AODV-Miner: Consensus-Based Routing Using Node Reputation

Edward Staddon, Valeria Loscri and Nathalie Mitton
Inria, France - {firstname.lastname}@inria.fr

***Abstract*—With the increase of Internet of Things (IoT) applications, securing their communications is an important task. In multi-hop wireless networks, nodes must unconditionally trust their neighbours when performing routing activities. However, this is often their downfall as malicious nodes can infiltrate the network and cause disruptions during routing. We grant nodes the ability to evaluate the behaviour of their neighbours and, through consensus inspired from blockchain's miners, agree on the credibility of each node. The resulting metric is expressed as a node's reputation allowing, in the case of a malicious node, to isolate it from network operations. By illustrating this in an AODV-like multi-hop routing protocol, we can influence route selection no longer based solely upon the shortest number of hops, but also the highest overall reputation. Simulation results revealed that our approach can decrease packet drop rates by $\approx 48\%$ in a static context when subjected to multiple black hole attacks compared to the original routing protocol.**

***Index Terms*—IoT, Reputation, Cyber Security, AODV, Blockchain**

I. Introduction

The Internet of Things (IoT) is becoming more prominent in every day life, processing increasing amounts of sensitive data. Indeed, some applications come with extreme risks which could result in severe consequences, from data breach to loss of life. However, in some cases, devices are deployed in sparse hostile environments, forcing them to employ other network paradigms to communicate. This is the case of multi-hop networks, where data is forwarded by intermediate devices to reach its destination. Therefore, securing this exchange is paramount as entrusting data to unknown nodes is a huge risk.

Many solutions combating threats in IoT networks have been proposed in the literature. Such approaches as trust based [1], grant the capability to identify different nodes, based on their previous actions in the network. This allows routing protocols to adapt to the situation, granting routing capabilities only to the most trustworthy [2], be it based on cooperation [3] or through the use of signatures [4] to identify potential threats. Furthermore, by analysing node actions and assigning them a trust value, it is possible to influence how routing is performed [5]. By using the notion of reputation, inspired from the human psyche, we can evaluate the actions of surrounding nodes, allowing protocols to avoid attacks [6]. However, many approaches have been designed as part of existing protocols and, therefore, only work in conjunction with them.

Another area is that of blockchain, which has long been a point of interest in security and when coupled with trust-based methods, can be highly fruitful [7]. Indeed, the blockchain, a decentralised immutable ledger [8] well known for its uses in cryptocurrencies such as Bitcoin [9], can be levied to distribute reputation values in a secure way to avoid tampering. All data is stored in structures called "blocks", each containing a reference to its predecessor in the form of a block hash. As a result, any modifications would ripple through the chain and be detected, rendering it immutable, its main advantage [10]. New blocks must also go through an extensive validation process prior to their insertion. This is performed by *Miners*, which calculate a *Proof of Work (PoW)*, confirming a blocks credibility which is subsequently verified by other miners before it can be inserted. Although computationally intensive, this consensus-based *mining* process is the backbone which keeps the blockchain functioning and secure. Due to its immutability, it has been used in other areas such as IoT Security, serving as a decentralised and secure medium for IoT applications [11]. Furthermore, in recent years it has also become an influence in securing routing techniques [12] [13], and has even been used in aviation to secure Unmaned Aircraft Systems against potential threats [14]. However, to access the blockchain it must be stored, which can become a heavy process the more blocks are added. This is further emphasised when the devices using the blockchain possess limited resources, such as storage, computational or even energy capacity.

In this paper, we propose a consensus-based reputation module, providing behavioural analysis to network activities, inspired by [15]. We employ a lightweight version of blockchain, reducing its functionalities to a dissemination tool only, repurposing its *Miners* with the extra responsibility of behavioural validation. Furthermore, we redefine the *PoW* method with our own consensus-based confirmation scheme, corresponding to the specifications and constraints of our network and validation models. As a consequence, these new *validation miners*, significantly different from their blockchain origins, hold a key position in the network. By designing our module with adaptability in mind, it can be used by different routing protocols to influence the route selection process. We illustrate this with the Ad hoc On-Demand Distance Vector (AODV) reactive routing protocol [16], in a new implementation called *AODV-Miner*. By using such a well known protocol as AODV, we can illustrate the functionalities of our approach, and how it interacts with the chosen protocol.

