An Analysis of the Ninux Wireless Community Network

Leonardo Maccari
DISI – University of Trento, Italy
Email: leonardo.maccari@disi.unitn.it

***Abstract*—Wireless community networks are wireless mesh networks created and managed by a local community with mainly two main goals: sharing Internet connection and supporting local services. They are an emerging trend in Europe and have received the attention of many researchers, since they are large accessible deployments of distributed wireless networks. This paper illustrates the features of the Rome-based Ninux community network, the largest in Italy, and studies some interesting features it offers related to routing metrics and centrality metrics.**

***Index Terms*—Community networks, Mesh networks, Routing metrics, Centrality metrics.**

I. Introduction

A Wireless Community Network (WCN) is a wireless mesh network created with a bottom-up approach. In a WCN a local group of users creates an alternative, self-managed, community-based networking infrastructure that is normally used for two main purposes: allowing inter-user interactions (messaging, talking, sharing etc.), and bringing Internet connectivity where it is not present. Nowadays the market offers low cost equipment that can be used to set-up wireless links over a distance up to tens of kilometers. Using a multi-hop approach a few Internet connections can be shared over a very large area.

WCNs are flourishing. In many European cities WCNs made of hundreds of nodes are present: in Athens is present a WCN made of more than 2000 nodes, while in Spain, the Guifi network is a composition of WCNs that accounts for more than 21.000 nodes and grows at an incredible pace of 50 per week. Tens of thousands of nodes connecting tens of thousands of individuals, families, associations, public offices with a non-profit approach and community-based organization.

After an initial interest in their early steps [1], WCNs have lately re-attracted the attention of academia [2], [3] and research funding [4]. The aim of this paper is to analyze the largest Italian WCN, the Ninux network (www.ninux.org). Ninux was started in Rome in the year 2001 and today is made of about 200 nodes spread all over Italy. The Rome-based Ninux community is the largest one while other 'Ninux-islands' are growing in other regions. Each one develops independently both in terms of network connectivity and in terms of organization. From now on, for simplicity the Rome-based island will be simply referred to as 'Ninux'.

The first contribution of this paper is the description of the Ninux network, Sec. II contains a brief description of its building blocks (hardware and software) while Sec. IV and V contain a more detailed analysis of the its topology. Ninux uses the Optimized Link-State Routing protocol (OLSR) [5], with the Expected Transmission Count metric (ETX) [6], which is reviewed in Sec. III. The current design of the ETX metric is hardly compatible with the use of Multipoint Relays (MPR), another important feature of OLSR. Sec. VI investigates the convenience of using ETX compared to simple hop-count metrics on the Ninux topology.

Finally the second main contribution of this paper is the analysis of group centrality metrics on the Ninux topology. Betweenness and closeness centrality are two different criteria to identify if a node is to be considered in the core or in the periphery of the network. Their definition can be easily extended to group of nodes. Sec. VII will show that group centrality metrics can help understanding important features of the network.

II. Description of the Ninux network

The nodes of the Ninux network don't have a single hardware and software configuration. Each participant is responsible for his own node and decides the configuration that best fits his needs. The community gives guidelines to guarantee the compatibility between nodes.

A. Hardware specifications

The majority of the nodes use one of two solutions: boxed indoor equipment or commercial devices for outdoor use.

In the first case, COTS access points, such as the TP-Link TL-wr841nd are modified using outdoor antennas, powered over Ethernet and enclosed in a plastic box. This is the easiest solution to deploy since it is low cost and it relies on omnidirectional antennas that do not need to be aligned, but features low ranges and low throughput. In the second case devices such as the Ubiquiti *nanostation* (see www.ubnt.com/airmax) are used. They have embedded panel antennas (beamwidth of 40 degrees) or even parabolic antennas (beamwidth of 10 degrees). This second solution needs more expertise to be installed but guarantees longer ranges and higher bit rates than the omnidirectional one. When using directional antennas, it is often necessary to install more than one device in the same building, in order to offer connectivity to neighbor nodes. In this case a set of independent devices are singularly powered and connected with an Ethernet switch forming a so-called *super-node*. With a composition of such devices it is possible to cover a large horizontal angle while keeping the advantages of long ranges and high bit rates. The technology used is a mixture of IEEE 802.11g/a/n standards depending on the availability of radios and on the interference condition in the installation place. The preference falls on 802.11n devices in order to achieve higher bit rates. All the devices support some automated configuration to limit the transmission power according to regional laws.

