies, or blockchains, to securely maintain and update a ledger of reputation data about the past performance of satellites and ground stations. In this section we review the salient aspects of blockchain and authentication.

A blockchain is a distributed, immutable hash-linked ledger of data. A "block" is composed of three things: a timestamp, a cryptographic hash of the data it stores, and the hash of the previous block. The "chain" refers to the collection of linked blocks. Blockchain technologies have several advantages and disadvantages.

### *Key Advantages of blockchains*

1. *Verifiable.* All participants in a blockchain network can verify recorded data via the cryptographic hash, and previous data cannot be altered.
2. *Agreed Upon.* A consensus mechanism governs the addition of new data.
3. *Consistent.* A peer-to-peer protocol ensures the ledger is consistent across the network.
4. *Decentralized.* With the right permissions, any node in the network can record data to the chain.

978-1-7281-2734-7/20/$31.00 ©2020 IEEE



Alice sends Message to Bob
Ciphertext = Public$_{Bob}$(Message)
signature = Secret$_{Alice}$(Ciphertext)
certificate = Secret$_{CA}$("Alice ↔ PA")
Alice
Bob
Bob receives Message from Alice
Secret$_{Bob}$(Ciphertext) = Message
Ciphertext = Public$_{Alice}$(signature) ✓
Public$_{CA}$(certificate) = "Alice ↔ PA" ✓

**Figure 1**. PKI Authentication

### *Key Disadvantages of blockchains*

1. *Computationally intensive.* Many blockchain technologies rely on a Proof-of-Work algorithm to enforce consensus, which means CPU (central processing unit) power is equivalent to voting power [2]. Even with other, less computationally intensive consensus mechanisms like Proof-of-Stake [3], the burden of verification and consensus is on each individual node. This is due to the lack of centralization. For small satellites, this is a concern because they are extremely memory and power limited.
2. *Relies on connectivity.* In terrestrial settings, the consistency of the blockchain is provided by the internet. Essentially all network nodes in the blockchain are connected at all times. However networks designed to be delay-tolerant do not have this persistency. Managing the ledger and ensuring consistency across network nodes in space requires additional consideration.

In this work we have designed and simulated a "permissioned" blockchain. Unlike Bitcoin which allows anyone with an internet connection to create and add blocks to the chain, permissioned blockchains leverage cryptographic algorithms to ensure only trusted network nodes can access the blockchain.

### *Authentication*

Because our distributed ledger technology is permissioned, only authenticated satellites and ground stations have access to the blockchain. This is guaranteed via the same Public Key Infrastructure (PKI) used for encryption/decryption, message verification, and authentication in state of the art satellite communications. PKI relies on the use of public and private/secret keys and a trusted Certificate Authority (CA).

An example is shown in Figure 1. Satellite A (Alice), Satellite B (Bob), and the CA each have a public key and a secret key. Alice can use Bob's public key to create a ciphertext or encrypted message which only Bob can decrypt via his secret key. Alice can use her secret key to sign the ciphertext so that Bob can verify the message came from her public key. Alice can request a certificate which associates her identity with her public key. This certificate is a message stating the association signed by the secret key of the certificate authority. Bob, along with every node in the network, has the certificate authority's public key and can confirm that Alice has a valid certificate.

When writing blocks to the blockchain, nodes include their certificates and sign all messages. This authentication method means all blocks are verifiable by all nodes, and untrusted nodes cannot contribute to the blockchain.

## 3. Related Work

### *Satellite Network Routing*

There is a large body of work on routing protocols in satellite networks, discussed in [4], [5], and the references therein. Routing protocols fall into one of two categories: proactive and reactive. In proactive schemes, each satellite stores the network topology onboard, and when a satellite needs to send data it finds a route and establishes a connection. As the network scales and the onboard storage shrinks, it can become difficult to maintain these routing tables. In reactive schemes, satellites try to find an optimal path only when they have data to send. Metrics for determining the optimal path might include delay, bandwidth, load, or hop count depending on the routing protocol implementation. In dynamic networks, a common approach is to consider a "topology slice," or the state of the network at a specific instance, and treat the slice as static. In some hybrid routing protocols, the network is divided into clusters, and a mix of proactive and reactive routing is used. For space networks specifically, which are characterized by long delays and intermittent connectivity, routing protocols generally have two additional characteristics. The first is that data is stored in bundles and a store-and-forward strategy ensures data reaches its destination in the event that connectivity has temporarily lapsed. The second is the maintenance of "contact graphs," which predict opportunities for connectivity based on the satellite orbital elements. The routing protocol this work builds on is a link state proactive routing protocol.