The rest of this paper is organised as follows: Section II defines our system model before presenting *AODV-Miner* in Section III. Section IV presents our results before finally discussing future endeavours and concluding this work in Section V.

978-1-6654-6975-3/22/$31.00 ©2022 IEEE

II. System Model

A. Network Model

We consider a connected wireless network scenario with N static nodes possessing omnidirectional antenna's with a fixed transmission range. Each node is aware of all traffic on the wireless medium in proximity to them at all times. They also possess the ability to determine their own role for the lifetime of a route during discovery, making them either a router or a validation miner, with priority given to routing. As a result, receiving a route request identifies the node as a router, whereas overhearing the request, identifies them as a miner. Subsequently, nodes can participate in multiple routes and can, as a consequence, take on multiple roles.

B. Validation Model

The role of validation miners is 1) to "mine a route", validating routing behaviour between neighbours; and 2) to "mine a block", confirming and distributing the results using the blockchain. For their first objective, each miner has the ability to validate the behaviour of its neighbours. By overhearing passing route requests, they can construct both forwards ($src \rightarrow dst$) and reverse ($dst \rightarrow src$) *Route Validation Tables* (*RVT*) containing the expected hops in order. Each "good" or "bad" action is categorised by the miner for each neighbouring routing node of a specific route. Since the miners parse and extract the expected next hop from their *RVT* to verify the activities, we can determine that the computation and spatial complexities are linked, resulting in $O(n)$, with n nodes in the table.

Once the route has expired, the miners begin their second objective and take on blockchain style responsibilities. Firstly, they aggregate their results into a temporary block which is then shared with neighbouring miners for confirmation. Once complete, the resulting confirmed blocks are again shared with neighbours, updating their network status. As stated previously, we use a lightweight blockchain approach to share blocks, which uses a custom *PoW* method, where miners simply analyse the block's contents and check if the actions are inline with their own vision, responding if an error has been detected. This reduces the number of exchanges needed and makes the miners assume their work is valid if no response is received, disseminating thereafter. In this case, two data structures are explored, increasing the structural complexity to $O(m \times n)$, with m entries in the received block. However, with a worse case scenario of $m = n$, we can deduce the computational complexity to be $O(n^2)$.

C. Threat Model

Routing threat. A malicious node can either simply destroy a packet, or send it elsewhere [17]. In the first case, be it either a complete destruction (*black hole*) or selective (*grey hole*), the concerned data no longer traverses the network. In the second case, the malicious node can either transmit the data to another node using another medium, called *Wormhole* or *redirect* the packet by simply modifying its destination. In case, the malicious node deviates from the expected behaviour and their action's are flagged as bad.

Packet threat. By *modifying* a packet, a malicious node can change its contents. To resolve this, each miner keeps a CRC16 hash of passing packets during validation, thus detecting any modification mid route. Furthermore, if a node re-transmits a packet which has already been seen, known as *replay*, the miners can detect an unexpected hop for the corresponding hash and label the nodes behaviour as bad.

III. Our contribution: AODV-Miner

This section introduces our consensus-based reputation module, illustrated with AODV, named *AODV-Miner*

A. Node Reputation

A node's reputation is calculated based upon their previous actions. If a node acts as expected, by routing a valid packet towards the correct next hop, it is considered to have performed a "good" action. Any other action taken is considered to be malicious and flagged as "bad". By keeping a record of all actions taken by a single node, it is possible to determine their reliability. We define S_{good_n} and S_{bad_n} as the sum of good and bad actions respectively for node n with W_n the action window size, i.e. the number of previous actions taken into account. By varying this value, we can change its precision, limiting it to the most recent actions, or opening it up to a larger portion of node's history.

$$S_{good_n} = \sum_{i=1}^{W_n} good\,actions_{n_i} \quad (1) \qquad S_{bad_n} = \sum_{i=1}^{W_n} bad\,actions_{n_i} \quad (2)$$

