Self-* Properties and P2P Technology on Disruption-Tolerant Management

Jéferson Campos Nobre, Pedro Arthur Pinheiro Rosa Duarte,
Lisandro Zambenedetti Granville, Liane Margarida Rockenbach Tarouco
Institute of Informatics – Federal University of Rio Grande do Sul
Porto Alegre, Brazil
Email: {jcnobre, paprduarte, granville, liane}@inf.ufrgs.br

***Abstract*—The introduction of self-* properties has been proven to be a feasible approach for the management demands of Disruption-Tolerant Networks (DTNs). Among the properties of the self-* management vision, self-healing figures as a key property to improve the dependability of the managed infrastructures. An interesting possibility to materialize self-* support in disruption-tolerant management is through the employment of Peer-to-Peer (P2P) technology. In this paper, we introduce a P2P self-healing service tailored for disruption-tolerant management. We implemented the proposed service using ManP2P-ng, an open source P2P-based network management system. In addition, an experimental evaluation is performed considering as case study the disruption-tolerant management of a Host-based Intrusion Detection System (HIDS) deployed on a specific kind of DTN: Internet access networks for remote villages. The results show the feasibility of our proposal and its interesting features.**

I. Introduction

Traditional network management models rely on some premises of the underlying managed network that not always hold, specially when considering non-conventional network technologies. Traditional networks, for example, present low delay and permanent connectivity. Examples of network technologies that challenge these premises include *Internet access networks for remote villages* [1] and *vehicular networks* [2].

Occasionally-connected and high delay networks can be embraced by the Disruption-Tolerant Network (DTN) concept. Although investigations on DTN technologies largely exist and despite the fact that several DTN scenarios explicitly require management features [3], just a few investigations have been carried out on disruption-tolerant management. Some investigations address restricted scenarios [4], but more general solutions have not been accomplished yet. Several disruption-tolerant management issues thus remain; the definition of a disruption-tolerant management system is still a challenging exercise [5].

One natural option to realize disruption-tolerant management is through the employment of Distributed Network Management (DNM) approaches, which in turn have led to improvements over the traditional centralized approaches. Indeed, some authors have already proposed the use of traditional DNM approaches to manage DTNs [4]. However, the complexity of DTNs demands management features that are not present, at least not directly, in usual DNM technologies (*e.g.*, support for intermittent connectivity). As one of the particular DNM technologies, Peer-to-Peer (P2P) management overlays present themselves as a mix of the characteristics from P2P and network management systems, improving, for example, connectivity robustness among management entities and load balance of management tasks [6].

The use of P2P technology in network management, also known as P2P-Based Network Management (P2PBNM), has several interesting characteristics for disruption-tolerant management, such as the use of local data and autonomy of management entities. In addition, P2PBNM eases the introduction of advanced features into network management, such as self-* properties [7]. Some authors [4] also argue that these properties are essential in disruption-tolerant management because they enable proactive actions to be taken based solely on local information and resources when DTN nodes get temporarily isolated from the rest of the network. Among the self-* properties, self-healing is quite appealing in disruption-tolerant management since healing procedures can be performed without having to communicate with a centralized entity (such as a management station) and this communication is usually intermittent and/or significantly slow.

In this paper, we propose a self-healing service for DTNs using P2P technology. We present an extension of a prototype P2PBNM system to support overlay operations in face of delays and disruptions. Our self-healing service exploits the support for disruption-tolerant transport and grouping of management messages. The main contribution of our work is to provide self-healing features to manage elements of DTNs regardless of the existence of centralized managers. We also show that it is possible to reuse management applications present on P2PBNM systems in DTN environments. For example, a P2P monitoring application for conventional networks can be adapted to cope with the harsh conditions of DTNs. We evaluate our proposal over a case study composed of the self-healing support for distributed Host-based Intrusion Detection Systems (HIDSs) in a specific DTN environment: Internet access networks for remote villages.

The remaining of this paper is organized as follows. In Section II, we present the main concepts related to DTNs and self-* properties in P2PBNM approaches. The disruption-tolerant management of Internet access networks for remote villages is depicted in Section III. In Section IV, our proposed solution and its associated concepts are described. The details of a proof-of-concept implementation are presented in Section V. In Section VI, we introduce a case study and its related experimental evaluation. Finally, conclusions and future work are provided in Section VII.