### *Context-Aware Routing*

Previous research has demonstrated that in a network with dynamic link states and a changing topology, incorporating context into routing protocols can be shown to prevent blackhole attacks [6], in which nodes malevolently redirect traffic to themselves. Context can refer to any information describing the state of the communication links, including bandwidth, capacity, delay, etc. Our work draws on previous work integrating data on past performance into a routing protocol. However, previously reputation has been based on probabilistic estimates or calculated locally. Our reputation-aware routing protocol is suited to handle a full ledger of globally agreed upon data.

### *Blockchain-based Reputation Systems*

While it was originally popularized for cryptocurrencies, the advantages of distributed ledger technologies have made blockchains appeal to many different domains [7]. Blockchain has been considered as a viable solution for applications in which a centralized database can be replaced with a distributed one, and information accumulates rather than being changed and replaced. It has been considered for reputation systems in education/academia [8] and peer-to-peer file sharing [9]. Blockchain has been shown to effectively store reputation data because the decentralized nature removes the risk of bias in the centralized server, makes the system more fault-tolerant (rather than one point of failure), and doesn't allow for reputation information to be falsified or tampered with. Satellite networks have not previously been considered as an application for reputation systems, but the changes in the operational environment of space have opened this as a research direction.

### *Blockchains in Space*

Blockchain has previously been considered in applications relevant to the space industry, namely manufacturing provenance [10], incentive schemes like in vehicular networks

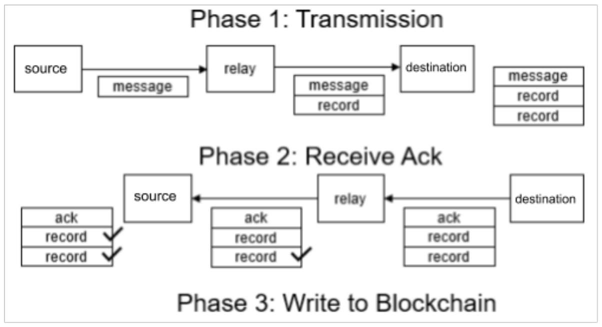


**Figure 2**. Block Creation

[11], computing in resource constrained networks [12], and command and telemetry logging in multi-satellite architectures [13]. While this last application is very relevant, the blockchain proposed in [13] is used for logging actual data transmitted across the network. New research challenges exist in using blockchains in a space network as a means to record and incorporate reputation. To our knowledge, this work is the first to investigate a blockchain-based reputation system in satellite networks.

## 4. System Model

The satellite network we consider in this work has the following concept of operations: messages are generated randomly at source nodes and need to reach destination nodes before a specified amount of time has elapsed. A source may transmit to a destination directly, or may use one or more relay nodes. We use the terminology “source” and “destination” to refer to the end-to-end link, and “sender” and “receiver” to refer to a single hop. Each block in the blockchain corresponds to one message.

### *Reputation Blockchain*

*Block Creation*—In our blockchain, a new block is created through three phases (Figure 2). Phase 1 of the reputation scheme is entered when the source has a message to transmit. It looks up the next hop in its routing table and sends the message and an empty record field to the next hop. At the next node, an entry is added to the record field which contains information about that specific data pass. This information includes the sender address, receiver address, and latency measured at the receiver. The receiver also includes its signature, so any node can verify that this information was not tampered with. In our implementation, we are concerned with the achieved latency, but any number of parameters could be recorded including bit error rate, data rate, signal to noise ratio, capacity, link utility, etc. If the node is not the destination for this message, it looks for the next hop in its routing table and relays the message to the next hop. At each relay node, the record is extended by one entry. When the message gets to the destination, the reputation scheme advances to phase 2 and an acknowledgement message (ACK) is created. This ACK serves three purposes: (1) it confirms to the source that the message was received, (2) it reports whether end-to-end requirements were met, and (3) it returns the record field to the source. This ACK returns along the reverse of the message transmission path. At each hop along the path, each node that was previously a sender has the opportunity to confirm or deny the reputation information that was recorded about its pass by the receiver. This incentivizes nodes to report information honestly. This will be expanded upon in the discussion of the reputation-aware routing protocol.

In the event that the satellite dynamics are such that the network topology changes at a rate where the original message path may not be available by the time the acknowledgment is transmitted, we present two alternatives. The first is to create a block each time a message is forwarded (rather than sent). This block-per-hop strategy means that senders can still confirm the reputation recorded by the receiver, regardless of whether the message path persists. However this solution has two disadvantages. One is that blocks no longer contain information on the end-to-end link quality, which is an important indicator of network performance. The second is that the increased rate of block creation implies more communication and computation burden, putting additional stress on the satellites.

