Flaw recovery in cloud based bio-inspired Peer-to-Peer systems for smart cities

Bogdan Mocanu[1], Florin Pop[1,3], Alexandra Mihaita (Mocanu)[1], Ciprian Dobre[1,3], Valentin Cristea[1], and Aniello Castiglione[2]

[1]Department of Computer Science, University Politehnica Bucharest, Romania,
[2]Department of Computer Science, University of Salerno, Fisciano (SA), Italy
[3]National Institute for Research and Development in Informatics (ICI), Bucharest, Romania
bogdan.mocanu@hpc.pub.ro, florin.pop@cs.pub.ro,
alexandra.mihaita@hpc.pub.ro, ciprian.dobre@cs.pub.ro,
valentin.cristea@cs.pub.ro, castiglione@ieee.org

Abstract. Internet of Things and Smart Cities concept have become very well spared in the last years especially with the growth of mobile data bandwidth from 3G to 4G networks. This evolution have led to a new type of ubiquitous connectivity through devices in an unconventional manner. The old, client-server approach is not anymore feasible, therefore the Peer-to-Peer concept must be applied. The most common approach in this case represents Cloud based Peer-to-Peer services. This paper presents SPIDER Peer-to-Peer overlay flaws analysis and several requirements for Peer-to-Peer applications considering those flaws.

Keywords: Peer-to-Peer, Bio-inspired Peer-to-Peer overlays, Cyber-physical infrastructures, Flaw management in Peer-to-Peer systems, Cloud based Peer-to-Peer systems

1 Introduction

Peer-to-Peer systems has become an emerging research topic in the last years. Many researchers focuses their attention on Cloud systems based on Peer-to-Peer overlays. This type of system has many applications in various domains like big-data dissemination, trust management [13], Internet of Things [5], smart cities [11], [3] and cyber physical infrastructures [8].

Flaws management in Peer-to-Peer systems represent a major research topic due do its impact in the systems availability. Therefore, the most important the capabilities Peer-to-Peer systems are self-adaption and self-reorganization with minimal resources keeping the functionality of the system.

The Peer-to-Peer overlay that is taken in consideration for this paper is the bio-inspired SPIDER Peer-to-Peer overlay, developed in University Politehnica Bucharest and presented in paper [9]. This overlay is inspired from nature, thus the construction of this overlay is considered to be bio-inspired. The network is constructed similar to spider webs because it has a fixed number of chains and a variable number of rings depending on the size of the system.

The Internet of Things is a new era in the field of computer science research domain. This type of research segment, basically deals with interconnected devices and sensors through a radio network and aggregate the data form those devices in order to provide information that supports the decision making process. One example of the Internet of Things or Internet of Objects is a smart home surveillance system. In this case, based on the information aggregated from different sensors like video cameras, door sensors, window sensors, heat sensors and proximity sensors, the user can be notified in real time in case his house is been robbed and take further legal actions.

In this paper we analyze several flaws in SPIDER Peer-to-Peer overlay and the way that the overlay recovers from those flaws. The rest of the paper is structured in 6 sections. In the second section is presented state of the art review of existing solutions for cyber-physical infrastructures. Section 3 presents the flaws of the SPIDER overlay. Section 4 is about application type for the SPIDER overlay. In this section are presented two type of applications: IoT and live video streaming. The requirements of this type of applications are presented in section 5. Also in this chapter is realized a mapping of application types, flaws and requirements. Finally, in section 6 are drawn the conclusions of this research.

2 Related work

Smart cities and cyber-physical infrastructures have become very attractive subjects of research in the past years. Trust management in bio-inspired Peer-to-Peer systems has many challenges especially coming from the fact that there is no central authority to trust.

The authors of [12] have presented an emergency situation management system based on a hybrid cloud architecture that manages storage and computing resources for command and control activities. The efficiency of the proposed approach is based on the fact that the system is aggregating information from motion sensors correlated with the signal strength from landmark nodes.

In paper [5], the authors present a model for dynamic fault reduction of devices in an IoT environment. One particular example of this modes takes the advantage of the door sensor and the heat sensor in order to replace the functionality of the video camera.

