RDSR-V. Reliable Dynamic Source Routing for video-streaming over mobile ad hoc networks

Jose L. Muñoz *, Oscar Esparza, Mónica Aguilar, Victor Carrascal, Jordi Forné

Department of Telematics Engineering, Universitat Politècnica de Catalunya (UPC), Campus Nord, C3 Building, Office 310, Barcelona 08034, Spain

**A R T I C L E  I N F O**

Article history:
Received 22 December 2008
Received in revised form 6 July 2009
Accepted 19 August 2009
Available online 28 August 2009

Responsible Editor: G. Morabito

Keywords:
MANET
QoS
Video-streaming
Reputation

**A B S T R A C T**

Mobile ad hoc networks (MANETs) are infrastructureless networks formed by wireless mobile devices with limited battery life. In MANETs for civilian applications, the network nodes may not belong to a single authority and they may not have a common goal. These MANETs are particularly vulnerable to selfish behavior, as some nodes may prefer saving resources to forward data. There are a few generic reputation-based systems for MANETs which could be used to enforce cooperation among nodes. However, we envision that the system performance can be highly improved by using cross-layer techniques that take into account the specific characteristics of each particular service. In this article, we propose a distributed and easy-to-implement routing mechanism based on reputation for the provision of MPEG-2 video-streaming services over MANETs. The main novelty that we introduce regarding the existent literature is that our proposal is service aware, that is to say, we consider the video-streaming service characteristics to develop a cross-layer design with the routing protocol. In addition, we do not introduce extra signaling overhead to monitor reputation because we use the standard video-streaming end-to-end signaling. Finally, simulation results show that our proposal clearly outperforms both standard Dynamic Source Routing (DSR) and OCEAN (a generic reputation-based mechanism).

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

A mobile ad hoc network (MANET) is a network created spontaneously by a set of wireless mobile devices. These devices communicate with each other on a peer-to-peer basis, without the need of a fixed network infrastructure nor centralized administrative support. MANETs suffer from link breakages and frequent changes of network topology because the nodes make arbitrary movements and they have a limited battery life. In addition, intermediate nodes are required to carry out end-to-end communications since the transmission range of wireless devices is limited. Therefore, cooperation among nodes is required.

MANETs that are used in tactical applications (such as emergency rescue or exploration missions) are managed by a single entity, thus nodes certainly collaborate to reach a common goal. On the contrary, commercial applications over MANETs (e.g. conferences, individual museum visits, freely city tours, peer-to-peer applications, e-gaming) are particularly vulnerable to selfish behavior. Selfish nodes just want to save resources, especially battery. This misbehavior is very specific of MANETs, in which nodes are mobile devices that usually have energy constraints. On the other hand, malicious nodes attempt to maximize the damage caused to the system and the only method to deal with maliciousness is to detect these nodes and eject them from the network [1]. Malicious behavior is beyond the scope of this article. Selfish nodes that do not cooperate may produce additional path breakages and we must encourage them to be cooperative by applying a cooperation enforcement mechanism. In this sense, reputation systems are proved mechanisms to prevent selfish behavior and to promote fair collaboration. In the literature we...
can find some generic reputation systems such as OCEAN [2], CORE [3] or CONFIDANT [4] that in general enhance the network performance in presence of selfish nodes.

In this article, we propose a novel service-aware reputation-based routing protocol for video-streaming over MANETs named RDSR-V (Reliable Dynamic Source Routing for Video). RDSR-V is a DSR-based variant that chooses the best path to forward each video-stream applying cross-layering between routing and service layers. Standard DSR always selects the shortest path to forward data. RDSR-V assigns a forwarding reputation value to paths and individual nodes that are being used or have been used to forward video-streams. Then, we use these reputation values to select the best path in order to obtain the best end-to-end Quality of Service (QoS). In our opinion, the high amount of traffic generated by typical video applications can clearly justify the use of a service-specific reputation system. On the other hand, to update the reputation scores, we monitor the video-streaming service feedback instead of using the typical watchdog mechanisms based on promiscuous monitoring. Promiscuous monitoring forces nodes to analyze great amounts of data not destined for them. This kind of mechanism can make the system impractical since nodes may waste too much battery in the monitoring process. On the contrary, we propose a novel approach which uses the video-streaming standard signaling packets (RR-RTCP packets). This way of monitoring the video-stream forwarding reputation allows a source node to monitor end-to-end QoS parameters without introducing new overhead. It is important to point out that, differently to the most reputation systems proposed in the literature, like CORE [3] or OCEAN [2], we are not directly monitoring the behavior of individual nodes, but we are monitoring the overall behavior of the transmission path from the source to the destination. This allows us to enhance the end-to-end QoS provided to the final user. Similarly to other reputation systems, RDSR-V introduces cooperation incentives but in our case, these incentives have been especially devised with the final objective of improving the end-to-end QoS of the video-streaming service. Also, a performance evaluation of the proposal has been carried out to show that RDSR-V clearly outperforms standard DSR and OCEAN for the video-streaming service in presence of non-cooperative nodes. The rest of the article is structured as follows: Section 2 provides the reader with the necessary background. Section 3 introduces RDSR-V, our reputation DSR-based routing protocol for the video-streaming service. Section 4 presents an analysis of our proposal including simulation results with ns-2. Finally, we conclude in Section 5.

2. Background

Video-streaming services enable the transmission of video files through the network in the form of continuous-time flows of data packets named video streams. This is a challenging service when deployed over a MANET [5,6]. In this article, we propose a distributed and easy-to-implement routing mechanism based on DSR (Dynamic Source Routing) [7]. The routing mechanism is also based on reputation for the provision of MPEG-2 video-streaming services over MANETs.

2.1. DSR (Dynamic Source Routing)

Basically, the DSR routing protocol works in two phases: route discovery and route maintenance. During the route discovery phase, the source node (i.e. the node performing the route discovery) broadcasts a Route Request (RREQ) packet. Each node that receives a RREQ adds its own IP address to the packet and broadcasts it again. The RREQ packets received at the destination node contain complete routes from the source node. After receiving an RREQ, the destination sends back a unicast Route Reply (RREP) packet. The RREP packet includes the list of nodes that must be traversed. As the route discovery is a costly process, nodes store discovered routes (or paths) in a Routing Table (RT) in cache for a certain period of time. Each route consists of a list of IP addresses. Finally, when a node discovers that the link with a neighbor node is broken, it sends an Error Message (RERR). When nodes receive RERR packets, they react to the topology change by starting a route maintenance phase to remove old routes and to find new ones if necessary. Also, the route maintenance phase can be started to remove inactive routes, i.e. routes that are considered old because they have not been used to forward any data for a certain period of time.

2.2. MPEG-2 (hierarchical scalable multi-layer encoded video)

The video-stream must be encoded and packed to be sent over the MANET. One of the most used encoding systems is the hierarchical scalable multi-layer encoded video (MPEG-2) [8]. MPEG-2 encoded video consists in sets of frames, typically somewhere from 4 to 20 frames each, called GoP (Groups of Pictures). In a GoP, there are three types of frames: I, P and B (Bi-directional) frames encode spatial redundancy and compose the base layer which provides a basic video quality. I-frames are absolutely necessary to decode the video sequence, so the entire GoP would be lost if the corresponding I-frame is not available at decoding time. P (Predicted) and B (Bi-directional) frames provide enhancement layers, so that fine-granularity scalability can be achieved. RTP/RTCP (Real Time Protocol/Real Time Control Protocol) packets over UDP (User Datagram Protocol) are usually used to deliver video frames. RTP packets are used to deliver the video frames in their payload, whereas RTCP provides feedback information associated to RTP, which can be used to monitor the QoS at the receiver side.

2.3. Cooperation enforcement

Service performance depends to a great extent on node cooperation. In fact, due to the lack of infrastructure, each single device within the network is required to be operational and efficient. Besides using the network resources, each node also contributes to the network operation, so that the higher the cooperation, the better the performance. Thus, it is important to encourage node participation to achieve a certain level of QoS. In the literature
there are several proposals to enforce cooperation among users. These proposals can be classified as credit-based and reputation-based:

- **Credit-based models.** Network tasks are treated as services that can be valued and charged. These models incorporate a form of virtual currency to regulate the dealings among nodes. Unfortunately, these systems require the existence of tamper-resistant hardware or a virtual bank, heavily restricting their usability in MANETs.