B. Software specifications

Independently of the hardware used, the original operating system is substituted with a version of the OpenWRT GNU/Linux distribution configured with the needed software. Each device in a super-node is assigned an IPv4 and an IPv6 address and two instances of the OLSR routing protocol are always running. This makes it really easy to extend a node adding further devices but increases the number of necessary IPs and the overall routing signalling. For this reason another configuration for the super-nodes has been tested: the switch is substituted with a router, it is assigned one IPv4 and one IPv6 address, and radio devices are bridged to the router using independent VLANs. This configuration has several advantages: only one IP is used for each super-node, signalling is reduced since only the router runs OLSR, and the original firmware can be left on the devices as long as it supports bridging and VLAN tagging. It is indeed more complex to initially set-up since it needs the set-up of a managed switch.

Some comprehension of the OLSR protocol is needed to better understand the findings of this paper. Since OLSR is largely described in the literature [5] the Sec. III will give just a brief description of its principles.

III. Review of the OLSR protocol

Let's first introduce some notation. In a network N each node i has a set of one-hop neighbors $N_1(i)$ reachable via only one hop. It also has a set of two-hop neighbors $N_2(i)$ reachable via two hops (note that by construction $N_1(i) \cap N_2(i) = \emptyset$). In OLSR each node periodically sends an HELLO message that is used to build the knowledge about the 1-hop neighborhood. Furthermore, each node inserts in its HELLO messages the IP address of any of its 1-hop neighbors, so that at steady state each node will have a full knowledge of its 2-hop neighborhood. Each node elects among the nodes of its neighborhood a set or Multipoint Relays (MPR). The MPR set $M(j)$ of a node j is an arbitrary subset of its symmetric 1-hop neighborhood $N_1(j)$ which satisfies the following condition: every node in the 2-hop neighborhood $N_2(j)$ of j must have at least a symmetric link towards a node in $M(j)$. Thus if $i \in M(j)$ then i "*covers*" some of the nodes in $N_2(j)$ and the whole $M(j)$ covers the complete $N_2(j)$. Once j has selected its MPRs it will communicate them that it has become one of their MPR *selectors*. Each MPR starts behaving as follows:

- It periodically generates Topology Control (TC) messages. A TC contains the list of the IP addresses of its MPR selectors
- It rebroadcasts the TCs received from its selectors.

TC messages contain an approximation of the local topology around an MPR and are received by all the nodes. In this way each node has enough information to compute the shortest path routes to any other node. Since only MPR nodes retransmit the TCs, TCs will reach all the nodes in the network using a fraction of retransmissions compared to plain flooding. Minimizing the size of each $M(j)$ is thus important to minimize the union of all the $M(j)$, that in turns minimizes the number of generated and forwarded TC messages.

In Ninux, OLSR is configured to use the ETX metric to estimate the quality of each link. ETX estimates the expected number of times a packet needs to be transmitted to reach a neighbor, taking into account losses due to collisions and interference. Since the timer used between every HELLO message is known, each node j estimates the number e of HELLO messages that it is expected to receive from one neighbor i in a certain time window. Node j will count the number r of HELLO messages actually received from i and will specify in its HELLO messages the ratio r/e. The value r/e is called the link quality (LQ) of the link from j to i. Node i will do the same, so that node j knows both the LQ value and the reverse value (neighbor link quality, NLQ). Since any unicast transmission in 802.11 requires a data frame and an ACK in the opposite direction, the probability of successfully sending a packet is approximated by $LQ \times NLQ$. The average number of frames needed to successfully send a packet is thus estimated as $ETX = \frac{1}{LQ \times NLQ}$. ETX is used as a link weight to compute shortest path routes using Dijkstra's algorithm.

Every node j has a perfect knowledge of its one-hop neighborhood and of the link weights to reach all the nodes in $N_2(j)$. Since TC messages contain the ETX metric only for the links between an MPR and its selectors, j has only an approximated knowledge of the rest of the network. MPRs will *hide* the presence of some links, which can influence the computation of the quality of the routes. In order to take this into account, in the OLSRd daemon (the GNU/Linux based implementation of the OLSR protocol) the default choice of $M(j)$ has been changed. Node j computes $M(j)$ with the aim of maximizing the link quality to every node in $N_2(j)$ and not with the aim of minimizing its size. This has two consequences, the first is that the number of MPRs is increased, the second is that the choice of the MPRs is much more unstable. In fact, the choice of $M(j)$ does not depend on the topology in the 2-hop neighborhood, which is expected to be stable, but with the link quality, that can fluctuate due to traffic load and interference. Having continuous re-computations of the MPR sets will trigger continuous changes in the routes and, in the worst case, temporary loops. For this and other reason, the Ninux community network, as other networks, decided to force every node to be an MPR. In this way a perfect knowledge of the quality of every link is traded with a larger production of signalling messages.