978-1-4799-3755-4/13/$31.00 ©2013 IEEE

II. Background

In this section, we first discuss the fundamentals regarding Disruption-Tolerant Networks (DTNs) and a related DTN protocol. Afterwards, we review the state-of-the-art on self-* properties in P2P-Based Network Management (P2PBNM). These properties improve the desirable characteristics of P2PBNM features in disruption-tolerant management.

A. Disruption-tolerant networks and HTTP-DTN

Nowadays, new classes of non-conventional computer networks have arisen in consequence of technological research and development. In these networks, one can observe that some assumptions of the TCP/IP architecture are relaxed, such as the existence of an end-to-end path between hosts, a non-excessive maximum round trip time, and a small probability of having packets discarded. Networks in which these assumptions cannot be met are embraced in the DTN concept [8].

Several "clean slate" protocols for DTN delivery have been already proposed (*e.g.*, Bundle Protocol [8]). However, some authors suggest the reuse of existing standards in order to simplify the deployment of DTNs [3]. One of these suggestions consists in exploiting the HyperText Transfer Protocol (HTTP) in DTN environments as a transport layer independent, session protocol among communicating DTN nodes (hop-by-hop). This HTTP usage is referred to as HTTP-DTN [3].

Messages can be grouped in HTTP-DTN through the use of *Package-** fields; these groups are thus called *packages* (in an analogous way to the use of *bundles* in the Bundle Protocol [8]). Grouping messages is useful in DTNs because it can save scarce resources such as bandwidth.

B. Self-* properties in P2P-Based Network Management

The use of P2P technology in network management, also known as P2P-Based Network Management (P2PBNM), is an approach to integrate characteristics of Distributed Network Management (DNM) and P2P overlays [6]. In P2PBNM, management tasks are performed by peers of the P2P management overlay through management components. These components advertise the tasks they perform as management services. Moreover, peers are organized into groups according to their management services.

P2PBNM eases the introduction of self-* properties in network management. These properties can be effectively deployed using concepts from P2PBNM, such as management components and services. In this context, some initiatives investigated the joint use of self-* properties in P2PBNM. In a previous work of our research group, Duarte *et al.* [7] proposed a self-healing mechanism through the composition of monitoring and healing components. DTNs were not supported, however, because the self-healing mechanism was operating assuming that network connections were stable, which is not the case in DTNs.

In summary, the joint use of P2P overlays and self-* properties in network management offers improvements in different aspects, such as scalability and reliability. Despite these improvements, P2PBNM cannot be directly used in DTN management without proper adaptations; there are still issues to be addressed when connectivity premises do not hold.

III. Disruption-tolerant management of Internet access networks for remote villages

The support of Internet access for remote villages (*e.g.*, DieselNet [1]) is an example of DTN environment in which disruption-tolerant management is required. These access networks typically follow the *island network model*, where a group of neighbor nodes sharing some property (*e.g.*, low-latency) forms a *connectivity island*. These islands are connected to one another through a data mule. Despite homogeneous network characteristics inside each island (*e.g.*, a specific remote village), these characteristics are not equivalent across the entire network.

The recurrent occurrence of disruptions in Internet access networks for remote villages causes the traditional end-to-end control loops to break. Because of that, these control loops are transferred to separate control loops among islands/nodes. Inside an island, it is feasible for one to access a device by being physically at it or via real-time access (known as "local management" in the DTN research area). However, the assumptions that make local management available in this kind of infrastructure remain only for restricted time periods and/or limited network portions. In contrast, "remote management" implies managing a node over a DTN, assuming that delays and disruptions may be experienced. We use the concept "disruption-tolerant management" to embrace both local and remote management in DTNs.

The execution of disruption-tolerant management tasks in island-like networks can be improved with the use of local information (data and logic). An approach to perform disruption-tolerant management tasks using local information is to employ the so called self-* properties [4]. These properties can overcome the propagation time necessary to exchange and process management commands and/or data. For instance, self-healing features can decrease the time to heal, since these features do not require a management station reaction to start the healing procedures.

The employment of P2P technology in network management may provide a substrate to address the requirements of disruption-tolerant management solutions. A P2P management overlay can merge local data and logic to perform remote network management tasks as well as ease the introduction of self-* properties. The details about this employment and how it is related with disruption-tolerant management are further discussed in the next section.

IV. A P2PBNM extension for disruption-tolerant self-healing

We propose to support disruption-tolerant management by introducing an extension on P2P-Based Network Management (P2PBNM) systems. Besides that, we explain the concepts behind a disruption-tolerant self-healing service tailored for this extension. The P2PBNM extension and the disruption-tolerant self-healing service are described in the present section.