- **Reputation-based models.** A subject commands the execution of an action to an agent. If the agent responds satisfactorily to the subject, the latter will increase the reputation value of the agent; otherwise, the corresponding reputation value will be decreased.

Reputation-based models are envisioned as the more promising solution for cooperation enforcement in MANETs. A reputation-based model has to provide mechanisms to measure, assign, update and use reputation values. Global reputation refers to the case in which every node knows the reputation of every other node in the network. This is achieved by exchanging indirect reputation messages inside the network. In local reputation, however, information is based only on direct observations of one-hop neighbors (first-hand information) and any second-hand reputation exchanges are disallowed, which reduces the amount of signaling. Reputation systems are composed of the following modules:

- **Monitoring module.** The subject collects information about the behavior of other entities. This information may come from own experiences (first-hand or local), or the node can also take into account recommendations from other nodes (second-hand or global). Notice that, in the latter, an effective mechanism to distribute reputation information among nodes must be deployed.

- **Qualifying module.** The subject computes a reputation value according to the previous experiences regarding each monitored agent. If recommendations are considered (second-hand), there will also be the need for an aggregation algorithm that assigns weights to each opinion (based on recommendation trust or credibility) to obtain the final reputation value of each agent. Finally, it is also necessary to set an initial reputation value to unknown agents.

- **Decision making module.** This module makes decisions and performs the corresponding actions considering the reputation values of each node. The subject can apply correction actions to other participating entities depending on their reputation scores. In this sense, there can be rewarding and punishing actions.

In [1], two kinds of selfish misbehavior are defined regarding the network layer:

- **Selfish routing misbehavior:** when a selfish node does not participate during route discovery. Hence, this silent node is excluded from the routing list of other nodes, saving its power as it is not required to forward packets for others. Nevertheless, it sends its own traffic through the network, unfairly preserving its resources while using others’.

- **Selfish forwarding misbehavior:** when a selfish node fully participates in the route discovery phase, but it refuses to forward packets for others, or it degrades other node’s service by performing a faulty forwarding.

Reputation systems can face both kinds of misbehaviors. They can frustrate the intentions of selfish nodes since these systems can cope with observable misbehaviors. If a node does not behave cooperatively, the affected nodes will deny reciprocal cooperation. Some reputation systems dealing with these misbehaviors have been proposed in the literature, such as CORE [3], CONFIDANT [4], SAFE [9] and OCEAN [2]. However, these proposals do not reveal the impact of misbehavior over each type of upper-layer service.

3. RDSR-V: Reliable Dynamic Source Routing for Video

In this section, we start presenting the main design requirements for RDSR-V and providing the reader with a general overview of the protocol. Afterwards, we describe RDSR-V in detail.

### 3.1. Requirements

- **Selfish forwarding misbehavior.** Selfish forwarding nodes can adversely affect the QoS of a video-streaming service because they can cause link degradation in an already established path. Unfortunately, MANETs are susceptible of having selfish nodes unwilling to be used for their resources, although still using the network to relay their traffic. They may respond positively to route requests but then fail to forward packets to save battery power. For that reason, we focus our efforts on designing a proposal able to identify faulty forwarding paths and selfish forwarding nodes.

- **Selfish routing misbehavior.** Our proposal is mainly focused on selfish forwarding nodes because they cause damage in already established paths which is the worst situation for the video-streaming service, but we also face to some extent the problem of silent nodes. Silent nodes do not participate in route discovery and, thus, they do not cause damages in ongoing video-streaming sessions. However, they do not cooperate by routing packets of other nodes. To deal with this problem, RDSR-V encourages silent nodes to cooperate, as it includes a rewarding mechanism to benefit cooperating nodes.

- **Routing mechanism based on DSR.** We chose DSR as a starting point to develop our proposal because it is an IETF standard that can be used to find all the available routes to a destination. Furthermore, the source node has the list of IP addresses of nodes that belong to a particular route. This allows nodes using DSR to control which nodes will be involved in a particular end-to-end communication. Our goal is to improve the DSR performance in presence of selfish nodes. Regarding this, we have developed a path selection algorithm that takes into account observed reputation.
Service specific (video-streaming). A major objective is to devise a reputation mechanism aware of the kind of service being forwarded. We specifically consider the transmission of MPEG-2 encoded video. The high amount of traffic generated by typical video applications can clearly justify the use of a service-specific reputation system.

Simple and easy to implement. Our objective is to develop a straightforward, completely distributed protocol which introduces very low overhead. A scalability analysis of the proposal is presented in Section 3.7.

Observations based on service feedback. We use an end-to-end mechanism to monitor reputation. We use service feedback provided by standard signaling RR-RTCP packets (Receiver Report-Real Time Control Protocol packets). This is an end-to-end way of monitoring the video-stream forwarding reputation and it allows the source to monitor QoS parameters of the forwarding paths without introducing extra overhead.

Only first-hand observation (local reputation). Our proposal is a local (first-hand) reputation system, like OCEAN [2], since it only relies on own experiences. This feature is important to keep the system simple and at the same time to reduce the amount of signaling.

Security requirements. There are an important number of works in the literature especially designed to deal with security attacks. The proposal presented in this article aims at enforcing cooperation among nodes which is a different purpose. However, our proposal can be applied along with secure routing protocols to improve their performance. A further security analysis of our proposal is presented in Section 3.8.

Rewarding and punishment actions to force collaboration. Similarly to many reputation systems, cooperation among nodes is enforced through rewarding and punishment actions. In RDSR-V, these actions have been especially devised with the final objective of improving the QoS for MPEG-2 video-streaming service.

3.2. Our proposal in a nutshell

A basic description of RDSR-V is shown in Fig. 1. RDSR-V fulfills all the design requirements which have been specified in Section 3.1.

Essentially, RDSR-V works as follows:

- The source selects the path to forward the video-stream taking into account not only the number of hops but also the reputation of paths.
- During the video transmission, the source uses service feedback information to switch to a different forwarding path if necessary. This could depend on degradation of the current path, on possible node unavailability, and on any other circumstance that could disturb the proper forwarding of the video-stream.
- The source updates the reputation scores of the corresponding path and of the nodes that belong to that path using the service feedback information (RR-RTCP packets).
- The mechanism performs punishment actions over selfish nodes as well as rewarding actions over collaborative nodes, seeking to enforce network cooperation. In RDSR-V, nodes assign different priorities in its internal queues when forwarding the packets for collaborative, medium-reputation or selfish nodes.
- Silent nodes are treated as medium-reputation nodes. For this reason, they do not obtain the best possible QoS for an intermediate node. This action is an incentive to cooperate.
- Finally, if the reputation of a node decreases until a certain threshold, the source treats it as non-collaborative, and RDSR-V reacts preventing that node from being part of forwarding paths.

As a local (first-hand) reputation system, RDSR-V inherits several characteristics of OCEAN [2]. However, since our proposal is specific for the video service and uses service feedback instead of observation of the neighbors’ behavior, there are important differences between OCEAN and RDSR-V. The rest of this section is devoted to explain RDSR-V in detail and to show these differences.

3.3. Monitoring module

Traditionally, the common principle for monitoring the forwarding behavior consists of verifying that the next node retransmits the packets sent to it. This requires using the promiscuous mode of the wireless interface and a watchdog process. Unfortunately, there are a number of situations
for which this mechanism is not efficient to appreciate the behavior of the next node. To give an example, it may happen that node A cannot hear the next node B correctly forwarding a packet to node C because A receives at the same time another packet from node D which produces a collision. This is the so-called hidden terminal problem and it is inherent in wireless networks. Furthermore, the extensive use of promiscuous listening may lead to unbearable CPU load and buffering capacity requirements, especially for resource-constrained mobile devices such as PDAs.

In RDSR-V, we propose to use service feedback from destination instead of promiscuous monitoring. In the case of video-streaming, we use the standard signaling RR-RTCP packets. The RR-RTCP packet is used as feedback because it contains information about packet losses, latency and delay jitter of the video stream. With the information provided by RR-RTCP packets a reputation forwarding value (RFV) has to be assigned as a function of the monitored service feedback. More specifically, RFVs are assigned in terms of the Fraction Packet Lost (FPL) included in the RR-RTCP packets. Notice that this way of monitoring the video-stream forwarding allows the source to monitor the end-to-end QoS. It is important to point out that, differently to the most reputation systems proposed in the literature [2,3], we do not directly monitor the behavior of individual nodes, but we monitor the overall behavior of a transmission path (from a source to a destination). These issues are further discussed when the qualifying module is explained in the next section.