The last feature of OLSR that is useful to recall is the possibility of including foreign network addresses in HNA messages. If a node is attached to another private network it can use Host and Network Association (HNA) messages to expose the existence of this subnet. Every other node will add routes to reach the subnets of those networks. A node advertising the `0.0.0.0/0` network is an Internet gateway.

IV. Methodology

Since every node in Ninux is an MPR, every node has a precise knowledge of the ETX metric for every link in the network. OLSRd integrates two plug-ins that can be used to remotely access all the topological information known to a node. This plug-in is used in the Ninux community to feed a database of existing nodes, that is publicly available on the Ninux website. Using the information contained in the database it has been possible to gather information on the number of nodes, their links, their position and the quality of each link.

As said, OLSR is used also on the wired links that connect two devices on the same super-node. Their ETX is constant and their maximum bit rate is much higher than the bit rate of wireless links. In order to have an unbiased evaluation of the wireless topology, in our analysis the devices that equip a super-node have been collapsed to only one logical node. The co-located devices were identified using the information stored in the SQL database with the help of the Ninux community. In practice, all the super-nodes are considered as if they had a bridged configuration (as described in section II-A).

The whole topology has been dumped once every 30 minutes for a whole day, producing 48 snapshots of the network graph. For each dump only the largest connected component has been used in order to filter out a few isolated nodes. The graph has been exported in a standard format and has been processed with the NetworkX Python library, a powerful software framework for graph analysis. A 24-hours time frame is not enough to describe the long-term evolution of the network, but it is enough to make an analysis of the main features of the network topology. In particular, it was possible to notice that the network graph is pretty stable and the variation of the ETX metric has a small dynamic during the period of observation. Having verified that the changes from one snapshot to another are pretty small, for the sake of clarity in the next section is reported only the analysis performed on a single snapshot.

This paper focuses on the topological properties of the graph, no data has been extracted on the generated traffic which will be the subject of future works.

V. The Ninux Network Topology

The Ninux network is made of 112 nodes and 136 links. The average shortest path has a weight of 6.5 (considering the ETX metric) and 5.9 hops. In the considered period the average measured ETX per link was 1.19. Only 8 links over 136 had an average ETX larger than 2 and they all were connecting a leaf node to the network. The average standard deviation of the ETX metric on the same link computed on all the 48 samples is less than 8% of the average. This shows that a particular care and expertise has been used to set-up the Ninux network, since the majority of the links have a high quality and present low fluctuations.

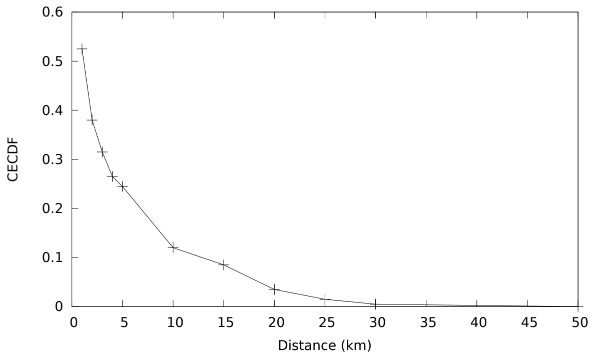

Figure 1. Link length CECDF

The average link length is 3.96 kilometers, the large majority of the links are shorter than 5 km as reported in Fig. 1, the longest link reaches almost 50 km.

50 nodes in the graph have only one link, 62 have two of more links, only two nodes have more than 7 links (they have 10). Fig. 2 shows the Complementary Empirical Cumulative Distribution Function of the degree of the nodes. A log-log scale is used, together with a linear fit function. Even if there are not enough samples to make an accurate estimation of the distribution, the points seem to fit quite well a linear trend at least for the first 7 samples. This would confirm the results of previous works on community networks [2].

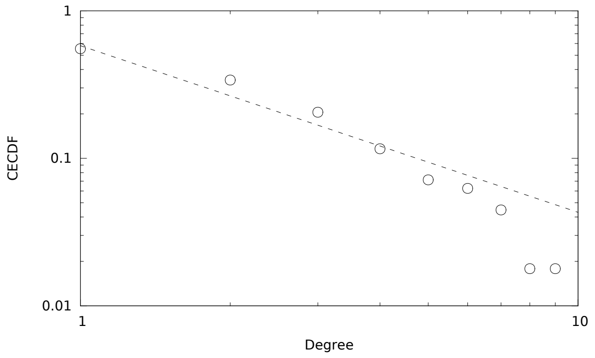

Figure 2. The degree distribution (log-log scale)

Lastly, Fig. 3 reports the distribution of weights of the path from any node to the closest Internet gateway (excluding the gateway themselves). The majority of the nodes can reach the Internet on a path with a reasonable cost.