A. A P2PBNM extension for disruption-tolerant management

The P2PBNM extension promotes two processes in each peer that forms the management overlay: the maintenance of the management overlay, now operating over a disruption-tolerant environment; and the intermediation on the execution

of disruption-tolerant management tasks. Such intermediation can enable management messages being transferred from management applications to managed devices even in scenarios with disruptive connections; it can also enable the realization of more sophisticated management tasks, such as disruption-tolerant self-healing.

The P2PBNM extension deals with network disruptions by using store-and-forward techniques and adapting disruption-tolerant protocols to overlay operation. The extension is not tied to any disruption-tolerant or IP-related protocols, thus it is possible to wrap different disruption-tolerant protocols through its facilities. A disruption-tolerant protocol that can be employed by the P2PBNM extension is the HTTP-DTN (described in Section II).

The P2PBNM extension for disruption-tolerant management also enables message grouping. The policies that define how to group management messages are enforced by the extension and specified by human administrators.

B. A disruption-tolerant self-healing service

The P2PBNM extension proposed in the present Section provides a message delivery service transparent of disruptions for management peers. However, beyond the features offered by the proposed extension, a self-healing service still has to be tailored to deal with DTN environments. In a previous work [7] we have introduced a P2P self-healing service for conventional networks based on *monitoring and healing workplans*. Such workplans compile, using a well-defined language, the administrator's knowledge on how to monitor and heal the managed infrastructure. Although workplans and the self-healing service itself do not require a specific network technology, our original self-healing mechanism cannot be directly deployed in DTNs because it assumes stable network connections.

From now on, we present the adaptations over the self-healing service, which enable the monitoring and healing of managed entities in DTNs. When required, the human administrator calls the self-healing service informing monitoring and healing workplans, as well as the target managed entities to be observed and eventually healed. The adapted version of our self-healing service for DTNs has the following full interface:

```
SelfHealing(
  managed entities, monitoring workplan,
  healing workplan, scheduling, unhealthy threshold,
  [candidate monitoring peers], [candidate healing peers])
```

The first parameter is a list of `managed entities` which are the targets of the self-healing service. Each entry in this list contains relevant information about the targets entities, such as network address, transport protocol and port, and operating system. The `monitoring workplan` and `healing workplan` parameters inform how the target entities must be monitored and healed, respectively. The `scheduling` parameter specifies how often the monitoring peers must interact with target entities, as well as the time span of the self-healing service for a specific request. The `unhealthy threshold` parameter sets how many unique unhealthy notifications (to be further described) must be received before starting a healing procedure. The `candidate monitoring peers` and `candidate healing peers`, both optional parameters, list candidate peers for monitoring and healing purposes, respectively, provided as an aid to deal with locality matters (*e.g.*, the nearest peers to a given target entity). When one of these parameters is not present, the overlay randomly chooses a subset of peers of the respective groups to host the workplans.

The execution of a monitoring workplan starts according to the scheduling definitions informed in `scheduling` parameter. Monitoring workplan execution occurs concurrently, without further synchronization or coordination among the peers which host it since the instability of end-to-end paths would either prevent or largely retard its scheduling. This design guarantees that the monitoring workplan will reach its targets with a probability equal to the sum of the probabilities of each monitoring peer to have an encounter with the target. The result of the execution of a monitoring workplan is the diagnosis of the operational status of managed entities. If this diagnosis indicates that a target managed entity is *healthy*, the workplan is rescheduled for another round of execution according to the rules specified in the `scheduling` parameter. However, if the diagnosis indicates that the managed entity is *unhealthy*, the monitoring peers that noticed the fault send an *unhealthy notification* to healer peers. In fact, *unhealthy notification* messages are sent to all peers that form a group able to heal the failing target entity. If the `candidate healing peers` parameter has been provided during service request, notifications are sent only to peers in that list.

Once the healing workplan is finished and reaches a positive result (*i.e.*, the target entity is operational again), healing peers notify back the monitoring peers. This notification enables the monitoring peers to resume their workplans so the target entities start being observed again. If the healing workplan, however, cannot heal the target entities, the healing peers request the monitoring ones to completely stop the previously paused monitoring workplan and to notify the human administrator about the unsuccessful attempt to heal the target entities.