Paper [14] presents TRANSIT an interesting approach for the transition between different types of mechanisms in live video streaming with respect of performance in an highly dynamic environment from fluctuation point of view. The proposed approach was evaluated through trace-based workloads. The experimental results showed the fact that the proposed solution offers good resilience.

Another interesting approach is presented in paper [7]. The authors of this paper evaluated the performances of SPS (Swarm-based Peer-to-Peer Streaming) for video streaming and proposed OLIVES, an ISP-friendly Peer-to-Peer live video streaming solution. The obtained figures for the evaluation of the OLIVES mechanism were obtained through simulation and they demonstrate the fact that the proposed solution is able to deliver high quality video streams over different real scenarios. The use cased taken in consideration in this paper were the excess connection bandwidth and the limited connection bandwidth.

The authors of paper [16] present a very good survey for smart greed communications. In this paper are described several requirements such: QoS expressed in terms of latency, bandwidth, interoperability, scalability, security and standardization.

The authors of paper [1], presents a scheduling algorithm for video broadcasting with maximum stream rate. The hypothesis on their paper is based on the fact that each peer in the system interacts only with a few peers in their neighborhood.

Despite the scientifically contributions of the authors of the papers mentioned above, we have presented and evaluated in terms of performance the auto-recovery mechanisms in case of several flaw scenarios for the SPIDER Peer-to-Peer overlay with direct impact in real Peer-to-Peer systems.

3 Type of flaws in SPIDER Overlay

This section analyses several types of flaw in the SPIDER Overlay and the impact of the reorganization of the logical structure of the system in order to regain functionality.

For a better understanding of the proposed algorithms for flaw recovery we have decided to use the flowing annotations:

- SPIDER overlay with N nodes organized in nr rings and nc chains : $S[nr, nc]$;
- Number of peers in the SPIDER Overlay : n;
- A node in the SPIDER Overlay named by its coordinates in the overlay (chain and ring) : $*[c, r]$;
- Node N neighbor table : $NTable(N)$;
- Upper neighbor of N : $NTable(N).UP$;
- Lower neighbor of N : $NTable(N).DOWN$;
- Left neighbor of N : $NTable(N).LEFT$;
- Right neighbor of N : $NTable(N).RIGHT$;
- Free position in the SPIDER Overlay : $free$;
- Node in the SPIDER overlay fall : $*[c, r].FALL$;
- Create node with empty neighbor table : $NTable(NewNod[c, r]).EMPTY$;
- Delete node form SPIDER Overlay : $*[c, r].DELETE$;
- Entire chain from SPIDER Overlay fall : $*[*, r].FALL$;
- Entire ring from SPIDER Overlay fall : $*[c, *].FALL$.

3.1 Local flaws

This types of flaw affect only locally the structure of the overlay. This means, that the changes, the overlay has to make are small and the effects of such flaws are minor. The availability of the overlay in this cases is not affected.

For the analysis of the local flaws we have taken in consideration a SPIDER Peer-to-Peer overlay with N nodes organized in r rings and c chains. In a full

SPIDER overlay are $N = nr \times nc$ nodes. The maximum number of nodes on a ring is $NoNodesRing = nc$ and the maximum number of nodes on chain is $NoNodesChain = nr$.

One node randomly fall Joining and leaving from Peer-to-Peer overlay systems are very common operations. Therefore, leaving from such system is realized without any announcements. One of the reasons why nodes leave the overlay very often might be battery life failure ar loss of radio connectivity.

The algorithm for one node randomly fall is given in Algorithm 1:

Algorithm 1 One node randomly fall

```
Randomly Node $X[c,r].FALL$
$NTable(X[c,r+1]).DOWN = free$
$NTable(X[c,r-1]).UP = free$
$NTable(X[(c-1)\%nc,r]).UP = free$
$NTable(X[(c+1)\%nc,r).UP = free$
```

The impact in this case is very minor because the structure of the SPIDER overlay is not affected. The only changes that must be made are only locally for the neighbors of fallen node. Therefore, there is computed only one operation/node for four nodes in the overlay resulting a total of 4 operations.