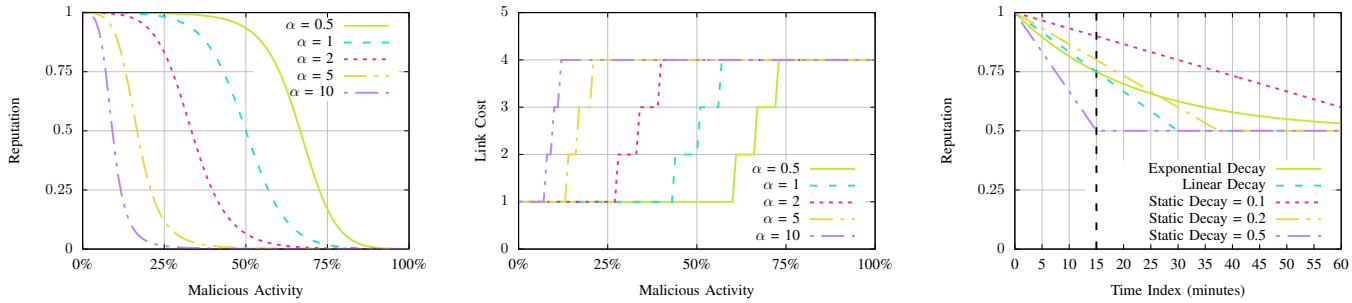
For a specific node, the reputation, $R_n \in [0,\ 1]$, is defined as a sigmoid function where the exponent $\delta_n \in [-1,\ 1]$ corresponds to the weighted value of the relation between S_{good_n} and S_{bad_n}:

$$R_n = \frac{1}{1 + e^{-\delta_n}} \quad (3) \qquad \delta_n = \beta \times \frac{S_{good_n} - \alpha \times S_{bad_n}}{S_{good_n} + \alpha \times S_{bad_n}} \quad (4)$$

where $\beta = 8$ is the sensitivity factor influencing the sigmoid function as in [15] and α the weight of malicious actions. By adjusting the value of α, we modify the severity of bad actions upon the reputation as is shown in Fig.1a. We can see that the value of α influences the reputation, illustrating that the higher the value, the more unforgiving the network becomes.

1) Link Cost: To identify the best route, AODV uses a hop counter which is incremented on each hop. In a similar fashion to [15], we replace the hop count with a different metric called *link cost*, corresponding to the network "cost" of using a specific node based upon their reputation. This method allows us to differentiate between good and bad nodes, where a low reputation will ensue a higher cost. When an RREQ or RREP packet is received, the node calculates the cost of the link between itself and the transmitter. By using the same base functionality of selecting the lowest *hop count*, we encourage the network to select the lowest *link cost*, thus containing as fewer malicious nodes as possible, increasing route integrity at the potential cost of longer routes. We define C_n as the cost of the link between n and its neighbours:

$$C_n = \lfloor (1 - R_{n_t}) \times (C_{max} - (C_{min} - 1)) + C_{min} \rfloor \quad (5)$$



(a) Reputation evolution (b) Link Cost evolution (c) Reputation Decay

Fig. 1: Impact of α on reputation and link cost, and reputation decay with a half life of 15min

where R_{n_t} is the reputation of node n at time t. Since R_{n_t} is normalised between 0 and 1, we can scale the resulting cost by defining minimum and maximum values, C_{min} and C_{max}. We use $C_{min} = 1$ meaning that even using a trustworthy node possesses a cost. Furthermore, the resulting cost is then reduced to the greatest natural number less than or equal to the calculated value. With a maximum value of 255, we can calculate the maximum possible cost based upon the number of potential nodes in the network:

$$C_{max} = \frac{255}{L_{max}} - 1 + C_{min} \quad (6)$$

By decreasing L_{max} (i.e. the maximum possible route length), we can increase the precision of the link cost function. For example, $L_{max} = 32$ would result with a maximum value of 8, whereas $L_{max} = 64$ would only allow for 4 values. The resulting graph can be observed in Fig.1b, corresponding to the link cost of Fig.1a with $L_{max} = 64$. The influence of α is once again visible where we can see that, similar to Fig.1a, the higher its value, the steeper the climb in cost. By using a dynamic scaling function based upon the size of the network, the precision of the *link-cost* metric can be adapted to the situation. Furthermore, by associating it with the `NET_DIAMETER` configuration variable used by AODV, our system can be integrated in a seamless manner.