V. Implementation

We implemented the P2PBNM extension and the delay-tolerant self-healing service previously described in Section IV as a proof of concept. The implementation was built on top of ManP2P-ng[1], which is an open source P2PBNM system that follows the service-oriented approach. In this section, first we describe a simplified version of the ManP2P-ng architecture considering the present implementation, then we discuss the implemented software components in more detail.

A. ManP2P-ng architecture

A simplified version of the ManP2P-ng architecture is depicted in Figure 1. In this Figure, the software components implemented in the context of the present work are highlighted along with the ManP2P-ng layers they belong. ManP2P-ng has embedded mechanisms for the maintenance of the management overlay and provides an *Application Programming Interface* (API) in order to extend the overlay features.

[1]ManP2P-ng - http://projects-redes.inf.ufrgs.br/gf/project/manp2png/

In the ManP2P-ng architecture, *Components and Plug-ins* (Figure 1a) embrace run-time management software components defined through component descriptors. Management components have a unique identifier used for grouping peers according to the services they expose. In addition, this identifier is used by system administrators to request the execution of management services.

Transport layers and addressing abstractions are provided by the *Networking Interfaces* (Figure 1b). These interfaces enable the overlay to operate transparently over many different interconnection technologies. The abstractions provided by these interfaces are the building blocks for messaging and overlay maintenance facilities. Transport protocol support can be expanded at run-time through transport adapters.

Delay-tolerant management components usually deal with delays and disruptions by using store-and-forward techniques, whose support is provided through *Storage Interfaces* (Figure 1c). These interfaces provide back-ends for transactional structured persistence (using database management systems) and unstructured persistence (using the peer own file system).

B. Implemented software components

The P2PBNM extension is implemented as a disruption-tolerant transport adapter in networking interfaces of ManP2P-ng (Figure 1b). We initially chose to implement the adapter using HTTP-DTN [3] because we were primary interested in opportunistic connections (*i.e.*, networks with frequently disruptions) without long delays. Since the delay-tolerant transport adapter requires data persistence, we use *SQLite*[2], a lightweight relational database management system, as a back-end in storage interfaces (Figure 1c) to meet this requirement.

The disruption-tolerant transport adapter uses the ManP2P-ng address structure to bind against HTTP-DTN addressing. Thus, peer addressing information is provided through HTTP-DTN specific header fields. These fields operate using HTTP/1.1-compliant specifications. For addressing purposes, *Content-Destination* and *Content-Source* fields identify the destination and the source of the data, which are filled with peers identifiers. Thus, it is possible to rely on ManP2P-ng routing and address capabilities for message delivering.

The disruption-tolerant transport adapter supports grouping management messages in order to save network resources. Considering HTTP-DTN, this is performed using the *Package-** fields and these message groups are then called *packages*. Initially, the policy to control which management messages may be grouped in HTTP-DTN packages is defined according to the source and destination fields. However, this policy may be adjusted at run-time in order to use other overlay parameters, such as grouping messages produced by or addressed to the same management component.

[2]SQLite - http://www.sqlite.org/

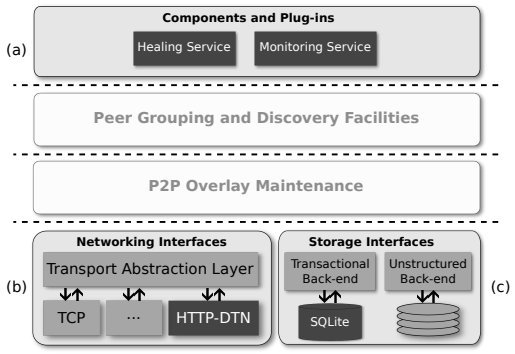

Fig. 1. ManP2P-ng Architecture

The self-healing mechanism is implemented in ManP2P-ng through two management components: the monitoring service and the healing service components (Figure 1a). Despite the fact that overlay characteristics are abstracted by ManP2P-ng P2P services and the P2PBNM extension, both components require some adaptations to properly perform in the presence of delays and disruptions. The adaptations implemented for the self-healing mechanism mainly reflect the parameters added in the mechanism interface, such as `scheduling` and `unhealthy threshold`. Furthermore, two optional parameters were also introduced (`candidate monitoring peers` and `candidate healing peers`) in order to deal with locality matters, such as choosing the nearest monitoring and healing peer groups. As such, it is now possible to encompass many management entities, as well as their specific data, in one service request by providing a list of managed entities through the `manage entities` parameter.