In case of one randomly node fall, the overlay does not need to change in order to reestablish its structure. Thus, there is no recovery method needed for this type of flaw.

Two nodes fall from same chain Another flaw taken in consideration in the SPIDER overlay, might be the falling of two neighbor nodes from the same chain. This type of flaw might happen when the two nodes have a common power supply and it falls. Even though, two nodes fall at the same time, this flaw is still considered to be a local one due to the fact that the structure of the overlay is not affected and the actions that must be done are only local.

The algorithm for two nodes fall from the same chain is Algorithm 2:

Algorithm 2 Two nodes fall from same chain

```
$X[c,r].FALL$ and $Y[c,r+1].FALL$
$NTable(Y[c,r+2]).DOWN = free$
$NTable(X[c,r-1]).UP = free$
$NTable(X[(c-1)\%nc,r]).RIGHT = free$
$NTable(Y[(c-1)\%nc,r+1]).RIGHT = free$
$NTable(X[(c+1)\%nc,r]).RIGHT = free$
$NTable(Y[(c+1)\%nc,r+1]).RIGHT = free$
```

Despite the fact that two nodes from the same chain fall at the same time, this flaw is considered to be a local one. The structure of the overlay is not compromised and the operations that must be realized affect only the neighbor tables neighbor nodes. Therefore, the number of operations in this case increases with 2 from the case of one randomly node failure.

In terms of auto-recovery, the two node fall from the same chain flaw can be reduced in fact to a random node failure, thus the structure of the overlay is not affected and there is no need for the overlay to auto-configure itself.

Two nodes randomly fall This type of flaw appears when two nodes in the SPIDER overlay fall at the same time from various reasons: power supply failure, physical malfunction etc. In this case, the flaw is considered to be local because it does not affect the structure of the overlay. This issue can be easily divided in 2 random node fall flaw.

When two randomly nodes fall in SPIDER overlay can be used Algorithm 1 for booth nodes resulting an 8 operation process for the overlay.

A particular case of two randomly node failure is two closed nodes from the same ring failure. In this case the flaw is transformed in the same manner mentioned above with the flowing exception. The number of operations realized by the SPIDER overlay nodes is calculated as $noOfOperations = oneRandomNodeFailureOperations * 2 - 2$ because the left and right neighbors have already set their positions to be free.

3.2 Global flaws

This types of flaws affect the entire overlay structure. This means that the changes that the overlay has to compute are consistent and the effects of such flaw are huge. When this type of flaws appear the overlay availability is affected.

In this section are presented two self-organization algorithms for the SPIDER Peer-to-Peer overlay for critical disaster scenarios:

- Entire Chain lost;
- Entire Ring lost.

Taking in consideration a network formed by$N$ nodes organized in nr rings and nc chains as shown in Figure 1. Each node is named by its coordinates: chain and ring. The aim of this section is to present how the SPIDER overlay reorganizes it self when the nodes form one chain fall all at the same time or when all the nodes from a ring all fall at the same time. In booth cases the SPIDER Peer-to-Peer network reorganizes itself in order to be totally functional again.

Entire Chain lost Scenario 1 Starting from the scenario overview presented with a n nodes network, the first critical disaster scenario implies the fault of one chain at the same time. For this scenario it is considered that all nodes $N[2, *]$, from Chain 2 disappears at once along with the chain. Therefore, the overlays have to