VI. Impact of the ETX metric

The purpose of this section is to study the impact of the ETX metric on the Ninux network. As said, mixing ETX metric and MPRs is problematic in OLSRd and for this reason in Ninux all the nodes are MPRs. This section investigates if the improvements given by ETX justify the overhead introduced by renouncing to MPR selection.

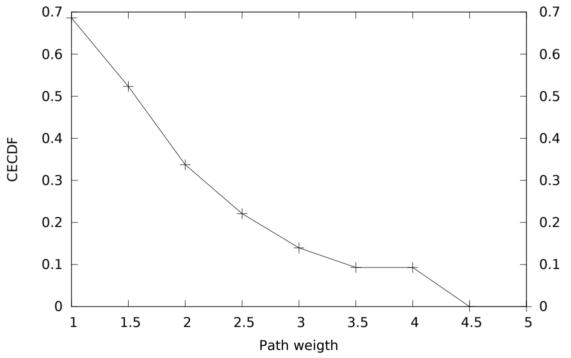

Figure 3. The distribution of the path weight to the closest Internet gateway

A. Notation

Let's introduce some more useful notation:

$G_w = (N, L_w)$ is the undirected network graph made of a set of nodes N and a set of links L_w. Each node corresponds to a Ninux node and each link $l_{ij} \in L_w$ is identified by the two nodes (i, j) that it interconnects. Link l_{ij} is assigned a weight w_{ij} corresponding to its ETX value. Only one link is present between two nodes.

$G_u = (N, L_u)$ is the undirected graph made of the same set of nodes and links as G_w in which each $l'_{ij} \in L_u$ is assigned the weight 1.

$W(l)$ is a function that maps a link l_{ij} to a weight. If $l = l_{ij} \in L_w$ it simply returns w_{ij}, if $l = l'_{ij} \in L_u$ it maps l'_{ij} to the weight of the correspondent link $l_{ij} \in L_w$, so $w_{ij} = W(l_{ij}) = W(l'_{ij})$. This function is well-defined since there is a one-to-one relation between each element of L_u and one element of L_w.

Q is the set of all couples (i, j) where $i \in N, j \in N, i \neq j$.

$P_w(i, j) = \{l_{ik}, l_{kn}...l_{mj}\}$ is the shortest path between i and j computed on G_w, using Dijkstra's algorithm.

$P_u(i, j) = \{l'_{ik}, l'_{kn}...l'_{mj}\}$ is the shortest paths between i and j computed on G_u. It is likely to have a set $\hat{P}_u(i, j)$ of shortest paths with the same length, and the choice of $P_u(i, j)$ in $\hat{P}_u(i, j)$ is purely implementation-dependent (here the first one is considered). We say that $P_u(i, j) = P_w(i, j)$ if the sequence of visited nodes is the same. For the sake of readability, if $\exists\ P_u(i, j) \in \hat{P}_u(i, j) \mid P_u(i, j) = P_w(i, j)$ we simply write that $P_w(i, j) \in \hat{P}_u(i, j)$.

$C(P(i, j))$ is the cost of a path $P(i, j)$ defined as $C(P(i, j)) = \sum_{l \in P(i,j)}(W(l))$. Note that $C(P_u(i, j))$ is not simply the length of $P_u(i, j)$, it is the cost of $P_u(i, j)$ computed considering the weights of the links from G_w.

Three metrics that are useful to evaluate the impact of ETX are:

$r(i, j)$: it is equal to 1 if $P_w(i, j) \in \hat{P}_u(i, j)$, 0 otherwise. r is $r(i, j)$ averaged over every couple $(i, j) \in Q$ and normalized to 1.

$s(i, j)$: The size of $\hat{P}_u(i, j)$, that is the number of redundant shortest paths in the non weighted graph. s is $s(i, j)$ averaged over every couple $(i, j) \in Q$

$c(i, j)$: The average $C(P(i, j))$ for $P(i, j) \in \hat{P}_u(i, j)$. c is $c(i, j)$ averaged over every couple $(i, j) \in Q$. More formally:

$$c = \frac{\sum_{(i,j) \in Q}(\sum_{P \in \hat{P}_u(i,j)} C(P))}{\sum_{(i,j) \in Q} s(i, j)} \qquad (1)$$

The approach used in the analysis is the following: for each $(i, j) \in Q$, $\hat{P}_u(i, j)$ and $P_w(i, j)$ are computed on G_u and G_w respectively. If $P_w(i, j) \in \hat{P}_u(i, j)$ then r is incremented. For each $P \in \hat{P}_u(i, j)$, $c(i, j)$ and $s(i, j)$ are computed. At the end all the metrics are averaged and/or normalized.