VI. Evaluation

In this section, we present an experimental evaluation considering a case study in order to observe the feasibility of our proposal. First, we describe this case study and its specific details. Then, we depict experiments performed considering the case study scenario.

A. Case study

Our case study consists of using our proposed disruption-tolerant self-healing service to assist a distributed Host-based Intrusion Detection System (HIDS) in an infrastructure that supports Internet access for remote villages (as described in Section III). A HIDS monitors and analyzes hosts in order to determine whether they are being attacked or compromised. However, a HIDS itself can be compromised by either new/unknown attacks or non-malicious faults. Furthermore, faults can have a bigger impact in the dependability of HIDS operating in this kind of infrastructure, since downtimes can be increased because human administrators may take too long to access the remote managed entities. In this context, self-* features (*e.g.*, self-healing) in management tasks become a necessary feature.

The fault monitoring and healing operation is traditionally performed through the periodical poll issue by a network management system. On the occurrence of faults, human administrators must diagnose and fix the system manually. Besides, traditional pull mechanisms that are usually employed in administrative remote commands increase the propagation time needed to handle data exchange.

The P2PBNM extension (described in Section V) can improve the dependability of HIDSs deployed in DTN environments. The transport adapter supports P2P overlay communication services to operate in a disruption-tolerant approach. The

monitoring component of ManP2P-ng is responsible for periodically observing the state of managed devices/systems and identify anomalies/faults. The healing component of ManP2P-ng waits for an anomaly/fault notifications issued by the monitoring service. Besides, the disruption-tolerant transport can be also used to push management data (*e.g.*, workplans) into remote villages.

B. Experiments

Our experiments aim to measure the total management traffic generated throughout the self-healing process. Our experimental setup is composed of three DTN islands (*#1*, *#2*, and *#3*), each one hosting its own HIDS infrastructure (encompassing one HIDS manager and eight sensors), in addition of eight management peers able to both monitor and heal target managed entities. Following the presented case study, a data mule travels through each island always in the same order, from *island #1* to *#2* and then *#3*. We call a *visit* a whole round of the mule arriving and leaving all islands to deliver and collect DTN traffic.

Inside an island, three peers are chosen to monitor the local HIDS infrastructure. Three other peers are picked up to monitor HIDS from the first remote island, and finally, three peers monitor the second remote island HIDS. The same strategy of using nine peers is employed for the healing purposes. The motivation for performing intra-island monitoring is to provide a location agnostic diagnosis of the problem.

We performed our experiments using a set of two virtual machine (VM) servers and one commodity computer. The first VM server hosted nine VMs (three VMs per DTN island), each one executing one instance of the ManP2P-ng with the proposed P2PBNM extension. The second VM server hosted another nine VMs: one for the HIDS manager and other eight for the HIDS sensors, all of them positioned in the third DTN island. We used OSSEC[3], an open source HIDS, as our managed entities. Finally, we used our third computer for simulating the data mule described in our experiment setup.

In the first experiment, we measure the total generated traffic from the data mule to deliver nine different monitoring and healing workplans for nine peers. This is the worst case scenario for the data mule in terms of management traffic. Such workplans describe how each of the nine peers must monitor and heal the HIDS infrastructure hosted in *island #3*.

Figure 2 shows the traffic inside each island during the first nine visits in the managed DTN concerning a typical experimental run. In *visit #1*, the management overhead is higher because of the workplans distribution. In *visit #2*, workplan acceptance notifications are still carried around the managed DTN, so the traffic is still considerable. From *visit #3* and on, it is possible to observe that the management traffic reduces significantly (less than 70 kb per visit), thus consisting of regular overlay maintenance and monitoring messages.

In our second kind of experiments, we force the HIDS manager from *island #3* to go down, emulating, for example, an attack or a non-malicious managed device fault. Three peers from each island detect the faulty HIDS manager and issue notification messages that report this situation. Given