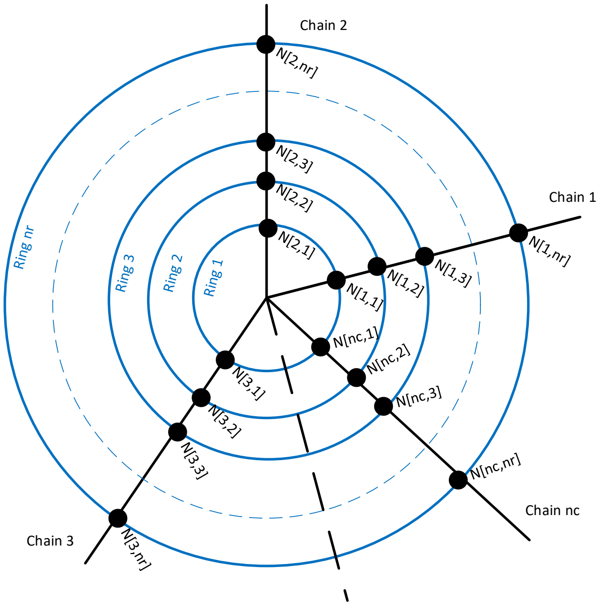

Fig. 1. SPIDER Peer-to-Peer Overlay network organized in nc chains and nr rings.

reconfigure it self from a structure with N nodes and nc chains in one of $N - nr$ nodes and $nc - 1$ chains Figure 2.

In this scenario the reorganization consists of updating the neighbor table of each node on the chain on the left and on the right the lost chain. Thus, every node on Chain 1 updates its left neighbor to the node on the same Ring and Chain 3, and every node on Chain 3 updates its right neighbor to the node on the same Ring and Chain 1.

The impact in this type of flaw is high because about 66% of the nodes of the overlay are affected and the number of the operations that must be computed for the reconfiguration of the SPIDER overlay is given by $noOfOperations = 2 * noOfRings$.

Entire Chain lost Scenario 2 Starting from the scenario overview presented with a n nodes network, the first critical disaster scenario implies the fault of one chain at the same time. For this scenario it is considered that nodes $N[2, *]$, from Chain 2 disappears at once but the chain remains. Therefore, the overlays have to reconfigure it self from a structure with N nodes and nc chains in one if $N - nr$ nodes and nc chains.

In this scenario the self-adaption technique that the overlay adopts is different from the ones mentioned the previous section. The main difference consists of creating 2 new nodes from the remaining ones on the lost chain as seen in

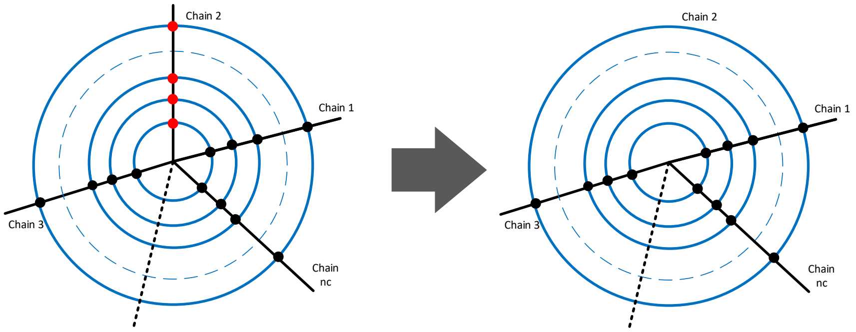

Fig. 2. SPIDER Overlay self-reorganization in case of chain fall scenario 1. The red nodes fall at the same time and the SPIDER Overlay becomes a $nc - 1$ chain overlay with $N - nr$ nodes.

Figure 3. After the creation of the new nodes the neighbors tables must be updated.

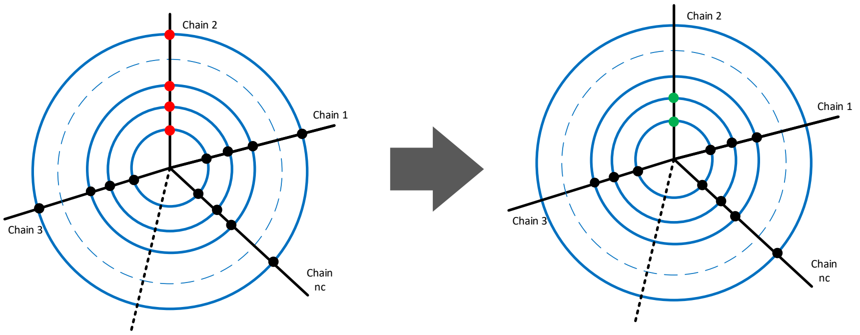

Fig. 3. SPIDER Overlay self-reorganization in case of chain fall scenario 2. The red nodes fall at the same time and the SPIDER Overlay becomes a nc chain overlay with $N - nr$ nodes. The green nodes are the new nodes created from the renaming ones.