Finally, it is worth to mention that our proposal, like OCEAN [2], is a local (first-hand) reputation system because it only relies on own experiences. Nonetheless, OCEAN only gathers information of neighbor nodes through promiscuous observations, whereas our mechanism gathers information of distant nodes thanks to the end-to-end QoS monitoring. RDSR-V is also more resilient to the rumor spreading phenomena than second-hand systems like CORE [3] or CONFIDANT [4] since RDSR-V does not use recommendations. In this way, we avoid evaluating the trustworthiness of the recommender since the only source of feedback information is the destination of the service. By contrast, the application entities (sender and receiver) are required to be authenticated and integrity of feedback packets must also be assured. In Section 3.8, we further discuss the most important security aspects of the proposal.

### 3.4. Qualifying module

RDSR-V qualifies the reputation data each time one of the following events occurs:

- Feedback packets:
  - A new RR-RTCP packet is received.
  - A certain RR-RTCP packet is not received when expected (its feedback timeout expires).
- Routes:
  - A new DSR route is discovered (that route must be cached).
  - A DSR route is inactive for a certain period of time or it is no longer valid because of a topology change (the route must be removed from cache).

#### 3.4.1. Processing feedback packets

The first event that we have to manage is the reception of a feedback packet. Let us assume that a source node $\mathcal{S}$ is sending a video-stream to a destination $\mathcal{D}$ using a certain path $\mathcal{P}$. Let us also assume that the path $\mathcal{P}$ is composed of $K$ nodes. When a service feedback RR-RTCP packet is received, the Fraction of Packet Losses (FPL) parameter is extracted and the function `ProcessFeedback()` of Algorithm 1 is executed. The function `MapReputationSample()` maps the losses (FPL) included in the RR-RTCP packet to a Reputation Forwarding Value (RFV). In general, in reputation systems, the reputation score is expressed with a numerical value that ranges from 0 to 1. The worst reputation is assigned to 0 (which in our case means the worst QoS) and the best reputation is assigned to 1 (which in our case means the best QoS and fully cooperative nodes). A timeout will expire if a feedback packet is not received when expected. The timeout will be interpreted as that the corresponding service data packets have been lost and the reputation score of the path will be updated (decreased) accordingly.

```plaintext
1 ProcessFeedback(P,FPL) {
2 RFV_sample = MapReputationSample(FPL);
3 UpdateHistory(P,RFV_sample);
4 PunishAndReward(P);
5 }
6 TimeoutFeedback(P){
7 ProcessFeedback(P,1);// If a feedback packet is lost, then FPL=1
8 }
```

**Algorithm 1:** Pseudo-code for processing RR-RTCP packets or feedback timeout.

In our proposal, reputation scores are related to the actual video performance perceived by users, which is really suitable for a video-streaming service. In particular, we use the users’ Mean Opinion Score (MOS), which provides a subjective evaluation of the video quality experienced by the users. The evaluation can be EXCELLENT(5), GOOD(4), FAIR(3), POOR(2) or BAD(1) as specified by ITU-T recommendations P.801 [10] and P.930 [11]. The MOS score is related to the PSNR (Peak Signal-to-Noise Ratio), which is an objective measurement of the video quality. This relationship can be found in the specific video literature [12,13] which are based on user polls. In addition, the relationship between the PSNR and the FPL has been extensively studied in the literature [12–16]. Table 1 shows the relationship between these parameters.

The thresholds $\beta$, $\tau$ and $\gamma$ of the previous table are critical design parameters since they set the reputation level
at which rewarding and punishment mechanisms are applied. Actions performed by RDSR-V are (see Section 3.5 for further details):

- **Actions related to nodes.** We use three priorities: packets of a source node will be forwarded with a high priority by intermediate nodes if the source’s reputation score is above the threshold $\beta$. If a source node has a score below the threshold $\beta$, it will be considered selfish and its packets will be forwarded with low priority. Finally, packets of medium-reputation nodes ($\gamma \leq \text{RFV} < \beta$) will be forwarded with medium priority. In addition, if a destination node has a reputation score under the threshold $\gamma$, it will be considered selfish. Sources do not provide video-streaming services to selfish destination nodes (however a selfish node can be redeemed as shown later in Section 3.5).

- **Actions related to paths.** RDSR-V implements a path selection algorithm that takes into account the reputation and tries to avoid paths with a reputation score below the threshold $\gamma$. Also, if the reputation of a path currently being used by a video-streaming session reaches the reputation threshold $\tau$, RDSR-V tries to switch the video-streaming session to another available path.

More specifically, we use an accumulated historical reputation score for paths and nodes to make the decisions. The function $\text{UpdateHistory()}$ is used to update this historical reputation (see Algorithm 2). As shown in the algorithm, we use an Exponentially Weighted Moving Average (EWMA) filter that takes into account the accumulated reputation and the current sample using a low aging factor $\alpha$: $\text{RFV}_{\text{history}} = (1-\alpha)\text{RFV}_{\text{history}} + \alpha \text{RFV}_{\text{sample}}$. As a conclusion, the aging factor $\alpha$, and the thresholds $\beta$, $\tau$, and $\gamma$ are parameters that greatly define how the system behaves. We have carried out a set of simulations to tune these parameters for a small-medium MANET\(^1\) and finally, we found that $\alpha = 0.25$ and $\text{RFV} = e^{-14 \text{FPL}}$ is a suitable conservative setting for our scenario (see Section 4.1 for further details). Below, we provide a discussion about the effect of these parameters.

The parameter $\alpha$ can be tuned to detect misbehaving nodes rapidly or to have an accurate measure of the trustworthiness of the nodes. With a low $\alpha$ (close to 0), it may take a long period of time to detect selfish nodes, but the detection is more trustworthy. With a high $\alpha$ (close to 1), a higher weight is given to current qualifications than to previous ones and thus, RDSR-V is able to detect misbehaving nodes faster. However, the system becomes very sensitive to instant misbehavior which can lead to erroneous detections of selfishness. When the EWMA filter is applied to update the reputation of individual nodes (see function $\text{UpdateHistory()}$ in Algorithm 2), notice that the aging factor is divided by the number of nodes of the path,\(^2\) i.e. $K$. This is a way of reducing a negative impact over non-selfish nodes that belong to a path in which there is a selfish node. Regarding this, the higher the number of nodes a path has, the lower influence that a reputation measure will have on each individual node. For instance, if a path is formed by a single node, the measure will be fully considered for this node. On the other hand, if the path is formed by various nodes, the measure will be shared among them.

The parameters $\beta$ and $\gamma$ essentially define the punishment and rewarding policies. A high parameter $\beta$ (close to 1) means that nodes must provide a very good QoS to obtain the reward. This makes the system very strict and may discourage cooperation. Also, being too strict is unfair with non-selfish nodes that belong to a path in which there is a selfish node. However, if we relax this threshold too much, nodes may cooperate just enough to obtain the reward. Similarly, if $\gamma$ were too low, nodes would not be punished until they extremely misbehave. Conversely, for too high values $\gamma$, nodes could be wrongly detected as selfish. The parameter $\tau$ defines when a reputation decrease triggers a switch of the current path.

Going back to the $\text{UpdateHistory()}$ function, we need to define what happens when we do not have previous history for a path or node. To manage unknown nodes, we assign a neutral value to the RFV.\(^3\) For never used paths, we can estimate their a priori reputation from the individual reputation values of nodes that belong to that path. In Algorithm 2, we provide the $\text{EstimateRFV()}$ function to estimate the RFV of a path using the bottleneck concept, that is to say, we take the lower (more restrictive) individual RFV as the reputation of the path.

On the other hand, as elaborated previously, DSR routes are stored in a Routing Table (RT). Essentially, DSR routes are formed by a list of IP addresses and an expiration time which is typically set to 300 seconds (DEFAULT\_DSR\_ROUTE\_TIMER). Our routing table keeps the same structure as the DSR routing table but we add a tuple $<\text{RFV}, \text{FEEDBACK\_TIMER}>$ to each route. In addition, to store the reputation of individual nodes, we create in cache a new table called Node Reputation Table (NRT) that is formed by tuples: $<\text{IP}, \text{RFV}, \text{ROUTES}, \text{SILENT\_TIMER}, \text{PRIORITY}, \text{CONDEMONS}, \text{STATE}, \text{STATE\_TIMER}>$. The IP address is the KEY field to access/update all the rest of variables of the NRT. The ROUTES variable stores the number of active paths to which a node

---

\(^1\) Small-medium MANETs of at most hundreds of nodes are the target scenario for DSR-like protocols.