$P_w(i, j)$ and $P_u(i, j)$ are the best routes that OLSR would compute with and without ETX. Metric r gives an estimation of how often, even when ETX is not used, the best route $P_w(i, j)$ is in $\hat{P}_u(i, j)$, so there is a chance that the best route chosen using ETX would be chosen even without using ETX (i.e. $P_w(i, j) = P_u(i, j)$). When $r = 1$ the probability that $P_w(i, j) = P_u(i, j)$ is given by $1/s(i, j)$. Now consider a couple (i, j) for which $P_w(i, j) \in \hat{P}_u(i, j)$, $s(i, j) = 2$, and $P_w(i, j) \neq P_u(i, j)$. It is perfectly feasible that $C(P_w(i, j))$ largely differs from $C(P_u(i, j))$. Metric c is used to compare $P_w(i, j)$ with the average cost of all the routes in $P_u(i, j)$.

The most evident defect of the hop-count metric is that it uses the shortest route (in terms of hops) even if it includes links that have a very low quality. Nevertheless, even when ETX is not globally used, it is implicitly computed by a node to evaluate the stability of the links to his neighbors. Every couple measures the number of missed `HELLO` messages and uses an hysteresis function to purge unreliable links. To take this into account, the analysis has been performed setting a threshold t_{etx} and producing two graphs $G_w(t_{etx})$ and $G_u(t_{etx})$ in which links from G_w that have an ETX value larger than t_{etx} have been purged. After the purging only the nodes forming the main connected component are considered.

B. Results

The graph in Fig. 4 reports the number of nodes, the number of links and the number of non-leaf nodes in $G_u(t_{etx})$ varying the value of t_{etx}. Since the average quality of the links is quite high, quickly the main connected component nears the majority of the nodes of the original graph (they are the same after the value of 2.2). When t_{etx} is set to 4 not only all the nodes but also all the links are included in the main connected component.

Fig. 5 depicts r and s and shows a very interesting behaviour. First of all the value of s is very low. As a trend it grows with t_{etx}, and it is always lower than 1.3. This means that there is not that much of redundancy for the shortest paths computed with simple hop-count metric. The second curve shows r, that is exactly one when t_{etx} is set to 1 (as expected) but stays always above 0.68 and approaches 1 when the full topology is considered. This is a very interesting result, since it shows that in the majority of the cases $P_w(i, j) \in \hat{P}_u(i, j)$. Those two results indicates that if the simple hop-count metric

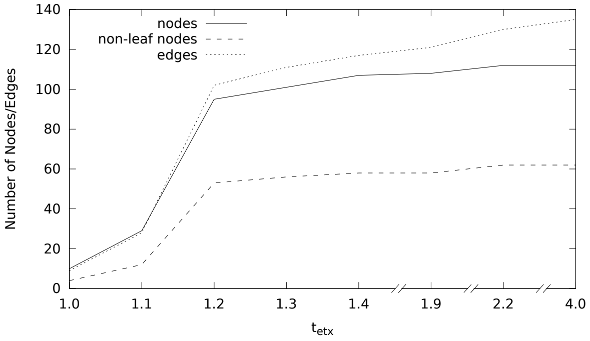

Figure 4. The properties of the largest connected component increasing t_{etx}

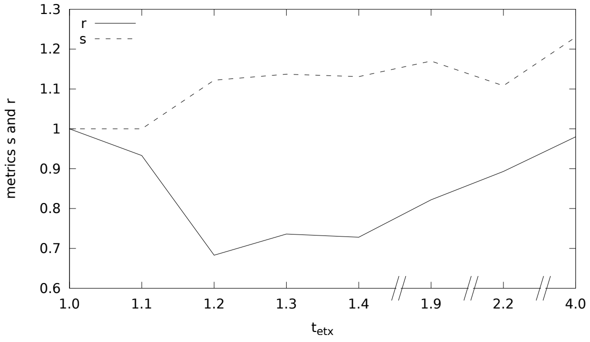

Figure 5. Average path weight

is used there is a high chance of using a route $P_u(i,j)$ that corresponds to $P_w(i,j)$.

As said, this does not mean that the quality of $P_u(i,j)$ must be similar to the quality of $P_w(i,j)$. For this reason Fig. 6 reports the average value $C(P_w(i,j))$ showing that it does not significantly differs from c.

Lastly, Fig. 7 reports the estimated number of MPRs that OLSR would need to keep the network topology connected. To compute this value the heuristic used in OLSR to choose the MPR set $M(j)$ of node j has been re-implemented in the NetworkX framework and evaluated on each node in G_w. Fig. 7 reports the number of MPRs in the network, that is $|| \cup_{j \in N} M(j)||$. Note that the heuristic is the original one from RFC 3626 that tries to minimize the size of $M(j)$ and doesn't take into consideration the quality of the links. The graph shows that the number of MPRs could be safely reduced to 44% of the nodes. The TC messages generated would be reduced to 44% of the current value and the number of forwarded TCs would be reduced by approximately the same factor, giving great benefit to the network.