The second alternative is to maintain the block-per-message protocol but allow the confirmation process to be more dynamic. When a node has an ACK to send, it floods the ACK by sending a copy to all of its neighbors, who then, if they have not already received this ACK, send the ACK to each of their neighbors, and so on. If ACKs are flooded in this manner, rather than returning along the reverse of the message transmission path, all nodes can process each ACK they receive and confirm or deny any reputation records which concern themselves. After a specified timeout, the source would need to compile the ACKs it received which refer to the message in question, and aggregate the confirmations. This process would put more burden on the network to flood and process ACKs, and more burden on the sender to aggregate confirmations before writing a block to the chain.

When the acknowledgement is received at the source, the reputation system enters phase 3, writing the block. The block contains a timestamp, the hashed record field and indicator of whether the message met end-to-end requirements, and a pointer to the most recent block at this node, which links it to the chain. Because this is a permission blockchain, blocks also contain a signature and a valid certificate indicating that the author was authenticated by the certificate authority. The process of confirming reputation information as it is returned via the acknowledgement is our distributed ledger technology’s consensus mechanism. It ensures that nodes are penalized for falsifying reputation data.

*Blockchain Consistency*— Once a block is added to the blockchain at a node, the blockchain is flooded to the network via peer-to-peer communications. The node who authored the block forwards the block to its neighbors. At each neighbor, if the block is new it is verified, added to the local copy of the blockchain, and similarly flooded to its neighbors.

We also consider the case where the blockchain data is not flooded. In this case, it is no longer a blockchain, but a localized ledger containing only information on passes for which the owner was involved. We analyze this system as well, to compare the affect of local and global reputation information, and consider the importance of consistency in the blockchain.

*Block Verification*—While the record data is verified when a message or ACK is received, the full chain is also verified when it is received by a satellite during the flooding process. Verification ensures that every block originated at an authenticated source and that every block has a parent. Each block

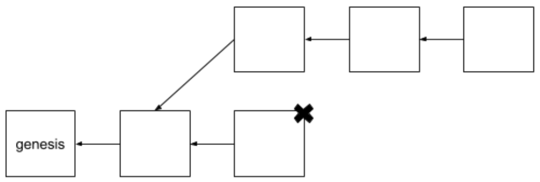


**Figure 3**. Handling Partitions: Standard Blockchain

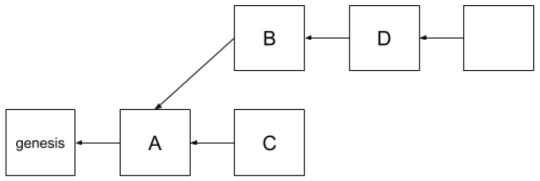


**Figure 4**. Handling Partitions: Partition-tolerant Blockchain

needs a parent block to ensure data is not tampered with. If reputation data is altered, the hash value of the child block will no longer point to the parent, and it will become obvious that the chain has been compromised.

*Handling Partitions and Delays*—For most applications, it is necessary for blocks to be limited to one parent and one child. This ensures that the chain is in fact a chain, rather than a tree. In most blockchain technologies, e.g. Bitcoin, Ethereum, and Hyperledger, this is implemented via a simple rule, shown in Figure 3. If a branch is detected, the longer strain is kept.

The reason the ledger can not be a tree depends on the application. In cryptocurrencies, for example, a branch would indicate that some amount was spent twice. This is a clear conflict. However in our scenario, in which we can expect partitions and delays in the network, trees can be formed innocuously. For example (Figure 4), imagine satellite A authors a block and floods it to satellites B and C, and then the link between A and C is lost such that no connection exists between B and C. Now satellite B authors a block and satellite C authors a block independently. Each of these blocks have the block authored by A as a parent, and yet they are not conflicting blocks. If satellite D comes into view of B and C and receives both blocks, it should treat them both as valid. For this reason our blockchain implementation draws from the work of [14] in the handling of blocks which share a parent. Essentially we replace the notion of a chain with a Directed Acyclic Graph (DAG). Blocks which share a parent are both accepted and stored, and new blocks have the most recently stored block as a parent.

Although sharing a parent is not an indicator that two blocks conflict, it is possible to have conflicting blocks. Each block corresponds to a specific message which has a source, destination, and timestamp. If two blocks appear to refer to the same message but have conflicting reputation data, the record kept in each of these blocks is flagged as conflicting. The satellite who authored these blocks is penalized, which disincentivizes authoring multiple blocks per message, which could cause traffic redirecting or other malevolent behavior through inaccurately weighting this message during reputation calculation. Since only the source of a message can author a block corresponding to that message, reputation records contain signatures and cannot be falsified, and blocks cannot be duplicated or tampered with, the blockchain reputation system is secure.