2) Reputation Decay: Once a node's reputation has been calculated, it will only evolve if the node participates in another route. However, if a node possesses a reputation of 0, it may not be used again in the near future even if it is no longer malicious. In many cases, the malicious device is abandoned by the attacker once it is no longer useful, thus no longer posing a threat bur remaining excluded from routing operations due to its low reputation.

To overcome this issue, we propose a new metric called *Reputation Decay*, where a node's reputation decays overtime when not used towards 0.5, a neutral reputation. By doing so, these abandoned or cleansed nodes can once again be used in routing, allowing them to prove their intentions. However, the decay value does not modify the list of good or bad actions, simply modifying the calculated reputation, making it easier to reincorporate nodes without changing their history:

$$Rd_{n_t} = (t - t_{R_n}) \times (\frac{\lambda}{t_{\frac{1}{2}R}}) \quad (7) \qquad R_{n_t} = R_n - Rd_{n_t} \quad (8)$$

where Rd_{n_t} is the reputation decay of node n at time t, λ the decay factor, $t_{\frac{1}{2}R}$ the half life of the reputation and R_{n_t} (as seen in (5)) the reputation of n at time t, after decay. Fig.1c presents different decay functions used by λ with $t_{\frac{1}{2}R} = 15\,min$. We can see that each function impacts the decay rate in different fashions, from the classic exponential half-life to a more direct Linear or static approach. We decided to use a linear decay function with $\lambda = 0.25$, meaning the reputation will return to neutral after $2 \times t_{\frac{1}{2}R}$.

B. RREP-2Hop

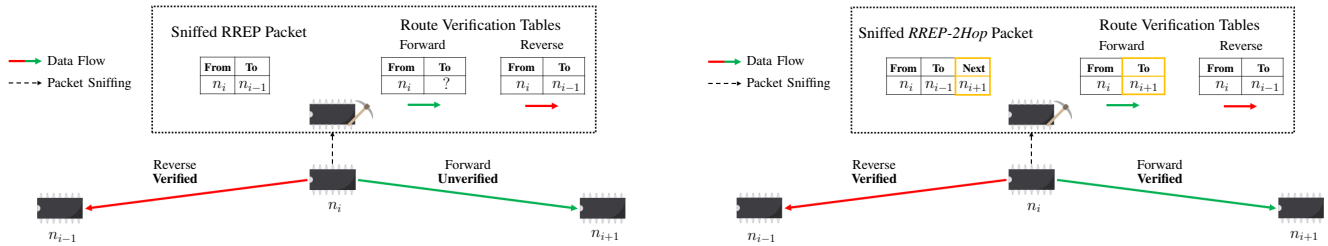
To accurately identify good and bad behaviour, miners need to know the next expected hop for a route. By overhearing RREP's transmitted between neighbours, it is possible to construct both forwards and reverse *RVTs* containing the exact sequence of hops. However, Fig.2a demonstrates the limitation of RREPs, where an RREP from n to $n-1$ only informs of the hop between them, but not the following towards $n+1$.

To remedy this, we propose an update to the RREP packet format called *RREP-2Hop* (Fig.3) to include the addresses of the next hop. Fig.2b illustrates the difference where, when compared to 2a, the hop $n+1$ is known thanks to its layer 2 address. Furthermore, by also providing the layer 3 address, 2-Hop routes can be constructed if so desired. We also add a new flag which allows the receiving node to be informed if the *2Hop* protocol is in use, allowing AODV to function with or without this new addition.

C. Behavioural Validation

In our approach, any node can be a router or a miner for a specific route. Role selection is performed during the route discovery phase by listening for and analysing passing *RREP-2Hop* packets. If a received packet is destined for that node, it is processed as normal, marking the node as a router for that route. If not, the node checks it isn't already a router for this route, as routing and mining for the same route would result in a conflict of interest. This is to reduce the risk of malicious routing nodes injecting false information during the validation phase, thus corrupting the reputation.