For this scenario we have taken in consideration that the structure of the overlay remains the same in terms of number of chains. Thus, when an flaw where all nodes from a chain fall at the same time the SPIDER overlay is reconstructed by keeping as mush as possible of the initial structure of the overlay. The self-recovery algorithm used in this case is presented in Algorithm 3:

Algorithm 3 Entire chain fall keeping the chain structure of the overlay

```
$X[*, r].FALL$
$NewNode[c, 1] = X[(c - 1)\%nc, r]$
$NTable(NewNode(c, 1)).EMPTY$
$NewNode[c, 2] = X[(c + 1)\%nc, r]$
$NTable(NewNode(c, 1)).EMPTY$
$X[(c - 1)\%nc, r].DELETE$
$X[(c + 1)\%nc, r].DELETE$
$NTable(X[(c - 2)\%nc, r - 1]).RIGHT = free$
$NTable(X[(c + 2)\%nc, r - 1]).LEFT = free$
$NTable(X[(c - 1)\%nc, r - 1]).UP = free$
$NTable(X[(c + 1)\%nc, r - 1]).UP = free$
for $i = 1; i < nr - 1; i++$ do
    if $i <= 2$ then
        $NTable(X[(c - 1)\%nc, i]).LEFT = X[c, i]$
        $NTable(X[(c + 1)\%nc, i]).RIGHT = X[c, i]$
        $NTable(X[c, i]).LEFT = X[(c + 1)\%nc, i]$
        $NTable(X[c, i]).LEFT = X[(c - 1)\%nc, i]$
    else
        $NTable(X[(c - 1)\%nc, i]).LEFT = free$
        $NTable(X[(c + 1)\%nc, i]).RIGHT = free$
    end if
end for
$NTable(X[c, 1]).UP = X[c, 2]$
$NTable(X[c, 2]).UP = free$
$NTable(X[c, 2]).DOWN = X[c, 1]$
```

The impact in this scenario is the highest. Not only, the number of updates in this scenario is very high but there must be created 2 new nodes on the lost chain. After this construction, a complex neighbor table update schedule must be effectuated. The number of operations in this scenario is given by $noOfOperations = newNodeCreation(1) * 2 + deleteExistingNodes(1) * 2 + 4 + 2 * newNodesUpdates(4) + (nr - 3)nodeUpdateOfTheNeighborChains(2) + 3$.

Entire ring fall Another frequent flaw in th SPIDER overlay might be the entire ring lost is depicted in Figure 4. This flaw happen when all the nodes from specific ring fall at the same time. This type of flaw is considered to be a global one because the structure of the overlay is affected and the overlay have to auto-recover in a manner that changes its structure.

For this scenario we are taking in consideration a SPIDER overlay network with N nodes constructed by nc chains and nr rings. If a ring from the overlay fall at the same time the overlay auto-adapts by making the upper and bottom connections of the fallen nodes. Therefore in this case the algorithm that might be uses is presented in Algorithm 4.

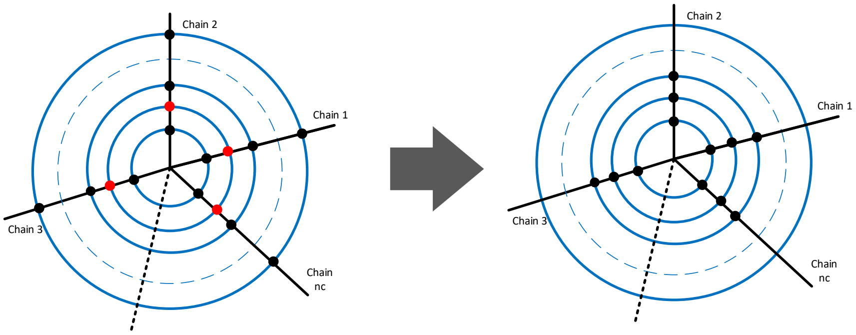

Fig. 4. SPIDER Overlay self-reorganization in case of ring fall. The red nodes fall at the same time and the SPIDER Overlay becomes a nc chain, $nr - 1$ rings overlay with $N - nc$ nodes.