*Reputation-Aware Routing Protocol*

The reputation blockchain proposed in the previous section could augment any choice of routing protocol. For simplicity, we assume a routing protocol which uses Dijkstra's shortest-path algorithm. Considering the network topology as a graph of vertices and edges, where vertices are satellites or ground stations, and edges are closed links, Dijkstra's algorithm will find the shortest path from a single vertex to any other vertex in the graph. Edge-weights in our implementation correspond to propagation delay between nodes, and are in units of time. Propagation delay is proportional to distance and for simplicity we consider other factors such as link quality to be consistent across the network. Therefore the edge-weights can analogously be thought of as distance. Here, we assume that satellites and ground stations can query their neighbors' routing information instantaneously. Realistically, this data would be sent periodically via a neighborhood discovery protocol. RFC 6130 [15] is an example of one commonly used neighborhood discovery protocol in mobile ad hoc networks, and would be appropriate for this scenario. We make the simplifying assumption that time *t* between "hello" messages approaches zero. For networks in which node mobility is either slow or predictable, topologies change slowly relative to the time messages are sent and this assumption seems reasonable.

Each satellite updates its routing table according to Dijkstra's algorithm at a period $t_{route}$. We use this shortest-path routing protocol as a baseline for comparison to evaluate our reputation-aware routing protocol.

The information stored in the reputation blockchain is used in two ways: link reputations, and node reputations. The link reputation refers to a specific link between Satellite A and Satellite B, for example. As implemented, it captures the average achieved latency of this link. As mentioned previously, many other link parameters could similarly be measured. The average achieved latency is calculated over the previous $b$ blocks. This blockchain lookback parameter allows the protocol designer to trade between processing time and information. A small lookback parameter means the reputation information may never reach a steady state average. On the other hand, a large lookback parameter requires storing and processing more blocks, putting additional burden on the satellite data-handling subsystem.

This link reputation information, if present for a link, replaces the edge-weight associated with this link during routing. This allows the routing table to reflect past performance, rather than the link state advertised during neighbor discovery. Since adversarial nodes could propagate false link state information about their neighbors during neighbor discovery, this link state information could be compromised. In contrast, the link reputation information stored in the blockchain cannot be tampered with and also illustrates a more comprehensive picture of past performance.

Node reputation refers to a specific satellite or ground station. Nodes are penalized for participating in a pass that results in an unconfirmed record. When updating the routing table, nodes will not route through satellites with a node reputation above the threshold $\eta$, where $\eta$ is typically between 1-10. Each participation in an unconfirmed record increments node reputation. The reputation table updates with period $t_{rep}$.

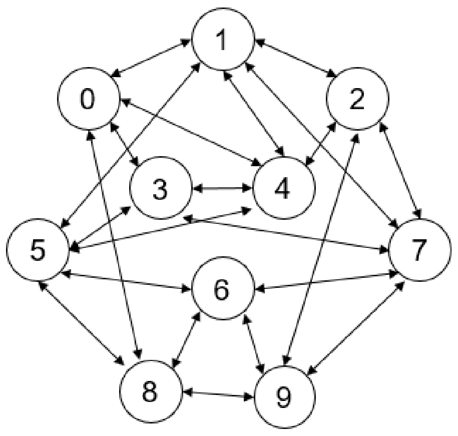


**Figure 5**. Sample Network (edge-weights not shown)

| Time | Source | Destination | Requirement |
|---|---|---|---|
| 0 | 1 | 9 | 2 |
| 4 | 3 | 6 | 6 |
| 7 | 6 | 0 | 10 |
| 12 | 2 | 4 | 14 |
| 13 | 2 | 8 | 16 |
| 20 | 6 | 7 | 21 |
| 25 | 0 | 9 | 27 |

**Figure 6**. Sample Schedule

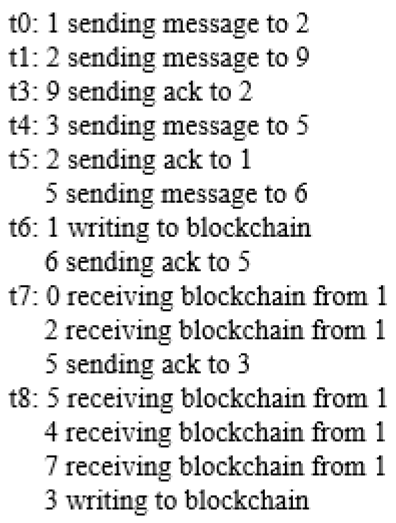
## 5. Simulation and Experiment Design

In order to analyze the performance of our reputation-aware routing protocol, we designed a discrete event network simulator. Written in Python, the simulator allows parallelization of simulated satellites via threads. The blockchain-reputation scheme and reputation-aware routing protocols are each integrated in the network simulator.