If all is well, it then extracts the different addresses from the packet header and adds the corresponding forward and reverse hops to the *RVTs* as seen in Fig.2b. By "mining a route", the miners are responsible for overhearing passing data packets and verifying both the packet's integrity and hop. For this,



(a) Validation with RREP

(b) Validation with RREP-2Hop

Fig. 2: Illustration of the need for RREP-2Hop

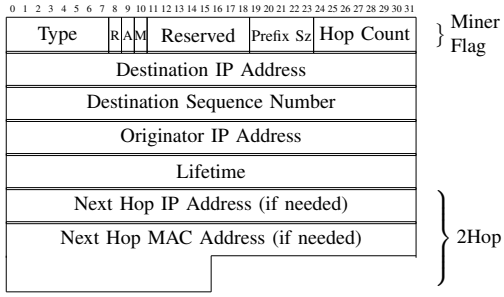


Fig. 3: RREP-2Hop packet structure

Algorithm 1 Route validation run at node n upon reception of pkt(llsrc,lldst,src,dst)

1: **if** New packet detected **then**
2: Create new buf_{pkt} entry with $hash_{pkt}$
3: set buf_{pkt} as valid
4: **else** Previous malicious activity detected ; Exit ;
5: **end if**
6: RTE = Get route entry for $[src \rightarrow dst]$
7: RVT = get validation tables from RTE for $llsrc$
8: **if** RTE & RVT both empty **then**
9: ▷ No route validation table, Malicious behaviour
10: Increment bad_{llsrc}; Set buf_{pkt} as invalid
11: **else**
12: $nextHop_{pkt}$ = get the next hop from RVT
13: **if** $nextHop_{pkt} \neq lldst$ **then** ▷ $lldst$ is not the next expected hop - Malicious behaviour
14: Increment bad_{llsrc}; Set buf_{pkt} as invalid
15: **else** ▷ Valid behaviour
16: Increment $good_{llsrc}$
17: **end if**
18: **end if**

each miner levies the corresponding *RVT* to determine the next expected hop. However, we are only able to validate packets originating from either end of the route and not intermediate transmissions.

As packets traverse the network, the different miners evaluate the behaviour of each routing node by performing *Route Validation* (see Alg. 1) so long as the route remains active. Once expired, each miner checks their *Packet Buffer* for potentially dropped packets which haven't completed all expected hops. If drops are detected, the corresponding node's bad actions are updated, before the miner begins preparations for blockchain dissemination. Before dissemination can take place, the data must first be confirmed using a custom consensus-based method. This method replaces the standard *PoW* used in blockchain, where instead of competing to solve a puzzle, the nodes simply request confirmation from their neighbours. By using such an approach, we not only reduce *PoW's* heavy computation, but also provide a simple method for sharing only valid data.

The miner, therefore, creates a temporary block containing all calculated actions which is then broadcast up to two hops to reach only miners who could have mined the same portion of route (i.e. the same nodes). Upon receiving a block, each miner proceeds with two calculations: First, they compute the difference ratio in common nodes between the received block and their own. If this value reaches a certain threshold (i.e 80%), the received block is considered invalid and the miner transmits their own instead.

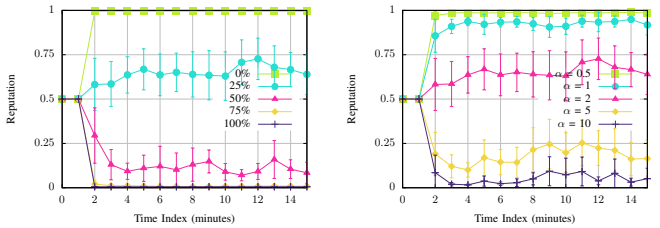
If, however, it is valid then the miner determines the efficiency factor by calculating the percentage of nodes in common in the received block, P_B with their own P_M, with M the nodes mined and B the nodes in the received block.

$$P_B = \frac{|M \cup B|}{|B|} \quad (9) \qquad P_M = \frac{|M \cup B|}{|M|} \quad (10)$$