\(^2\) Fourth line in Algorithm 2.

\(^3\) This value is assigned in the 8th line of function AddRoute(), Algorithm 3.
currently belongs. If a node does not belong to any route, the SILENT_TIMER is activated. The PRIORITY variable shows the current priority of the node. The CONDEMNS variable shows the number of times that the node has been condemned and the STATE variable shows if a node is currently condemned or not. Finally, the STATE_TIMER is used to manage condemnations. How to use these variables in detail is going to be explained next. The functions in Algorithm 3 describe how the RT and the NRT are updated when a path is added to or removed from these tables.

Algorithm 2: Pseudo-code for updating RFVs related to path \( \mathcal{P} \).

```plaintext
input: A new route \( \mathcal{P} \) discovered by DSR
1  AddRoute(\( \mathcal{P} \)) {
2    CreateEntry(\( RT,\mathcal{P} \)); // Create the entry for the new path
3    foreach node \( \mathcal{K} \in \mathcal{P} \) do // Update or create the NRT entries
4      if Get(\( IP_\mathcal{K},RFV \)) == null then // RFV already exists
5        Set(\( IP_\mathcal{K},\text{ROUTES},1 \)); // ROUTES++
6      else
7        CreateEntry(\( NRT,IP_\mathcal{K} \));
8        Set(\( IP_\mathcal{K},RFV,0.5 \)); // Initial RFV=0.5
9      Set(\( IP_\mathcal{K},\text{ROUTES},1 \));
10     end
11   end
12 }

Algorithm 3: Pseudo-code for processing events related to route management.

```
3.4.2. Routes
The function AddRoute() describes the actions performed when the DSR route discovery routine discovers a new path $P$. In this case, the new path is stored in the Routing Table as a new entry. Then, the list of IPs of the path is used to create the individual entries in the NRT, if they do not previously exist. Also the ROUTES variable of each node is updated; a node may belong to several routes and the ROUTES variable stores the number of active paths to which a node currently belongs. The variable ROUTES is increased or decreased when a route containing the node is created or removed from cache, respectively. The function RemoveRoute() is called when a route is going to be removed due to inactivity or due to a topology change. The ROUTES variable of each node of the route is also decreased. If ROUTES reaches zero, this means that the node does not belong to any active route. Then, the SILENT_TIMER is activated. If the timer expires, the node is considered silent and it is also removed from the NRT. The timer is reset if the node is included again in any path.

```
1 CreateEntry(Table, KEY)
2 AllocateInCache(SizeOf(Key));
3 AddEntry(Table, Key);
4 }
5 AllocateInCache(size)
6 while FreeCacheSize () < size do
7     if SizeOf(NRT)> 0 then RemoveOldest
8         (NRT) else RemoveOldest (RT)
9 end
Algorithm 4: Cache management pseudo-code.
```

Cache management. The number of paths and nodes that can be managed by our reputation system depends on the available cache size of each particular node. Thus, it is required a cache policy that establishes which routes and nodes can be stored in cache. In particular, we use the typical policy of keeping the most recent information, i.e. we place more priority to store routes and nodes more recently used or discovered in front of older ones. This policy is especially suitable for MANETs, where the network topology is highly dynamic. Algorithm 4 describes how to allocate space in cache when an entry is added. Basically, if additional space is necessary, we start removing data from the NRT, and if this is not enough, we remove entries from the RT. We proceed in this order to give priority to routing information.

3.5. Decision making module

This module makes decisions and performs actions considering reputation. This includes actions to: select and change video-streaming routes and actions to enforce cooperation among nodes.