The simulation begins with the definition of a network. A network, like the one shown in Figure 5, is defined by its nodes, edges, and edge-weights. A node represents a satellite or base station and is characterized by its average induced latency. Edges represent closed links, and edge-weights represent advertised expected latency. For simplicity, the following experiments assume links represent persistent connections over the simulation time. Messages are of fixed length. Message transmissions can be scheduled before or during the simulation. In the following experiments messages have a randomly selected source and destination, and transmission start times are drawn randomly from a uniform distribution over the simulation time. An example schedule is shown in Figure 6. The "Requirement" field represents the time at which the message must be received at the destination in order for the transmission to be considered a success.

The parameters which are considered during an experiment include

- Routing: shortest-path or reputation-aware
- Reputation information: local or global
- Blockchain lookback, $b$
- Reputation update period, $t_{rep}$
- Routing update period, $t_{route}$



```
t0: 1 sending message to 2
t1: 2 sending message to 9
t3: 9 sending ack to 2
t4: 3 sending message to 5
t5: 2 sending ack to 1
    5 sending message to 6
t6: 1 writing to blockchain
    6 sending ack to 5
t7: 0 receiving blockchain from 1
    2 receiving blockchain from 1
    5 sending ack to 3
t8: 5 receiving blockchain from 1
    4 receiving blockchain from 1
    7 receiving blockchain from 1
    3 writing to blockchain
```

**Figure 7**. Sample Simulation Log Output

The output of an experiment includes a written log of the events which transpired. The first 15 lines of the simulation log output for the network and schedule corresponding to Figures 5 and 6 are shown in Figure 7. Various analytics are also collected, including the percentage of messages which met the specified requirements, the average number of hops per message, the average latency per hop, and the average latency per message. The network simulator is fully customizable, and could be extended to consider additional performance metrics.

## 6. Results

The plots in Figure 8 show the performance results of three sample networks. The first was made up of 30 satellites all of which behaved nominally. The second was made up of 10 satellites, 2 of which behaved adversarially, and the third was made up of 30 satellites, 5 of which behaved adversarially. The behavior of adversarial nodes is dictated by an additional parameter, defect probability $p_{defect}$. At each opportunity to transmit, adversarial nodes drop the message with probability $p_{defect}$ or double their induced latency with probability $p_{defect}$. At each opportunity to confirm reputation data, adversarial nodes falsify reputation data with probability $p_{defect}$.

The numbers shown in Figure 8 are the average performance over seven instantiations of network topology. Each instantiation is generated according to the procedure outlined in Algorithm 1, with the parameter assignments tabulated in Table 1. From left to right, the performance results in Figure 8 show an increase in available information. "Shortest-path" refers to the un-augmented Dijkstra's routing protocol. "Local, b=30" refers to the reputation-aware routing protocol with local reputation information and a blockchain lookback of 30, whereas "local, b=M" has a blockchain lookback of 200, the length of the message history. The next two series follow the same pattern but for global reputation information.

The trend in these plots show that with more information, the percentage of messages which are successful increases and the latency per hop decreases. This is because the routing protocol learns the actual performance and reliability of the network, rather than the advertised performance of each link.

**Algorithm 1** Initialize Network

```
Number of satellites = $N$
Number of messages = $M$
Degree = $d$
Advertised latency distribution = $l_a$
Additional induced latency distribution = $l_i$
Simulation time = $T$
1:  for satellite index=1:$N$ do
2:      Create a satellite with additional induced latency drawn from $l_i$
3:      Add satellite to the network
4:  end for
5:  for each satellite in the network, $i$ do
6:      for each satellite $j$ of $d$ randomly selected satellites do
7:          if edge $i \leftrightarrow j$ doesn't exist then
8:              create edge $i \leftrightarrow j$ with advertised latency drawn from $l_a$
9:          end if
10:     end for
11: end for
12: for message index=1:$M$ do
13:     create a fixed size message
14:     schedule the message at time $t$ uniformly selected in the range (0, $T$)
15: end for
16: return Network
```

**Table 1**. Network Parameter Values

| Parameter | Value |
|---|---|
| $M$ | 200 |
| $d$ | 5 |
| $l_a$ | $\begin{cases} 0.5 & \text{w.p.} \frac{1}{3} \\ 1.5 & \text{w.p.} \frac{2}{3} \end{cases}$ |
| $l_i$ | $\begin{cases} 1.5 & \text{w.p.} \frac{1}{3} \\ 3.5 & \text{w.p.} \frac{2}{3} \end{cases}$ |
| $T$ | 100 |
| $p_{defect}$ | $\frac{1}{5}$ |