Algorithm 4 Entire ring fall

```
$S[nr, nc]$ with $n$ nodes
$X[c, *].FALL$
for $i = 0; i < nc; i++$ do
    $NTable(X[c, r + i + 1]).DOWN = X[c, r - i - 1]$
    $NTable(X[c, r - i - 1]).UP = X[c, r + i + 1]$
end for
$nr = nr - 1$
```

This type of flaw is considered a global flow because the overlay has to change its structure. The number of operations needed to be executed in order to auto-recover the SPIDER overlay in this case of flaw is given by $c * 2$, because for each chain there must be realized 2 connections between the upper neighbors to the lower ones and all the way around.

4 Type of applications of SPIDER Overlay

4.1 Live video streaming

A particular type of application that is SPIDER peer-to-peer overlay based is live video streaming for video surveillance of a vegetables Farm. The use case of vegetable farming is coming from ClueFarm project [10], developed in University of Politehnica Bucharest.

The use case taken in consideration is a farm formed by 3 farming zone, and each zone consisting in 3 or 4 green-houses. The overlay in this case consists of 3 chains representing the number of farming zones and 4 rings representing the maxim number of green-houses in each zone.

The flaws that might happen in this particular use case are:

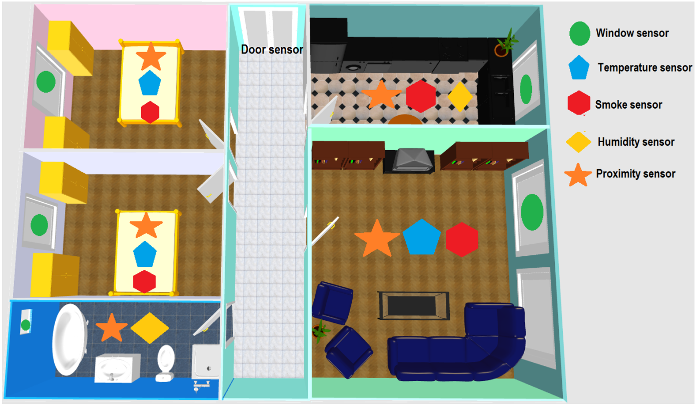

Fig. 5. IoT support use case. Smart-home sensor monitoring. 5 room (1. Kitchen, 2. Living room, 3. Bathroom, 4. First bedroom, 5. Second bedroom).

- One random node fall - in case of a camera malfunction;
- Two random nodes fall - in case weather phenomenon that might affect the green house;
- Two random nodes fall - in case of weather phenomenon that might affect the green houses;
- Entire chain fall - in case farm zones reorganization.

Another use case of live video streaming presented also in paper [8] is river pollution monitoring. This use case is based on [10] developed in University of Politehnica from Bucharest. This use case is based on an logical overlay develop based on the SPIDER overlay. The overlay is formed by 4 chains representing the number of rivers in a specific geographical area, that is monitored and 2 rings representing the number of the observation points placed o that river. Physically the number of observation points vary from the number of tributaries of each river that is monitored.

In this use case the possible flaws that might appear are:

- One random node fall - in case of a observation point failure;
- Two random nodes fall - in case of severe weather phenomenon that might affect the observation points, or blackouts;
- Two random nodes fall - in case of severe weather phenomenon that might affect the observation points or blackouts;
- Entire chain fall - in case of natural hazards like earthquakes or extremely sever weather conditions.

4.2 IoT support for smart houses

Taking in consideration the high interest of the research communities in the Internet of Things, we present a use case for IoT support based on the SPIDER overlay.

The underlay of the IoT support use case is presented in Figure 5.