If $P_M >= P_B$, we consider B to be more efficient as it contains more nodes overall. By using the efficiency factor, we can send as few blocks as possible, thus increasing efficiency and reducing overhead. However, this process relies on other miners to "overrule" previously transmitted blocks, indicating that they are no longer considered valid and theirs should take its place. This serves two purposes: correcting miners and determining the most efficient block. If, however, no response is received, the transmitter miner considers their block valid. They then hash the contents, including the hash of the previous block, before adding it to the blockchain. It is then broadcast up to two hops so all neighbouring nodes can extract the list of actions.

D. Implementation

Further to the two *RVTs*, each node contains a *Packet Buffer* and a *Node Reputation Table*. The former stores the CRC16 hashes of passing packets during routing, with their next expected hop. The latter contains the list of node actions extracted from the blockchain used to calculate the reputation with Eq. (1) - (4). In our implementation, we emulate a lightweight blockchain, where the blocks are not stored but



(a) Reputation overtime with varying malicious activity

(b) Impact of α with 25% malicious activity

Fig. 4: Evolution of reputation

TABLE I: Simulation Parameters

Parameter	Value	Parameter	Value
Area	150m×150m	Transmission Range	50m
Max length (L_{max})	64	Number of Nodes (N)	30
Malicious Activity	100%	Malicious Weight (α)	2
Reputation Decay	Linear	Window Size (W_n)	5
Initial Reputation	0.5	Number of Simulations	100
Simulation Duration	15 min.		

broadcast up to two hops, reaching only the neighbours of the nodes contained in the block. Another change is the redefinition of the *PoW* consensus-based block validation method. This functionality is integrated directly into the validation miners, allowing them to automatically confirm their work, without interactions with the blockchain. As a result, only confirmed blocks are ”inserted” into the chain, keeping the contained information as valid as possible.

With each passing RREQ and *RREP-2Hop*, the receiver calculates the *reputation decay* and *link cost* using Eq. (8) and (5) of the transmitter. By checking that the new cost is higher than the previous, we can protect against potential field overflow. By only forwarding RREQs with lower *link costs*, we can propagate more reliable routes towards the destination, which waits a certain amount of time for as many RREQs as possible, before responding only to the most reputable path.

IV. Results

AODV-Miner was implemented with Contiki-NG [18] and simulated using Cooja. The different parameters used in the simulations are presented in Table I. Each node possesses a wireless interface using the IPv6 netstack with 6LoWPAN and a non-beacon-enabled always on CSMA radio to reduce potential collisions. For our analysis, we consider Malicious Nodes to perform *black hole* attacks, which are distributed throughout the network at random. This preliminary study allows us to validate our implementation using a simple form of attack, paving the way for more advanced attack types in the future. We also consider that this attack drops only data packages, leaving AODV or block related traffic intact. Our analysis pitches *AODV-Miner* against its older brother, AODV.

A. Reputation Analysis

Fig.4a shows the calculated reputation based on malicious activities. With $\alpha = 2$, a 25% malicious node has a reputation close to neutral, whereas higher rates of activity rapidly decrease the value. We can also see that they are attributed

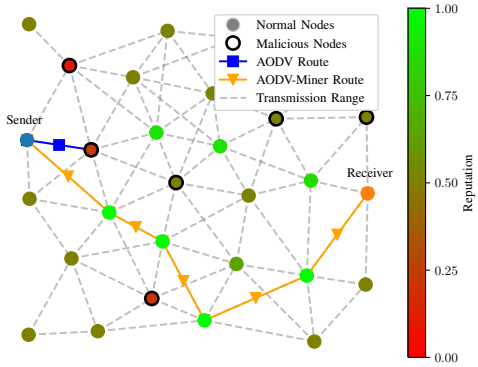


Fig. 5: Visualisation of route reputation after 15 min with 25% malicious nodes.