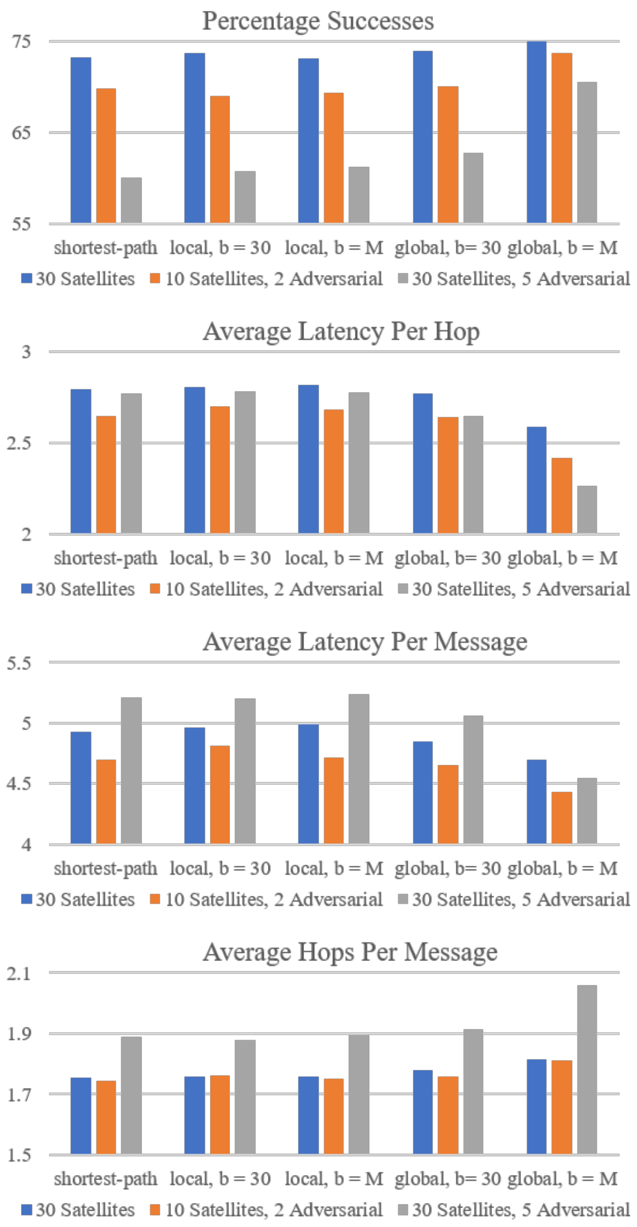


**Figure 8**. Performance Results

By increasing the average hops per message, i.e. relying more on relay satellites, the reputation-aware routing protocol can actually decrease the average latency per message. This supports our position that the reputation system can improve network performance.

Compared to the 5% decrease in average latency per message achieved by the reputation system with $b = M$ in the 30 satellite network, the two experimental setups with adversarial network nodes saw even greater improvements. The 10 satellite, 2 adversarial network saw a 7.3% decrease in latency per message. The 30 satellite, 5 adversarial network saw a 12.5% decrease in latency per message. This demonstrates that our reputation system performs well in the presence of adversarial nodes, and is actually best-suited for this use case.

*Scalability*

To consider how performance of our blockchain-based system would scale, we conducted a similar experiment for a larger network ($N = 500, M = 200, T = 220$). The plots in Figure 9 show the performance of the last 100 messages transmitted over the network. This setting saw a 3.2% decrease in average latency per message in global setting with $b = M$. For this large network, the local reputation and small blockchain lookback experimental setups did not provide a significant improvement in either latency or percentage successes.

*Resiliency*

In addition to considering how the network performs in the presence of adversarial nodes, we also consider resiliency, or how easily the network can bounce back from the introduction of adversaries. Figure 10 show the average latency per message and percentage of successful messages for a network of size 15 which is exposed to 3 new adversaries at t=100. These plots show performance over time.

After the adversaries are introduced, the only experimental setup which does not suffer a performance decrease is the global, $b = M$ reputation system. This demonstrates that the blockchain-based reputation system learns to distrust adversarial nodes, resulting in overall network resiliency.