3.5.1. Select and change video-streaming routes
In standard DSR, path selection always chooses the shortest route (minimum number of hops) among the possible paths available to a destination. In RDSR-V, path selection takes into account the observed reputation to select the best available path (the one that provides best QoS). In RDSR-V, path selection works as follows: let us assume that we have $P$ possible paths to a destination node $D$. Then, function SelectPath() of Algorithm 5 is used to select the best forwarding path.

```
1 SelectPath($D$)
2 $P_{known} \Rightarrow null; D_{known} = \infty; RFV_{known} = 0; // To store best path with known RFV
3 $P_{unknown} \Rightarrow null; D_{unknown} = \infty; RFV_{unknown} = 0; // To store best path with unknown RFV
4 foreach path $P$ to destination $D$ of the $P$ available ones do
5     if Get($P$, RFV) == null then // $P$’s reputation is unknown
6         if Get($P$, Distance) < $D_{unknown}$ then
7             $P_{unknown} \Leftarrow P, D_{unknown} = Get(P, Distance); RFV_{unknown} = Get(P, RFV);
8         if Get($P$, Distance) == $D_{unknown} & & EstimateRFV(P) < RFV_{unknown}$ then
9             $P_{unknown} \Leftarrow P, D_{unknown} = Get(P, Distance); RFV_{unknown} = Get(P, RFV);
10      end
11 if Get($P$, RFV)! = null then // $P$’s reputation is known
12     if Get($P$, RFV) < $RFV_{known}$ then
13         $P_{known} \Leftarrow P, D_{known} = Get(P, Distance); RFV_{known} = Get(P, RFV);
14     end
15 end
16 // Criterion to select the best path
17 if $D_{unknown} < D_{known}$ then return $P_{unknown}; // Shortest path preferred
18 if $D_{unknown} == D_{known} & & RFV_{unknown} > RFV_{known}$ then return $P_{unknown}; // If tie, best reputation preferred
19 if $RFV_{known} \leq BAD$ then return $P_{unknown}; // Unknown path is preferred to a bad path
20 return $P_{known}$;
Algorithm 5: Pseudo-code for selecting a path to destination $D$ over $P$ available ones.
As a guideline to understand the `SelectPath()` function, it is worth to remark that we use the path reputation score as the primary selection criterion if this parameter is known. If the reputation score of a path is unknown (for example, because the path has not been used for video-streaming yet), we use the distance to the destination as the selection criterion. On the other hand, when we need to make a selection among paths with known and unknown reputation scores, we use distance as the primary criterion and reputation as the secondary one. This is because we consider that the shortest path is usually the best path if nodes behave cooperatively (which is the behavior expected for most nodes). Summarizing:

- If we know the RFVs of all the paths to a destination, the path with the best reputation score will be selected as the best one.
- If we do not know the RFV of a certain path, but this path is the shortest one among the available paths, we select it as best path (since the shortest path will be probably the best forwarding path assuming most nodes are cooperative).
- In case we have several shortest paths (of the same length), we use their reputation (actual or estimated) to select the best path.
- Finally, if the path with the best known reputation provides a bad QoS (i.e. it has a RFV that is under the BAD reputation threshold), we try to select a new path with unknown reputation from the Routing Table. This is performed although the estimated RFV for the new path is also under the BAD reputation threshold (since the current path provides a bad QoS, we try to get a better QoS with a new path of unknown reputation).

### 3.5.2. Cooperation enforcement

We use the IEEE 802.11e MAC [17] to carry out the actions in reward for cooperative behavior. In IEEE 802.11e, each packet is mapped into an Access Category (AC). There are four AC with different parameters of the EDCA (Enhanced Distributed Channel Access) mechanism: Arbitration Interframe Spacing (AIFS); Minimum Contention Window ($CW_{min}$); Maximum Contention Window ($CW_{max}$); and Transmission Opportunity (TXOP limit). Basically, the smaller these parameters are, the higher the priority is. Using this virtual scheduler we give priority to collaborative nodes over non-collaborative ones. Fig. 2a depicts the original queuing mechanism without rewarding operation. In this case, all the video frames share the same queue whatever the reputation of the end user is. Fig. 2b shows the operation when there is a rewarding mechanism. In this case, whenever a node with high reputation ($RFV \geq \beta$) requires a movie, the source sends I and P frames through AC0 (i.e. the highest AC) and B frames through AC1, so relaying nodes will forward them through their high priority queues. If the node has a medium reputation ($\gamma < RFV < \beta$), the source sends I and P frames through AC1 and B frames through AC2. If the node has a low reputation ($RFV < \gamma$), the source sends I, P and B frames through AC3, i.e. the lowest priority queue. This way, the intrinsic characteristics of the MPEG-2 hierarchical scalable multi-layer encoded video-streaming applications have been taken into account to reward those cooperative nodes seeking to improve the whole service performance. Thus, users realize that it is worthwhile to cooperate and behave properly to gain a priority treatment in the network and thus to perceive a better video quality. Notice that control data (e.g. feedback RR-RTCP packets) are always sent with the highest priority.

In addition, each source has a list of selfish nodes, that is to say, a list of nodes that have a bad reputation because they provide a bad QoS. A node is considered condemned while it is included in this list. The source does not provide the video-streaming service to destination nodes considered selfish. Selfish nodes are neither considered as possible intermediate nodes in video-streaming forwarding paths. Notice that the node could still ask for video service to other sources since each source has its own local set of reputation values that depends on its own experience.

![Access Category mapping in the IEEE 802.11e MAC.](image-url)
Each condemn has associated a condemn time $T_{\text{condemn}}$ that is doubled for each new condemn. The second-chance mechanism works as follows: the node is included again in forwarding paths after it has served the condemn. Also, it can be again a destination node of the video-streaming service, and its reputation qualification is set to the initial value. The number of condemns of a node is stored in a variable called `CONDEMNS`. Since the condemn time is doubled for each new condemn a re-offender node takes longer to be redeemed. On the other hand, the condemn counter is decreased by one if the node has been out of the list of condemned nodes during a period of time that equals its last condemn time.

Finally, if the reputation of a path currently being used by a video-streaming session reaches the POOR reputation threshold, we interpret this fact as an indication of QoS degradation and we react by trying to switch the video-streaming session to another available path.

3.6. Numerical example

Next, we give a numerical example to illustrate the rewarding, punishment and path selecting mechanisms of RDSR-V (using the settings of Section 4.1). Let us suppose that we are in the network shown in Fig. 3. The network is composed of one source node ($S$), two destinations ($D_1$, $D_2$) and four intermediate nodes ($N_1$, $N_2$, $N_3$, $N_4$). $S$ has established two video-streaming sessions: one with node $D_1$ and another one with $D_2$. $S$ has two available paths to $D_1$: path $P_1$ (using intermediate nodes $N_1$ and $N_3$) and path $P_2$ (using intermediate nodes $N_1$ and $N_4$). Likewise, in order to reach $D_2$, the source will send its packets through nodes $N_1$ and $N_2$ (path $P_3$).

We also consider that all the nodes have a reputation value (RFV) of 0.6 at the beginning. Regarding to the paths, there are two routes with a reputation value of 0.9 (path $P_1$ and path $P_3$) and another one with a value equal to 0.7 (path $P_2$). As the reputation scores of both paths $P_1$ and

---

4 Eighth line of Algorithm 6.
5 Lines 18th to 20th of Algorithm 6.
6 Eleventh line of Algorithm 6.
are known, according to the SelectPath() function, the former will be selected as forwarding path by \( \mathcal{S} \) since it has a higher RFV. Let us consider that after a period of time, \( N_3 \) starts behaving selfishly and dropping the 16% of the packets addressed to \( \mathcal{S}_1 \). Consequently, the source will receive RR-RTCP packets from \( \mathcal{S}_1 \) with a FPL = 0.16. Then, the MapReputationSample() will return: \( \text{RFV}_{\text{sample}}^{\mathcal{S}_1} = e^{-14.016} = 0.1 \) (using the mapping function proposed in Section 4.1). Next, the reputation sample will be used locally by \( \mathcal{S} \) to update the accumulated RFVs of \( \mathcal{S}_1, N_1 \) and \( N_3 \). Notice that the reputation value of \( N_1 \) will suffer two updates, one concerning the path \( \mathcal{S}_1 \) and a second one regarding path \( \mathcal{P}_2 \). However, \( N_3 \) will only experiment one update as it only belongs to one path. \( \mathcal{S} \) will update the reputation values of \( N_1 \) and \( N_3 \) (which is now selfish) as follows:

The source will take into account the accumulated RFV of \( N_3 \) (0.6) and the current RFV sample (0.1) to update the accumulated RFV: \( \text{RFV}^{\text{accumulated}}_{N_3} = 0.6 \cdot (1 - 0.25/2) + 0.1 \cdot 0.25/2 = 0.538 \).

For \( N_1 \), it will happen the same for this update (\( \text{RFV}^{\text{accumulated}}_{N_1} = 0.538 \)). But additionally, the accumulated RFV of \( N_1 \) will also be updated by \( \mathcal{P}_3 \) in which packet losses are in this example 0.8%. This percentage is mapped to \( \text{RFV}^{\text{sample}}_{N_1} = 0.538 \cdot (1 - 0.25/2) + 0.9 \cdot 0.25/2 = 0.583 \). Notice that the new accumulated reputation value for \( N_1 \) will be 0.583 which is obviously higher than the current accumulated RFV of \( N_3 \).

At the same time, the RFV of \( \mathcal{P}_3 \) will be also updated. After the reception of the first RR-RTCP packet: \( \text{RFV}^{\text{accumulated}}_{\mathcal{P}_3} = 0.9 \cdot (1 - 0.25) + 0.1 \cdot 0.25 = 0.7 \). Finally, after the reception of four RR-RTCP packets, \( \text{RFV}^{\text{accumulated}}_{\mathcal{P}_3} = 0.353 \). As this value is below the \( \tau \) threshold (\( \tau = e^{-14.006} = 0.432 \)), the source tries to select another path calling SelectPath() function. In this case, the SelectPath() will select \( \mathcal{P}_2 \) because this path has a better RFV. Notice that the RFV of \( N_3 \) will remain at 0.393 and it will be treated with medium priority.