[3]OSSEC - http://www.ossec.net/

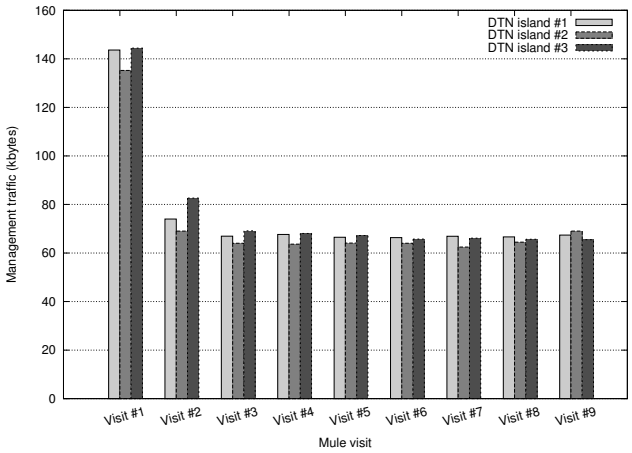

Fig. 2. Monitoring and Healing plans distribution

that the *unhealthy threshold* (defined in our experiments as 6) is crossed, since nine *unhealthy threshold* notifications are sent, the healing workplans are triggered also in nine peers, three from each island. In order to heal the faulty HIDS, not only the broken manager need to be fixed, but also the eight sensors from *island #3* must be reconfigured. In Figure 3, we present the impact of the self-healing procedure in the managed DTN concerning a typical experimental run. The failure in the HIDS manager is emulated during the mule's *visit #52* and its effects are more perceptible in *visit #53*. This increased traffic is mainly due to unhealthy notifications issuing by the monitoring peers of all islands.

During *visit #54*, the management traffic increases even more because the mule carries remaining unhealthy notifications from the previous round in addition of carrying action requests resulting from the execution of the healing workplans. In *visit #54* the faulty HIDS is recovered and the first *healthy notifications* are issued by the healing peers from *island #3*. That contributes for the increased traffic of *visit #54* too. In *visit #55*, the last *healthy notifications* are issued. The monitoring peers from all three islands then resume their monitoring workplans. From *visit #56* and on, the self-healing procedure is complete and the management traffic returns to its normal levels.

In our last kind of experiments, in order to evaluate the relationship between the size of the HIDS infrastructure and

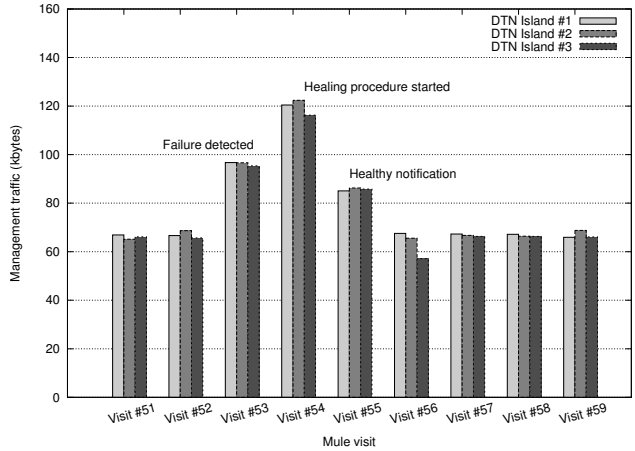

Fig. 3. Management traffic during the self-healing process.

978-1-4799-3755-4/13/$31.00 ©2013 IEEE

the generated management traffic, we vary the number of sensor nodes in the HIDS infrastructure from one to eight, although only one manager node is still in place. Different than the previous experiments, we focus on the traffic generated by a single healing peer when it executes a healing workplan. Two message passing strategies, during the healing procedure, are considered. The first one consists in sending individual messages through single HTTP-DTN packages. In the second one, HTTP-DTN message packaging is employed; in this case, management messages are grouped into a HTTP-DTN package that is constantly reassembled until the mule visits the DTN island that hosts the healing peer. When the mule finally arrives at the island, the message grouping stops and the HTTP-DTN package is forwarded.

Figure 4 presents the traffic generated by the healing workplan, executed by a single healing peer, considering a varying number of sensor nodes, as mentioned before. Three curves are depicted: one for P2P healing traffic without employing HTTP-DTN (inclued just as a baseline since it does not support disruption-tolerant management), a second one for HTTP-DTN packages carrying a single management message, and a last one using HTTP-DTN packages carrying several management messages, as discussed in the previous paragraph. The results presented in Figure 4 show that transporting management messages through HTTP-DTN in P2P overlay is feasible since the overhead is not significantly higher than when the same management tasks are performed in a non-DTN environment. Moreover, HTTP-DTN packaging features smooth the overhead growth, which can be seen in the similar angular coefficient of both curves (P2P only self-healing and HTTP-DTN Packed P2P self-healing).