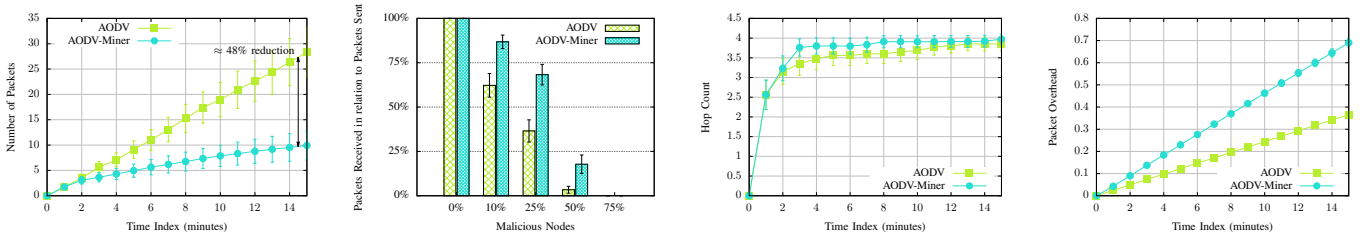
directly after the first route expires at around the 1 min. mark and stay in the same overall vicinity.

Fig.4b extends this and illustrates the influence of α with 25% malicious activities. We can see that, conforming to our initial hypothesis, the higher the value of α, the quicker the reputation drops, and vice-versa. We can, therefore, actively influence the weight of bad behaviour, instantly punishing a node for misbehaving or forgiving them quickly.

Fig.5 presents the status of one of the simulated networks of 30 nodes after 15 mins. with 25% of them acting as black holes (thick circled), superimposing node reputation and the most used route by AODV and *AODV-Miner*. We can see that where AODV uses the most direct route via a malicious node, *AODV-Miner* is able to select a clear path. We can also see that we are able to assign a bad reputation to three malicious nodes, allowing them to be avoided, whereas all nodes in the route have received a good reputation. Furthermore, we can see that other nodes also possesses varying levels of good reputation, meaning that at some point in time they were used to route data, with all other nodes possessing a neutral reputation of 0.5. As a result, we can conclude that the reputation metric is crucial in our approach to limit malicious nodes from impacting routing activities.

B. Route Analysis

Fig.6 compares the efficiency of *AODV-Miner* to AODV. We use the number of packets dropped ($|Sent| - |Received|$) to determine the network throughput, visible in Fig.6a and 6b. We can see that the number of packets dropped is reduced by $\approx 48\%$ with 10% of nodes being malicious, resulting in a clear increase in the corresponding throughput. It is also noticeable that whatever the percentage of malicious nodes, *AODV-Miner* possesses a higher throughput than AODV. It must be noted, however, that not all drops are prevented since reputations are computed based upon malicious activities, allowing time for nodes to wreak havoc. Furthermore, in many cases traversing a malicious node with a *link cost* of 4, still has a lower cost than five nodes with a *link cost* of 1. That being said, this better efficiency comes at a cost as confirmed by Fig.6c, where we can see that routes are on average longer in *AODV-Miner*. Another cost is related to the activity of the miners, where sharing blocks results in an uptake in packet transmissions.



(a) Packets dropped with 10% malicious nodes

(b) Throughput with varying presence of malicious nodes

(c) Average route length with 10% malicious nodes

(d) Normalised overhead with 10% malicious nodes

Fig. 6: Routing efficiency between *AODV-Miner* and AODV in the presence of malicious nodes

Fig.6d puts this into perspective by normalising the overhead of *AODV-Miner* in relation to AODV. We can see that our overhead is higher than AODV's, confirming that there is a compromise where the increase in security comes at a cost.

V. Conclusion & Future Works

In this paper, we introduce a consensus-based reputation metric to identify the most trustworthy route available. By exploiting blockchain technology to disseminate the reputation in the network, we assure that no intruders can falsely modify a nodes reputation. Furthermore, by introducing a validation technique based on miner consensus, we can quickly and accurately identify both highly trustworthy nodes as well as malicious entities. Finally, by incorporating a reputation decay functionality, we also reduce the risk associated with compromised trustworthy nodes as well as assure the reintegration of sanitised nodes back into the network. By analysing its functionalities in conjunction with a reactive routing protocol such as AODV, we demonstrate that our system can detect and avoid malicious devices. Through extensive simulations pitching this protocol *AODV-Miner* against routing based threats, we have not only proved the efficiency of our approach against AODV, but also the importance of reputation-based routing in multi-hop networks.