At this point, \( N_1 \) will not receive a condemnation as it is not considered as bad yet (\( \text{RFV}_{N_1} > \gamma \)). Although it is likely to be condemned if continues misbehaving while forwarding video-streams. Finally, suppose that the nodes of path \( \mathcal{P}_2 \) and path \( \mathcal{P}_3 \) are not selfish, the reputation score of \( N_1 \) will increase and the node will finally reach a high priority treatment.

### 3.7. Scalability analysis

RDSR-V is a local reputation system and as such it is based on direct observations only (second-hand reputation exchanges are not used). RDSR-V inherits the main scalability behavior from local reputation systems, which in general have less cost and are more reliable and more efficient than global reputation systems [18]. Reputation systems can be contrasted along several scalability issues [19]:

- Local reputation (traditional systems like OCEAN): each node maintains reputation values of the neighbor nodes (at a single hop) with which there has been an interaction. Typically, a cache of limited size is used and old entries are removed if additional memory is required.
- Local reputation (RDSR-V): each node maintains the reputation values of paths and nodes with which there has been a video-streaming interaction. We also use a cache of limited size in which old entries are removed if additional space is necessary. To give a quantitative figure, let us assume that we have a MANET of 500 nodes and that there are 4 paths available to each destination. Let us consider also that we use variables of 16 bits to store the parameters of tuples of RDSR-V. Then, in the worst case, we just require 17 KB\(^7\) to deploy RDSR-V in each node.

### Local reputation (traditional systems like OCEAN):

- Global reputation: disseminating the reputation information greatly increases the volume of network traffic. In a global reputation system, the number of second-hand messages distributed during each disseminate period is \( O(N^2) \).
- Local reputation (traditional systems like OCEAN): in local reputation systems there is no second-hand information. Therefore, local reputation does not introduce additional network traffic.
- Local reputation (RDSR-V): in RDSR-V, we do not introduce extra network traffic since we use the end-to-end signaling that generates the video-streaming service. The RTCP protocol is the signaling protocol primarily used for quality-of-service monitoring. The goal is to keep signaling overhead as low as possible. Therefore, it was specified in [20] that RTCP can consume 5% of the session bandwidth at most, in which transmission capacity is further divided into two parts, where the RR can consume 75% and the Sender Reports (SR) can consume the remaining 25% of the capacity allowed to the RTCP.

### Overhead computation:

- Global reputation: global reputation systems have to process local observation and they need an additional computational overhead to decide whether to accept or reject second-hand messages. Then, also the reputation table has to be updated accordingly.
- Local reputation (traditional systems like OCEAN): local reputation systems only need to process local observation. However, traditional systems are based on the packet-by-packet observation which implies a significant computation overhead and implementation complexity. In fact, in practice, the extensive use of promiscuous listening may lead to unbearable CPU load.

---

\(^7\) This calculation comes from the following: in RDSR-V the reputation of a node uses a tuple of 7 parameters and one IP address. Each path uses a tuple of 2 parameters. Considering that all the nodes and paths are in the cache \( \Rightarrow (7 \times 16 + 32) \times 500 + 4 \times 500 \times (16 + 2) = 17 \text{ KB} \).
– Local reputation (RDSR-V): once more, we do not introduce extra computational load in RDSR-V since we use the end-to-end signaling that generates the video-streaming service.

3.8. Security analysis

Secure routing protocols use cryptography tools to protect MANETs against message manipulation and they can also provide message authentication. However, these protocols cannot cope with selfish nodes that do not forward received packets (selfish forwarding misbehavior) or that do not participate in route discovery (selfish routing misbehavior). To deal with these problems, cooperation enforcement mechanisms such as RDSR-V are required. Next, we discuss the main security issues related to our protocol and how a secure version of DSR such as ARIADNE [21,22] and a network secure protocol like IPSec [23,24] can be integrated with RDSR-V to achieve a fully secure and reliable video-streaming service.

- **MANET topology.** The MANET topology is maintained by the routing protocol. For example, when a link is broken, nodes receive an error message and they react removing the affected routes. Hence, without the presence of malicious nodes, a path break does not negatively affect the reputation score of its nodes since broken paths are removed from the routing tables. However, as DSR is not a secure routing protocol, a malicious node could attack these error messages and thus, RDSR-V would not be able to distinguish between packets lost due to lack of cooperation and packets lost due to a topology change. In presence of malicious nodes, RDSR-V can use a protocol like ARIADNE to securely discover the MANET topology. In particular, ARIADNE is a protocol that secures most severe attacks against topology discovery: excessive route discovery floods, modifying discovered routes by dropping nodes or altering the node list. ARIADNE also protects against malicious nodes sending bogus/fake route error messages or failing to send route error messages for broken routes.

- **Message authentication and integrity.** Authentication and integrity for signaling messages (topology messages and feedback packets) should be used if security protocols are available: ARIADNE can authenticate DSR topology messages while IPSec can be used to provide end-to-end authentication and integrity of RR-RTCP packets. Hence, a selfish node that modifies feedback packets dishonestly to gain reputation will be detected.

- **Feedback packets dropping.** A timeout will expire if a feedback packet is not received when expected. The timeout will be interpreted as if the corresponding service data packets have been lost and the reputation score of the path will be updated (decreased) accordingly. After some feedback packets are lost, the reputation score of the path will decrease under the threshold $\tau$ and RDSR-V will try to switch the service to a different forwarding path.

- **Context of reputation.** A drawback of generic reputation systems is that reputation can be gained in one service and exploited in another one. As RDSR-V is service specific, reputation gained in a video-streaming session cannot be used in another service. On the other hand, nodes may decide to behave cooperatively or selfishly based on the current context. For instance, a node could behave selfishly for a given set of sources, whereas this node could be collaborative for another set. This would increase the reputation of this node for sources with which it has collaborated and would decrease the reputation score for sources with which it has not collaborated. As a conclusion, the reputation acquired in a particular context cannot be used in another context because in RDSR-V, a global reputation score does not exist, but each source has its own reputation table.

- **Silent nodes.** Silent nodes do not participate in route discovery and, thus, they do not cause damages in ongoing video-streaming sessions. Nevertheless, they do not collaborate by relaying other nodes’ packets. RDSR-V encourages silent nodes to collaborate since it rewards cooperative nodes by providing their packets with a high transmission priority. However, a selfish node’s strategy could be to behave well at first to increase its reputation and then to adopt a “selfish routing misbehavior”. RDSR-V introduces a timeout mechanism to deal with this attack. A silent node cannot keep a high reputation level because the SILENT_TIMER is activated for nodes that do not appear in routes, and when this timeout expires, the reputation score of the node is reset.

- **False accusation.** Since RDSR-V is a local reputation system, false accusation does not exist as such. However, there is a similar drawback because the misbehavior of a single node affects the reputation score of the other nodes in a path. Although this is inherent in our monitoring mechanism, the design of RDSR-V alleviates the effects of this drawback as follows:
  - A collaborative node can be part of different paths. Therefore, a certain source node will update the score of its intermediate nodes using the feedback of all the paths to which intermediate nodes belongs. Hence, the individual score of a cooperative node is increased faster if it belongs to several good paths.
  - To update the accumulated reputation score of an individual node, we divide the aging factor $\alpha$ by the number of nodes of the path $K$. Thus, the accumulated reputation score is only slightly affected by a single bad measurement.
  - RDSR-V tries to select a new path to forward the video-stream if the QoS provided by the current path decreases until the threshold $\tau$. In this case, our mechanism tries to change the current path before it severely affects the reputation of individual nodes. Notice that paths lose reputation faster than nodes (since path reputation update does not consider the factor $K$).
  - The rationale to select paths is distance and reputation. Selecting shortest paths helps the detection of misbehaving nodes and favors the correct reputation qualification of cooperative nodes since the shorter the path is, the smaller $K$ is, which gives a more reliable measure of reputation. As a conclusion, the probability of the false accusation effect is smaller
in shorter paths. Finally, selecting paths with high reputation also provides collaborative nodes with a high score. All these design features allow RDSR-V to reduce the false accusation effect and to isolate real misbehaving nodes.

- **Rumor spreading.** Our system is more resilient to the rumor spreading phenomena than second-hand systems like CORE [3] or CONFIDANT [4] since we do not use recommendations. We also avoid evaluating the trustworthiness of the recommender since the only source of feedback information is the destination of the service which has been authenticated.

4. Evaluation

We have used the Network Simulator ns-2 [25] to adjust the system parameters for a typical DSR MANET scenario⁸ and to evaluate the performance of our proposal RDSR-V. We also compare RDSR-V with OCEAN, which is commonly used as a reference among local reputation systems for MANET.