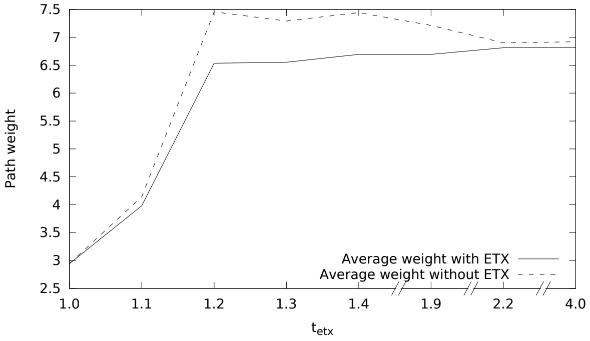

Figure 6. Shortest path weight with and without ETX metric

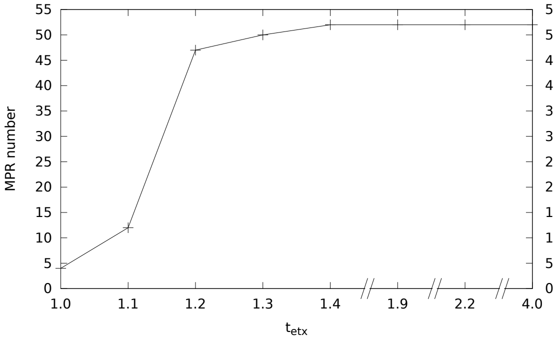

Figure 7. Estimated MPR number

Note also that if only the non-leaf nodes are chosen to be MPRs, there is still a significant improvement possible on the number of MPRs.

C. Discussion

From the presented results it would seem reasonable to drop ETX and re-introduce MPRs. Nevertheless, dropping ETX metrics would have at least two foreseeable bad consequences. First, even if the average difference shown in Fig. 6 is quite close, the maximum difference may be larger. When t_{etx} = 4, in the worst case $P_u(i,j)$ has a cost that is 3.6 units heavier than $P_w(i,j)$. Second, if ETX is not used, bad links will be used up to the moment when they break down. If $P_u(i,j)$ includes a link l_{mn} that connects node m and node n and the traffic on l_{mn} increases over its capacity, its ETX will increase. This will continue up to when l_{mn} becomes unusable and nodes m and n will agree that the link is broken. At that time $P_u(i,j)$ will avoid using l_{mn}. But when the link becomes unloaded, its ETX will decrease again, so that the link will return usable and the $P_u(i,j)$ will possibly switch back to using l_{mn}. ETX smooths this process and protects from constant route fluctuations [6].

Given this, a solution that seems to be feasible is to use ETX metrics with the original heuristic for the choice of MPRs. In this case when node i selects the path to node j it will use an approximated knowledge of the network, since it will only know the ETX for a subset of the links (the links connecting all the MPRs to their selectors), and this will potentially lower the quality of the choice. Nevertheless the routing decision on the path to the destination is performed at each hop by a node that is able to use all its available knowledge, which is a perfect knowledge in its two-hop neighborhood.

With this approach the stability of the MPR choice would be preserved and signalling would be reduced while keeping the benefits of ETX. Note also that `TC` messages can be extended in order to carry information on all the links of an MPR, so that a perfect knowledge of all the links can be given to any node at the cost of having larger `TC`s.

Summing up, from the analysis performed it can be seen that even if the ETX metric is not used at all, the difference in the quality of the paths is not so evident, and that dropping ETX would make it possible to use MPRs to sensibly save resources. Indeed, the ETX metric has some other advantages over hop-count that must be preserved, so that a mixture of MPRs chosen with the original heuristic and ETX metric seems a reasonable trade-off.

VII. Group centrality metrics

In graph theory centrality metrics have been largely used to identify the properties of nodes. In particular, in social science the centrality of a node is often used to determine the influence that a person has on the other participants of the social network. In the context of wireless networks, centrality has not received much attention up to recent times [7] [8].

The concept of centrality in a specific graph is not unique, in this paper two definitions of centrality are considered, shortest path betweenness centrality (C_{sp}, or simply *betweenness*) and closeness centrality (C_c) [9].

Given the graph G_w the shortest path betweenness $C_{sp}(k)$ of node k is defined as the fraction of shortest paths between any couple of nodes (i, j) passing through k. Assuming that the traffic matrix is homogeneous (or it is unknown), $C_{sp}(k)$ is a good and unbiased estimator of the fraction of traffic that a node will route over the total traffic generated in the network. If one wants to place a traffic analyzer in the network (for instance an Intrusion Detection System, IDS), the node with the highest C_{sp} is the best choice to analyze the highest fraction of the overall traffic.