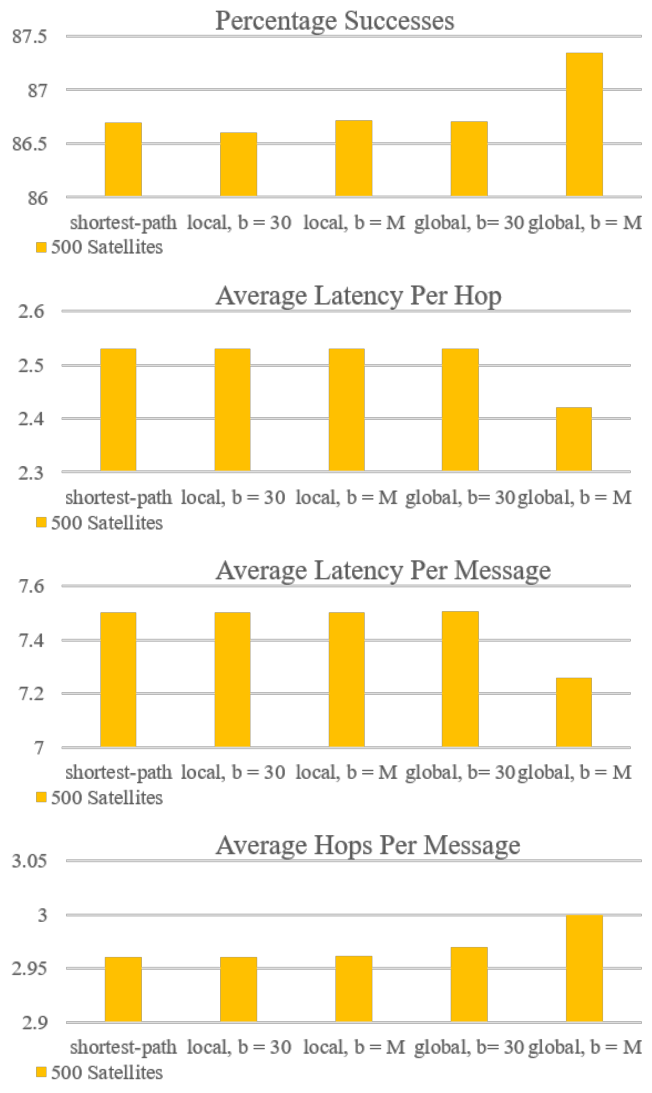


**Figure 9**. Performance of Scaled-up Network

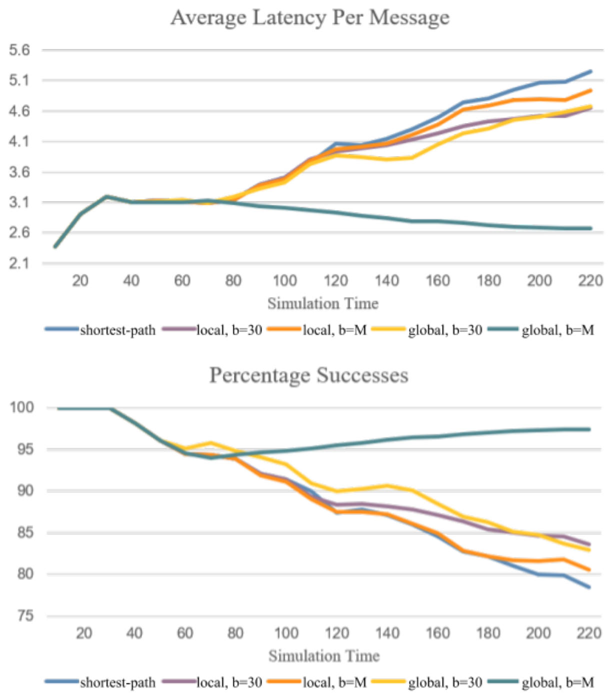


**Figure 10**. Performance of Network Exposed to Adversaries

## 7. Practical Considerations

Although it provides the demonstrated advantages, the introduction of blockchain and reputation-aware routing adds some complexity and additional burden. For example, there is the complexity associated with updating the reputation and routing tables. However, routing tables are nominally updated periodically in dynamic networks and the additional complexity of updating the reputation table wouldn't significantly change this cost.

There is also the computational burden of processing the blockchain. In our implementation, this algorithmic complexity is bounded by $O(b(N-1))$ where $b$ is the blockchain lookback parameter and $N$ is the number of network nodes. Updating the routing table is bounded by $O(N^2)$. Given the ability to control the blockchain lookback parameter, this computational cost could be adjusted by the satellite operator to not exceed the hardware limitations.

One of the often-cited disadvantages of blockchain is the storage burden. For our ledger, each block consists of, at minimum, the hashed addresses of the source and destination, the reputation information, and a pointer to the previous block. Assuming, for instance, SHA256 cryptographic hash functions, a single block header containing the two addresses and pointer would be on the order of 96 bytes. Assuming each reputation record contains the sender, receiver, and 128 bits of data, each record would be on the order of 80 bytes. In the worst-case scenario a message is passed through every node in the network, and block sizes are therefore bounded by $96 + 80(N-1)$ bytes. Given that even CubeSat processors have at least 4 Gbit SDRAM [16], a blockchain lookback value of 1,000 and a network size of 100 would only use 1.6% of the working memory. A network with 1,000 nodes could store a blockchain of length 6,248 on a single cubesat processor.