4.1. Conservative setting for a DSR MANET scenario

We have carried out a set of simulations to tune the aging factor \( \alpha \) of the EWMA filter used to compute the RFV. As a conclusion, we have observed that RDSR-V can be tuned to detect misbehaving nodes rapidly or to have a correct measure of the trustworthiness of the nodes. In the literature, there are other approaches that also weigh past observations and that take different considerations from ours. Some proposals like CORE [3] give more relevance to the past observations (i.e. a low \( \alpha \)) to preserve nodes from being penalized in case of recent sporadic misbehaviors. Other approaches like SAFE [9] give more significance to the recent observations (i.e. a high \( \alpha \)). In this way, a node with good reputation that becomes selfish will harm the system in a shorter period of time. In our case, we have found two thresholds for \( \alpha \) which determine the performance of our proposal in a typical small to medium size MANET scenario. If \( \alpha \leq 0.4 \), RDSR-V identifies new selfish nodes slowly because a higher relevance is given to the historical values than to the instantaneous ones. However, RFV values tend to be more realistic because at the end they are only affected by selfish behavior and not by false accusations. For \( \alpha \leq 0.4 \), it may take a longer period of time to detect selfish nodes, but the detection is more trustworthy. If \( \alpha \geq 0.9 \), a higher weight is given to current qualifications than to previous ones, and RDSR-V is able to detect misbehaving nodes faster. However, the system becomes very sensitive to instant misbehavior of the nodes. Thus, it may happen that we wrongly expel nodes that are not actually selfish. Values of \( \alpha \) between 0.4 and 0.9 could be used in some specific scenarios, in which sources want to detect selfish nodes quickly and with a reasonable degree of trust.

Moreover, there is a trade-off between the aging factor \( \alpha \) and the RR-RTCP packet frequency. If the feedback frequency is low, a single reputation sample will include a lot of temporal information. In this case, \( \alpha \) should be high. In addition, a low feedback frequency implies also low overhead for the system. However, a low feedback frequency makes the system react slowly against misbehaving nodes. If we want to detect misbehavior quickly, we have to raise the feedback frequency. At the same time, this increases the system overhead. In this case, the aging factor should be low to slowly evolve the EWMA filter. When using a frequency for RR-RTCP packets of 5 s (which is a typical value in video-streaming applications), then, \( \alpha = 0.25 \) is a suitable value that makes the system able to detect selfish nodes properly without the interference of the false accusation effect while keeping the overhead low.

On the other hand, we need to define the mapping function between packet losses (FPL) and the reputation score (RFV). It is important to properly map these thresholds to their corresponding numerical values to implement a given rewarding/punishment policy. A high \( \beta \) parameter means that nodes must provide a very good QoS to obtain the reward. Similarly, if \( \gamma \) were too low, nodes would not be punished until they exhibit extreme misbehavior. Taking into account these considerations, an assignment of a 25% for EXCELLENT/GOOD nodes (\( \beta = 0.75 \)) and another 25% for BAD nodes (\( \gamma = 0.25 \)) provides a conservative setting for rewards and punishments.

Finally, since losses (FPL) and perceived video quality (MOS) are approximately related by an exponential function, we can apply a simple exponential regression to obtain the mapping function using the following points (RFV, FPL): \((0, \infty), (0.25, 0.1), (0.75, 0.02) \) and \((1, 0)\). Finally, we obtain the following expression: \( RFV = e^{-0.14 \cdot FPL} \). This relationship is shown in Fig. 4. Notice that this is a possible setting of the mapping function but, depending on different rewarding and punishment policies, different values of \( \beta \) and \( \gamma \) might be chosen.

4.2. Performance evaluation

Table 2 summarizes the main simulation settings.

First of all, we evaluate RDSR-V performance in a scenario in which we introduce selfish nodes that do not forward a fraction of the received packets. Fig. 5 shows the behavior of the RDSR-V protocol when one selfish node is included in the forwarding path. In this first scenario, nodes are static (mobility will be included later). The network size is 400 × 400 m. As observed in the figure, the selfish node is detected by RDSR-V and condemned at (A). Then, the source avoids the paths that include this condemned node during a punishment period \( T_{\text{condemn}} \). Consequently, losses decrease and the QoS improves. At (B) the selfish node has served its sentence and it is re-inserted. However, it insists on misbehaving (C) and losses grow until the selfish node is detected again (D). Then, it is included again in the list of faulty nodes, this time during \( 2T_{\text{condemn}} \). At (E) the node is re-inserted again. Unfortunately, it persists in misbehaving (F) until finally, at (G) it is definitively
excluded since it has accumulated the maximum number of allowed condemns.

On the other hand, DSR does not include any cooperation enforcement mechanism, thus it continuously suffers the misbehavior of the selfish node. Notice that with pure DSR the FPL is about 15%, which corresponds to about 24 dB of PSNR according to Table 1. This is equivalent to a bad video quality. With RDSR-V the user will notice a significant improvement in the video quality with PSNR = 27 dB at (B), PSNR = 28 dB at (E) and finally will reach PSNR = 30 dB (high quality) after definitive exclusion of the selfish node. This happens in 4 min out of 117 min of total duration of the video stream.

Next, we show the effect of the actions in reward performed by our system. For this purpose, we analyze the RDSR-V operation with and without rewarding actions. RDSR-V without rewarding actions includes only the mechanism for forwarding path selection and the actions of punishment, while RDSR-V with rewarding actions considers the complete set of actions of our the proposal including the queuing mechanism sketched in Fig. 2b. Fig. 6a shows the PSNR measured at the destination node regarding the percentage node selfishness (percentage of packets not forwarded by the selfish node). It can be clearly appreciated that RDSR-V with rewarding actions offers higher PSNR than RDSR-V without rewarding actions and than DSR. Fig. 6b depicts the PSNR of a video-stream received by users with different RFV (high, medium and low). As RDSR-V rewards cooperative users, we can observe that users with high reputation perceive the video-stream with high quality too. This way, users realize that it is worth cooperating.

**Table 2**

Simulation settings.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes</td>
<td>20</td>
</tr>
<tr>
<td>Number of video connections</td>
<td>1 or 2</td>
</tr>
<tr>
<td>Transmission range</td>
<td>80 m</td>
</tr>
<tr>
<td>Mobility pattern</td>
<td>Random waypoint</td>
</tr>
<tr>
<td>Mac TX rate</td>
<td>11 Mbps</td>
</tr>
<tr>
<td>Video sequence</td>
<td>MPEG-2</td>
</tr>
<tr>
<td>Video format</td>
<td>YUV, QCIF, PAL</td>
</tr>
<tr>
<td>Nominal bandwidth</td>
<td>11 Mbps</td>
</tr>
<tr>
<td>$CW_{min}[AC0]$, $CW_{max}[AC0]$, AIFS[AC0]</td>
<td>7, 15, 2</td>
</tr>
<tr>
<td>$CW_{min}[AC1]$, $CW_{max}[AC1]$, AIFS[AC1]</td>
<td>15, 31, 2</td>
</tr>
<tr>
<td>$CW_{min}[AC2]$, $CW_{max}[AC2]$, AIFS[AC2]</td>
<td>31, 1023, 3</td>
</tr>
<tr>
<td>$CW_{min}[AC3]$, $CW_{max}[AC3]$, AIFS[AC3]</td>
<td>31, 1023, 7</td>
</tr>
<tr>
<td>Mapping function</td>
<td>$e^{-14/RV}$</td>
</tr>
<tr>
<td>Initial RFV</td>
<td>0.5</td>
</tr>
<tr>
<td>Aging factor ($\alpha$)</td>
<td>0.25</td>
</tr>
<tr>
<td>RR-RTCP frequency</td>
<td>5 s</td>
</tr>
<tr>
<td>Punishment period ($T$) for the first condemn (counter = 1)</td>
<td>25 s</td>
</tr>
<tr>
<td>Maximum number of condemns allowed</td>
<td>3</td>
</tr>
<tr>
<td>Cache table size</td>
<td>32 KB</td>
</tr>
<tr>
<td>Simulation time</td>
<td>500 s</td>
</tr>
</tbody>
</table>
The following simulations show the effect of mobility in the performance of our proposal. In Fig. 7 the percentage of packet losses throughout time is presented. In this case, the scenario is composed by 20 nodes in a 100 x 100 m area. The average speed of the nodes is 2 m/s. Also, there is only one selfish node which drops 15% of the packets. Fig. 7a depicts how RDSR-V considers that the path being used has low reputation at 25 s. Consequently, the source changes to another path during 25–50 s. The source marks that selfish node as condemned and therefore it is not going to be used as a forwarding node in the paths established from that source. At \( t = 50 \) s, the condemn finishes and that node is re-inserted to be used again. Notice that DSR still uses that path during the period 25–50 s, as it does not include any mechanism to detect selfish nodes. At \( t = 50 \) s the source re-selects a path including the selfish node again, so there are losses again. At \( t = 90 \) s the path is again considered to have low reputation and it is discarded by the source. Now the punishment period doubles to 50 s. After that moment, the node is again available as forwarding node and the source selects it at \( t = 155 \) s. The same happens again at \( t = 182 \), and as the node has accumulated the maximum number of condemnations it is definitively expelled from the network for that source. That source is not going to use paths involving that node for the rest of the simulation, whereas DSR is not able to distinguish it and will continue suffering losses. Fig. 7b shows the percentage of packet losses throughout time in the same scenario than in Fig. 7a, but now there are two selfish nodes that also drop 15% of the packets each. In this situation, RDSR-V takes longer to completely detect the two selfish nodes, not being detected until \( t = 900 \) s. The time needed to detect all selfish nodes will depend on the number of selfish nodes, their percentage of misbehaving and their speed. Again, DSR continues suffering losses until the end of the simulation, as it is not able to detect selfish nodes.

### 4.3. Comparison with OCEAN

In [2] the authors compare OCEAN with some second-hand reputation systems and conclude that first-hand systems obtain similar results in terms of network throughput while being much simpler. As RDSR-V is also a first-hand system, this section is exclusively devoted to compare RDSR-V with OCEAN.

Firstly, we present the main similarities between both proposals:

(a) Both proposals have been designed taking the Dynamic Source Routing (DSR) protocol as the routing engine to find forwarding paths.

(b) Neither RDSR-V nor OCEAN deal with security issues like node authentication, securing routes, message encryption or collusion of nodes. These issues are addressed by secure routing protocols such as
OCEAN and RDSR-V exclusively address the cooperation enforcement mechanism which in fact can be used together with a secure routing protocol.

(c) Both proposals include a chance mechanism intended to allow nodes previously considered misleading to become part of the network again.

(d) OCEAN deals with silent nodes taking into account the past forwarding performance. Similarly, RDSR-V performs rewarding actions to nodes with high reputation to encourage silent nodes to cooperate and gain a priority treatment in the IEEE 802.11e queuing mechanism.

On the other hand, the main differences between OCEAN and RDSR-V are:

(a) OCEAN is general-purpose, whereas RDSR-V is specifically designed for the MPEG-2 video-streaming service over MANET.

(b) In OCEAN each node maintains counters called chip-counts for each neighbor. A node earns chips upon forwarding a packet for a node, and loses chips upon asked by a neighbor to forward a packet. This way, when a node decides to forward a request, it checks its chipcount for that node. If it falls below a threshold, the node denies the request. The problem is that this mechanism may result in sudden link breakages even when the destination node is a cooperative node. This makes OCEAN not very suitable for video-streaming services. On the contrary, RDSR-V can provide the service with temporary QoS reductions if necessary, but avoiding abrupt service breakages.

(c) OCEAN considers two types of misbehaving nodes: misleading nodes (also called selfish forwarding nodes) and selfish nodes (also called silent nodes). Our proposal is mainly focused on selfish forwarding nodes because they cause damage in already established paths, which is the worse situation for the video-streaming service. On the other hand, silent nodes do not participate in route discovery and, thus, they do not cause damages in ongoing video-streaming sessions (although obviously, they do not collaborate in forwarding data for others). However, RDSR-V also deals with silent nodes because if a node becomes silent, it will not appear in the routes of source nodes, and it will be considered as a node with normal priority after a timeout.

(d) In OCEAN, reputation is monitored using a watchdog mechanism. That is to say, when a source sends a packet, it keeps listening the wireless channel. If the source does not hear the packet within a time-out, it registers a negative event for the neighbor. Conversely, RDSR-V does not use promiscuous monitoring but end-to-end monitoring. OCEAN obtains local reputation measurements, whereas RDSR-V achieves a full end-to-end design without using extra overhead or high computational resources.

(e) In both proposals nodes maintain their own faulty list. In OCEAN, when the rating of a node falls below a certain threshold, the node is added to a faulty list. This list is carried in the route request messages of the DSR routing protocol and each forwarding node appends its own list. The problem is that the length of the route discover packet header might become quite big. On the contrary, in RDSR-V, nodes maintain locally their own faulty list, which is never broadcasted.

(f) In OCEAN, a route is considered good or bad depending on the reputation of the next hop. In RDSR-V, rates of paths depend on feedback information regarding the whole end-to-end video quality which provides better decisions from the point of view of the final user.

(g) OCEAN uses the IEEE 802.11 DCF (Distributed Coordination Function), whereas RDSR-V uses the IEEE 802.11e EDCF (Enhanced DCF) which is able to provide QoS. This feature is especially interesting for multimedia services such as video-streaming services.

Finally, we have conducted a series of simulations to compare the performance of RDSR-V with OCEAN. To do this, we use an open source implementation of OCEAN over ns-2 [28].

Fig. 8 shows the percentage of packet losses for RDSR-V, OCEAN and DSR. The performance is analyzed varying the number of selfish nodes in the network. The network area is 100 × 100 m and the simulation time is 300 s. We can

![Fig. 8. Packet losses varying the number of malicious nodes.](image-url)
observe that OCEAN outperforms DSR except when the percentage of malicious nodes is higher than 70%. RDSR-V, in turn, outperforms OCEAN and DSR even in the presence of a great number of malicious nodes.

Fig. 9a shows the percentage of packet losses throughout time. In this simulation, there are four selfish nodes and there is an interfering traffic that produces a constant bit rate (CBR) of 500 Kbps. Again, we can observe that RDSR-V outperforms OCEAN and DSR.

Fig. 9b depicts the percentage of I-frame losses for the same simulation. With RDSR-V, we obtain generic packet losses of about 14% but only about a 10% are I-frame losses. In the case of DSR and OCEAN, both have the same percentage of packet losses and I-frame losses (26% and 19% respectively). These results prove that RDSR-V is a service aware mechanism because it protects I-frames which are the frames that carry the most important data for the video decoder process.

5. Conclusions

Reputation-based models are efficient mechanisms to enforce cooperation among MANET nodes. In our opinion, these reputation models can be highly improved by considering the specific characteristics of a particular service. The high amount of traffic generated by typical video applications can also justify the use of a service-specific reputation system. For these reasons, we have designed RDSR-V, a QoS-aware reputation-based routing protocol for video-streaming applications over MANETs. RDSR-V is a DSR-based variant that chooses the best path to forward each video-stream applying cross-layering between routing and service layers to obtain the best QoS. Punishment and rewarding mechanisms are also included based on the specific characteristics of the encoded video-streaming. It is worth remarking that RDSR-V is straightforward to implement and completely distributed. RDSR-V introduces no additional signaling overhead as it uses existing RTCP packets to update the forwarding reputation and it does not require reputation data exchanging. Finally, we have simulated the system in ns-2. A performance evaluation of the proposal has been carried out to show that RDSR-V clearly outperforms standard DSR and OCEAN in presence of non-cooperative nodes. Furthermore, RDSR-V improves the overall QoS of the video-streaming service because it is a service aware mechanism. In this sense, RDSR-V is designed to protect the I-frames which are the frames that convey most important information for the user’s video decoder. As a consequence, the user perception of the video quality improves noticeably.

Acknowledgments

We would like to thank the valuable comments and suggestions made by the anonymous reviewers to the improvement of this article. We also thank the contributions to this article made by the Ph.D. students Carlos Gañán and Javier Parra-Arnau. This work is funded by the Spanish Ministry of Science and Education under the projects CONSOLIDER-ARES (CSD2007-00004), SECCONET (TSI2005-07293-C02-01), ITACA (TSI2005-65393-C02-02) and P2PSEC (TEC2008-06663-C03-01).

References


Oscar Esparza received his M.S. degree in Telecommunication Engineering in the Technical University of Catalonia (UPC) in 1999. In the same year he joined the AUNA Switching Engineering Department. Since 2001 he works in the Department of Telematics Engineering of the UPC as Associate Professor. In 2004, he received the Ph.D. degree in Mobile Agent Security and Network Security. He can be contacted at Technical University of Catalonia Jordi Girona 1-3. Campus Nord, Mod C3, UPC. 08034 Barcelona. Spain. E-mail http://osc.ar.esparza@entel.upc.edu.

Mónica Aguilar-Igartua received the Telecommunication Engineering degree and the Ph.D. in Telecommunications Engineering in 1995 and 2000 respectively, both from the Polytechnic University of Catalonia (UPC), Barcelona, Spain. Currently, she is working on service-level QoS management frameworks, focusing on dynamic procedures to compute Service Level Agreements (SLA) parameters. Her research interests also include modelling and performance evaluation of broadband network nodes, as well as of multimedia servers and video streaming. Her current research activity is focused on the provision and management of QoS over Mobile Ad Hoc Networks (MANETs). She is an associate professor in the Department of Telematics Engineering at the ETSETB. In 1995, she joined the High-Speed Systems and Networks Design and Evaluation Group, and in 2001 she joined the Telematics Services Research Group (SerTel) at the Department of Telematics Engineering of the UPC.

Victor Carrascal received his Degree in Telecommunication Engineering from UPC in 2003. During the period 2002–2004 he was working at Tradia at the Telecontrol Department, doing some developments and demonstrations of the capabilities of the data transmission of TETRA over IP. In 2004 he joined the “Distributed Multimedia Services with QoS (Quality of Service) Group” in the Telematics Services Research Group at the Department of Telematics Engineering of the UPC. He is interested in research about QoS over video streaming on routing ad-hoc protocols, including rate control protocols.

Jordi Forne received the M.S. degree in telecommunications engineering from the Technical University of Catalonia (UPC) in 1992 and the Ph.D. degree in 1997. In 1991, he joined the Cryptography and Network Security Group at the Department of Applied Mathematics and Telematics. Currently, he is an associate professor with the Information Security Group at the Department of Telematics Engineering of the UPC. His research interests span a number of subfields within information security and privacy. He can be contacted at Technical University of Catalonia Jordi Girona 1-3. Campus Nord, Mod C3, UPC. 08034 Barcelona. Spain. E-mail http://jorde@entel.upc.edu.

Jose L. Muñoz received the M.S. Degree in Telecommunication Engineering in the Technical University of Catalonia (UPC) in 1999. In the same year he joined the AUNA Switching Engineering Department. Since 2000 he works in the Department of Telematics Engineering of the UPC, currently as Associate Professor. In 2003, he received the Ph.D. Degree in Network Security. He can be contacted at Technical University of Catalonia Jordi Girona 1-3. Campus Nord, Mod C3, UPC. 08034 Barcelona. Spain. http://jose.munoz@entel.upc.edu.