The closeness centrality $C_c(k)$ of node k, instead, is an estimation of how many hops are needed to spread an information from k to all the nodes in the network. The definition that best serves the purposes of this paper is that $C_c(k)$ is the average distance from k to any other node i in the network. If one wants to place a service in the network (like a web server, a streaming server, a VoIP server etc.) the node with the lowest C_c is the best choice to minimize its distance to any node in the network. For both centrality measures the G_w graph is used, so that distances are weighted using ETX; thus, $C_c(k)$ is the number of wireless frames that will be needed to successfully send one IP packet averaged over the path from any node i to the service placed on k.

The definition of both metrics can be extended to groups of nodes. The group betweenness of a group γ of nodes is defined as the fraction of shortest paths between any couple of nodes (i, j) passing through at least one node $k \in \gamma$. Again, as an example, if one wants to place an IDS on a group γ of nodes in order to maximize the overall fraction of traffic analyzed, the group with the highest $C_{sp}(\gamma)$ is the best choice.

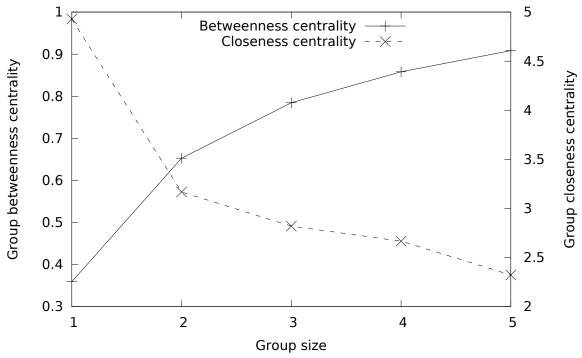

Figure 8. Group centrality metrics

The closeness group centrality of γ is the average of the minimum shortest paths from any node i to any of the nodes in γ. More formally, given γ, for each node i we can define a set $\Gamma(i)$ as follows:

$$\Gamma(i) = \{C(P_w(i, j)) \; \forall j \in \gamma\} \tag{2}$$

we define $C_c(\gamma)$ as the quantity:

$$C_c(\gamma) = \frac{\sum_{i \in \{N \setminus \gamma\}} min(\Gamma(i))}{||N \setminus \gamma||} \tag{3}$$

Among all the groups of the same size, the more central one is the one with the lowest $C_c(\gamma)$.

Following the previous example, if a service can be replicated on a set γ of nodes, then the group with lowest $C_c(\gamma)$ is the best choice. This group definition reflects a situation in which every node i is aware of the presence of the service on the nodes in γ and it is able to freely choose the best one based on the quality of the routes. As a further example imagine that, given a network graph G_w one can choose a set of nodes γ where to place Internet gateways. The group with the lowest $C_c(\gamma)$ is the group that will give the best average Internet connectivity to the nodes in the network.

Finding γ with the highest group betweenness has been shown to be an NP problem [10] while a brief analysis of the literature did not produce any complexity estimation for the closeness group centrality as defined in Eq. (3). For this paper the results have been obtained with a greedy algorithm that exhaustively explores all the combinations of groups of nodes of a certain size (the source code is available at the websites osps.disi.unitn.it and www.pervacy.eu).

Fig. 8 reports both group betweenness centrality and group closeness centrality with a group size ranging from 1 to 5. There is a performance gap between using a single node or more than one node while for larger group sizes the curves have a smaller slope.

A. Discussion

One of the claims that is often associated to mesh and ad-hoc networks is that they are difficult to *wiretap* due to their distributed nature. For the Ninux network this is not necessarily true, since Fig. 8 shows that if an attacker is able to control a very small number of nodes in the network he is also able to sniff 90% of the overall traffic. The other side of the coin is the mentioned IDS scenario. An IDS is a host that extract traffic traces at any level in the networking stack and looks for known patterns corresponding to worms, viruses etc. When the traffic load is high the IDS will need proper hardware to accomplish its function, which is hardly compatible with the poor hardware of mesh nodes. If the community wants to enforce an IDS it will have to add some specialized hardware to some existing node; group betweenness can be used to choose the nodes that maximize the analyzed traffic.

Betweenness centrality is thus important to characterize security and privacy features of a wireless community network both for an attacker and for a network manager.

Closeness centrality is a concept that can be successfully coupled with OLSR. As said, using OLSR the nodes can expose the addresses of foreign networks they are attached to. This principle has been extended to identifiers other than IP addresses such as network shares or DNS names [11]. Using this approach a service can be replicated on the nodes in γ, and every node i in the network will be aware of the IP address at which the service is available. Node i will also be able to access the service from the node in γ that has the lowest cost to be reached, just as it currently happens to find Internet gateways.