However, our increased efficiency comes at a cost with a rise in packet overhead due to the lightweight implementation of blockchain. Indeed, although our module uses as few communications as possible for block validation, storage is still an issue, meaning each block must be broadcast to the other nodes. Due to the static nature of our scenarios, broadcasting blocks up to two hops is sufficient to inform nodes of the status of their neighbours. By extending this preliminary analysis with variable probability based attacks, such as grey holes, we provide more of a challenge compared to the situation provided by black holes. Furthermore, since our module was developed outside of a specific routing protocol, it can be adapted onto other protocols for further in-depth analysis.

Acknowledgements

This work was partially supported by a grant from CPER DATA and by the European Union's H2020 Project "CyberSANE"

References

[1] F. Bao, I.-R. Chen, M.J. Chang, and J.-H. Cho. Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection. *IEEE Transactions on Network and Service Management*, 9(2):169–183, 2012.

[2] D. K. Bangotra, Y. Singh, A. Selwal, N. Kumar, and P. K. Singh. A trust based secure intelligent opportunistic routing protocol for wireless sensor networks. *Wireless Personal Communications*, pages 1–22, 2021.

[3] N. Djedjig, D. Tandjaoui, F. Medjek, and I. Romdhani. Trust-aware and cooperative routing protocol for iot security. *Journal of Information Security and Applications*, 52:102467, 2020.

[4] J. Tang, A. Liu, M. Zhao, and T. Wang. An aggregate signature based trust routing for data gathering in sensor networks. *Security and Communication Networks*, 2018, 2018.

[5] Weidong Fang, Wuxiong Zhang, Wei Yang, Zhannan Li, Weiwei Gao, and Yinxuan Yang. Trust management-based and energy efficient hierarchical routing protocol in wireless sensor networks. *Digital Communications and Networks*, 7(4):470–478, 2021.

[6] L. Guillaume, J. van de Sype, L. Schumacher, G. Di Stasi, and R. Canonico. Adding reputation extensions to aodv-uu. In *IEEE Symp. on Comm. and Vehicular Technology in the Benelux (SCVT)*, 2010.

[7] A. Moinet, B. Darties, and J.-L. Baril. Blockchain based trust & authentication for decentralized sensor networks. *ArXiv*, abs/1706.01730, 2017.

[8] NARA. Blockchain white paper. White paper, National Archives and Records Administration, February 2019.

[9] A. M Antonopoulos. *Mastering Bitcoin: Programming the open blockchain*. " O'Reilly Media, Inc.", 2017.

[10] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen. A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107, 2020.

[11] M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, and M. H. Rehmani. Applications of blockchains in the internet of things: A comprehensive survey. *IEEE Com. Surveys Tutorials*, 21(2), 2019.

[12] C. Machado and C. M. Westphall. Blockchain incentivized data forwarding in manets: Strategies and challenges. *Ad Hoc Networks*, 110:102321, 2021.

[13] H. Lazrag, A. Chehri, R. Saadane, and M. D. Rahmani. A blockchain-based approach for optimal and secure routing in wireless sensor networks and iot. In *Int. Conf. on Signal-Image Technology Internet-Based Systems (SITIS)*, 2019.

[14] J. Wang, Y. Liu, S. Niu, and H. Song. Lightweight blockchain assisted secure routing of swarm uas networking. *Computer Communications*, 165:131–140, 2021.

[15] M. A. A. Careem and A. Dutta. Reputation based routing in MANET using Blockchain. In *Int. Conference on COMmunication Systems NETworkS (COMSNETS)*, 2020.

[16] S. R. Das, C. E. Perkins, and E. M. Belding-Royer. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.

[17] E. Staddon, V. Loscri, and N. Mitton. Attack categorisation for iot applications in critical infrastructures, a survey. *Applied Sciences*, 11(16), 2021.

[18] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes. The contiki-ng open source operating system for next generation IoT devices. *SoftwareX*, 18:101089, 2022.