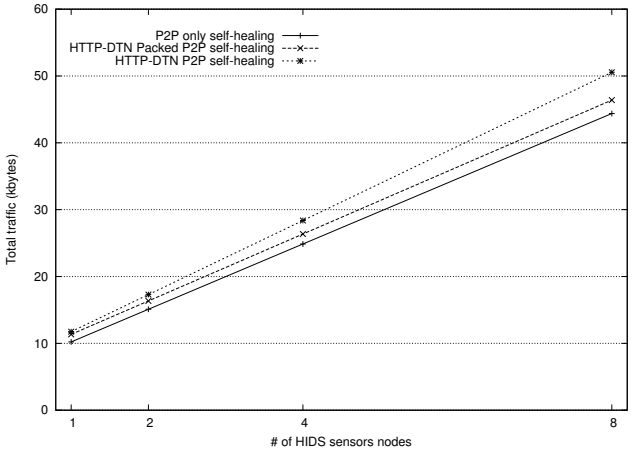

Fig. 4. Total management traffic during the Self-Healing process.

The results present in this Section show important features of our proposal. First, the traffic produced during workplan distribution and regular monitoring does not impact importantly, considering our case study. Second, despite an increase in management traffic, the healing procedure still keeps an acceptable traffic load. Third, it is also shown that the introduction of HTTP-DTN in P2P overlay does not increases significantly network overhead, specially when using HTTP-DTN packages.

VII. Conclusions and Future Work

Disruption-Tolerant Networks (DTNs) technology imposes novel challenges for network management. Usual techniques employed by management solutions (*e.g.*, control loops) cannot be directly applied on DTN scenarios since frequent connection disruptions explicitly defy their basing premises. Furthermore, the self-* properties are implicit requirements to manage these networks since the same challenges which prevents the straightaway employment of current solutions also prevents administrators to promptly manage these networks.

In this paper we have initially argued that the intrinsic characteristics of P2PBNM makes it a suitable candidate for a common substrate to manage DTN environments. We then introduced an extension for P2PBNM systems which uses IETF's HTTP-DTN in order to provide store-and-forward capabilities as well as message packaging. Besides the aforementioned, this extension permits the reuse of management components already present in P2PBNM systems. In this context, we detailed the adaptation of a self-healing service using the prototype P2PBNM system ManP2P-ng. The results of an experimental evaluation show that the traffic produced during the different stages of the self-healing service does not have a substantial impact on the traffic load, specially when using message packaging.

The research for DTN management techniques is in its early days and there is plenty of room for improving the work described in this paper. Despite the good results achieved in the scenario chosen for evaluating our solution, it is necessary to evaluate more complex scenarios, increasing the number of healed elements and their heterogeneity as well as the number of peers in the management overlay and churn rate. Furthermore, the security features of our solution also need improvement, since in many DTN scenarios these features are mandatory, such as in military DTN environments. Thus, we intend to investigate the use of HTTPS and S/MIME for improving the hop-by-hop and end-to-end security in the management communication sessions.

References

[1] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*. IEEE, 2006, pp. 1–11.

[2] J. Zhao and G. Cao, "VADD: Vehicle-assisted data delivery in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1910–1922, 2008.

[3] L. Wood, W. Eddy, and P. Holliday, "A bundle of problems," in *Proceedings of the IEEE Aerospace conference 2009*, 2009, pp. 1–17.

[4] C. Peoples, G. Parr, B. Scotney, and A. Moore, "Context-aware policy-based framework for self-management in delay-tolerant networks: A case study for deep space exploration," *Communications Magazine, IEEE*, vol. 48, no. 7, pp. 102–109, 2010.

[5] E. Birrane and R. G. Cole, "Management of Disruption-Tolerant Networks: A Systems Engineering Approach," in *Proceedings of the SpaceOps 2010 Conference, Alabama, United States. April 2010*, 2010.

[6] L. Z. Granville, D. M. da Rosa, A. Panisson, C. Melchiors, M. J. B. Almeida, and L. M. R. Tarouco, "Managing computer networks using peer-to-peer technologies," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 62–68, 2005.

[7] P. A. P. R. Duarte, J. C. Nobre, L. Z. Granville, and L. M. R. Tarouco, "A P2P-Based Self-Healing Service for Network Maintenance," in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011), Dublin, Ireland. May 2011.*, 2011.

[8] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant networking architecture," RFC 4838 (Informational), Internet Engineering Task Force, April 2007.