Again, in such a situation the choice of the most central group γ is fundamental to minimize the average cost to reach the service. As Fig. 8 shows, with $||\gamma|| = 5$ the average cost is lower than 2.5 frames per IP packet delivered.

The Networking Lab of the DISI department has a robust background on P2P networking, with particular attention to P2P video streaming. Currently the PeerStreamer [12] software platform is being tested on community networks, and in particular in the community-lab offered by the CONFINE project [4]. As a future development we plan to study the integration of centrality metrics with P2P streaming in order to permanently run PeerStreamer on a small set of central nodes that will minimize the cost of the access to the stream from any other node in the network. This would minimize the time to access the video for the first time and could be the basis for a distributed optimization of the placement of PeerStreamer nodes in the network. In a plausible and challenging scenario, PeerStreamer could just be embedded on every node and activated dynamically on nodes that have a high centrality. Initial steps in this direction have been already produced with the study of the centrality of OLSR MPRs [13].

VIII. Conclusions

The aim of this paper was to explore the Ninux topology and to describe its main features in order to have a better knowledge of its dynamics and possibly take wiser choices to improve its performances. Relevant results have been produced regarding the impact of the ETX metric and group centrality metrics. Since the Ninux features seem to be compatible with other community networks there are high chances that the obtained results could be exported to other networks as well. Next steps in this line of research involve the study of the same topics on other community networks in order to verify the similarities and the integration of centrality metrics with the suggested applications, IDS and video streaming.

Acknowledgements

This work is funded by the Trentino program of research, training and mobility of post-doctoral researchers, incoming Post-docs 2010, CALL 1, PCOFUND-GA-2008-226070 and by the European Commission under Grant Agreement No. FP7-288535 “CONFINE”: Open Call 1, *Open Source P2P Streaming for Community Networks –OSPS–*.

The author wishes to thank the Ninux Community and the CONFINE researchers for the support, with particular reference to Henning Rogge for the insights on the OLSRd implementation.

References

[1] S. Jain and D. Agrawal, “Wireless community networks,” *IEEE Computer*, vol. 36, no. 8, 2003.

[2] D. Vega, L. Cerda-Alabern, L. Navarro, and R. Meseguer, “Topology patterns of a community network: Guifi.net,” in *IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain*, 2012.

[3] P. Frangoudis, G. Polyzos, and V. Kemerlis, “Wireless community networks: an alternative approach for nomadic broadband network access,” *IEEE Communications Magazine*, vol. 49, no. 5, 2011.

[4] A. Neumann, I. Vilata, X. León, P. E. Garcia, L. Navarro, and E. López, “Community-lab: Architecture of a community networking testbed for the future internet,” in *IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain*, 2012.

[5] IETF Network Working Group, “Request for Comments: 3626, “Optimized Link State Routing Protocol (OLSR)”,” 2003. [Online]. Available: http://tools.ietf.org/html/rfc3626

[6] D. Johnson and G. Hancke, “Comparison of two routing metrics in OLSR on a grid based mesh network,” *Ad Hoc Networks*, vol. 7, no. 2, 2009.

[7] D. Katsaros, N. Dimokas, and L. Tassiulas, “Social network analysis concepts in the design of wireless ad hoc network protocols,” *IEEE Network*, vol. 24, no. 6, Dec. 2010.

[8] M. Kas, S. Appala, C. Wang, K. Carley, L. Carley, and O. Tonguz, “What if wireless routers were social? approaching wireless mesh networks from a social networks perspective,” *IEEE Wireless Communications*, vol. 19, no. 6, pp. 36–43, 2012.

[9] M. Newman, *Networks: an introduction*. OUP Oxford, 2009.

[10] P. Ou and Z. Li, “A variant betweenness centrality approach towards distributed network monitoring,” in *Proceedings of the 2011 Fourth International Symposium on Parallel Architectures, Algorithms and Programming*, Washington, DC, USA, 2011.

[11] F. S. Proto and C. Pisa, “The olsr mdns extension for service discovery,” in *6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks. Rome, Italy*, 2009.

[12] R. Birke, E. Leonardi, M. Mellia, A. Bakay, T. Szemethy, C. Kiraly, R. Lo Cigno, F. Mathieu, L. Muscariello, S. Niccolini, J. Seedorf, and G. Tropea, “Architecture of a network-aware p2p-tv application: the napa-wine approach,” *Communications Magazine, IEEE*, vol. 49, no. 6, 2011.

[13] L. Maccari and R. Lo Cigno, “Betweenness estimation in olsr-based multi-hop networks for distributed filtering,” *Elsevier Journal of Computer and System Sciences*, accepted for publication in 3Q 2013.