There is some additional strain on the communication system and its energy consumption. An additional message broadcasting the blockchain must be sent for each data message, potentially limiting the link utility for telemetry, data, and command messages. However, DTN bundles on the order of 10-100kBytes have been shown suitable for space networks [17]. For a network with blockchain lookback 100 and an average of 3 hops per message, a blockchain transmission would be on the order of 30kBytes of data, which is within the size range for a single bundle. As the blockchain contains useful information on the state of all links, this broadcast could replace standard neighbor discovery messages. A "hello" message sent during neighbor discovery is on the order of 45 bytes and sent every 2 seconds [15], so a blockchain of 100 blocks sent every 20 minutes would take equivalent transmission power. The strain on the communication system is highly dependent on the specific implementation, hardware, and network size, and should be investigated further.

## 8. Conclusions and Future Work

In this work we have considered the design, implementation, and analysis of a novel blockchain-based reputation system and reputation-aware routing protocol to improve performance and resiliency in small satellite relay networks. We have determined that augmenting routing with reputation information can enhance network performance and resiliency to adversaries. This reputation can be stored via a blockchain, or distributed ledger technology, with some alterations to the standard consensus and consistency mechanisms.

We have also considered the trade-off in terms of complexity, which should be studied further for specific networks and protocol implementations, as this analysis depends on network topology and selected hardware. Additional further work could include analysis of dynamic networks, additional threat models, and extension of the blockchain to be integrated with neighbor discovery or anomaly prediction.

## References

[1] J. Shieber, "Amazon joins spacex, oneweb and facebook in the race to create space-based internet services," 2019.

[2] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] F. Saleh, "Blockchain without waste: Proof-of-stake," *Available at SSRN 3183935*, 2019.

[4] R. Radhakrishnan, W. W. Edmonson, F. Afghah, R. M. Rodriguez-Osorio, F. Pinto, and S. C. Burleigh, "Survey of inter-satellite communication for small satellite systems: Physical layer to network layer view," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2442–2473, 2016.

[5] X. Qi, J. Ma, D. Wu, L. Liu, and S. Hu, "A survey of routing techniques for satellite networks," *Journal of communications and information networks*, vol. 1, no. 4, pp. 66–85, 2016.

[6] G. Dini and A. L. Duca, "Towards a reputation-based routing protocol to contrast blackholes in a delay tolerant network," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1167–1178, 2012.

[7] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[8] M. Sharples and J. Domingue, "The blockchain and kudos: A distributed system for educational record, reputation and reward," in *European Conference on Technology Enhanced Learning*. Springer, 2016, pp. 490–496.

[9] R. Dennis and G. Owen, "Rep on the block: A next generation reputation system based on the blockchain," in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*. IEEE, 2015, pp. 131–138.

[10] S. A. Abeyratne and R. P. Monfared, "Blockchain ready manufacturing supply chain using distributed ledger," 2016.

[11] L. Li, J. Liu, L. Cheng, S. Qiu, W. Wang, X. Zhang, and Z. Zhang, "Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 7, pp. 2204–2220, 2018.

[12] G. S. Ramachandran and B. Krishnamachari, "Blockchain for the iot: Opportunities and challenges," *arXiv preprint arXiv:1805.02818*, 2018.

[13] R. Mital, J. de La Beaujardiere, R. Mital, M. Cole, and C. Norton, "Blockchain application within a multi-sensor satellite architecture," 2018.

[14] J. A. Tran, G. S. Ramachandran, P. M. Shah, C. B. Danilov, R. A. Santiago, and B. Krishnamachari, "Swarmdag: A partition tolerant distributed ledger protocol for swarm robotics," *Ledger*, vol. 4, 2019.

[15] T. H. Clausen, J. W. Dean, and C. Dearlove, "Mobile ad hoc network (manet) neighborhood discovery protocol (nhdp)," 2011. [Online]. Available: https://tools.ietf.org/html/rfc6130

[16] SpaceMicro, "Cubesat space processor (csp)," September. [Online]. Available: spacemicro.com/assets/datasheets/digital/slices/CSP.pdf

[17] C. V. Samaras and V. Tsaoussidis, "Adjusting transport segmentation policy of dtn bundle protocol under synergy with lower layers," *Journal of Systems and Software*, vol. 84, no. 2, pp. 226–237, 2011.