Therefore, the number of nodes in the overlay is given by the total number of sensors in the house. In this particular case the total number of nodes is 19 organized in a SPIDER overlay of 5 rings and 5 chains. The number of chains is given by the number of chambers in the overlay and the number of chains is given by the maximum number of sensors in a room.

The formed SPIDER overlay is presented in Figure 6.

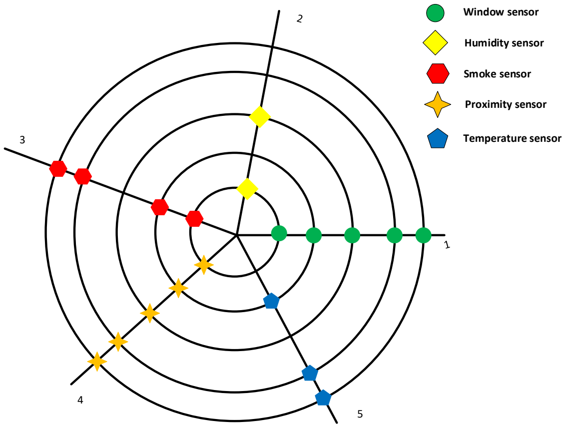

Fig. 6. SPIDER Overlay for the IoT support use case formed by 5 chains and 5 rings. Each ring represents a room and on each chain is only one type of sensor.

Taking in consideration the IoT support use case mentioned above we can present several potential flaws. Therefore, the most common flaw that might happen in a smart home Peer-to-Peer system is the malfunction of one or several sensors. This flaw appears when the sensors have no energy left in their batteries. In this case, the SPIDER Peer-to-Peer overlay flaws, that matches this use case, are the one random node fall or 2 nodes fall. This type of flaws can be frequent and the recovery from this state is realized by replacing the broken sensors with new ones.

Another flaw that might appear in this use case is the entire ring fall. This might happen all the sensors in a room falls at the same time, due to when due to a major problem.

5 Requirements for Peer-to-Peer applications

The requirements of the live video streaming and IoT Peer-to-Peer applications are presented in Table 1.

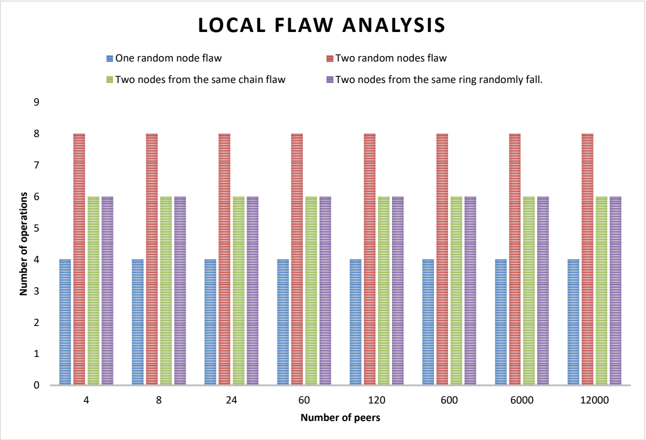

Fig. 7. Number of operations needed for SPIDER self-adaption in case of local flaws.

Table 1. Peer-to-Peer overlay application requirements

Application type/requirements	**Live video streaming**	**IoT Support**
Physical infrastructure	Cable/WiFi	WiFi
Transport protocol	UDP	UDP/TCP
Bandwidth	High	Low
Self-recovery from local flaws	Yes	Yes
Self-recovery from global flaws	Yes	Yes
Throughput	High	High

5.1 Application type - Flaw mapping

The mapping of the application types to the flaws self-recovery features of the SPIDER overlay are presented in Table 2.

5.2 Flaw analysis experimental results

The number of operations for each local flaw is presented in Figure 7.

The number of operations for each global flaw is presented in Figure 8.

The most important observation is the fact that SPIDER overlays organized on rings are fault safer than SPIDER overlays organized on chains.

6 Conclusion and future work

In this paper we have presented several possible flaws of the SPIDER Peer-to-Peer overlay and the way the system auto-recovers from those state. Also were

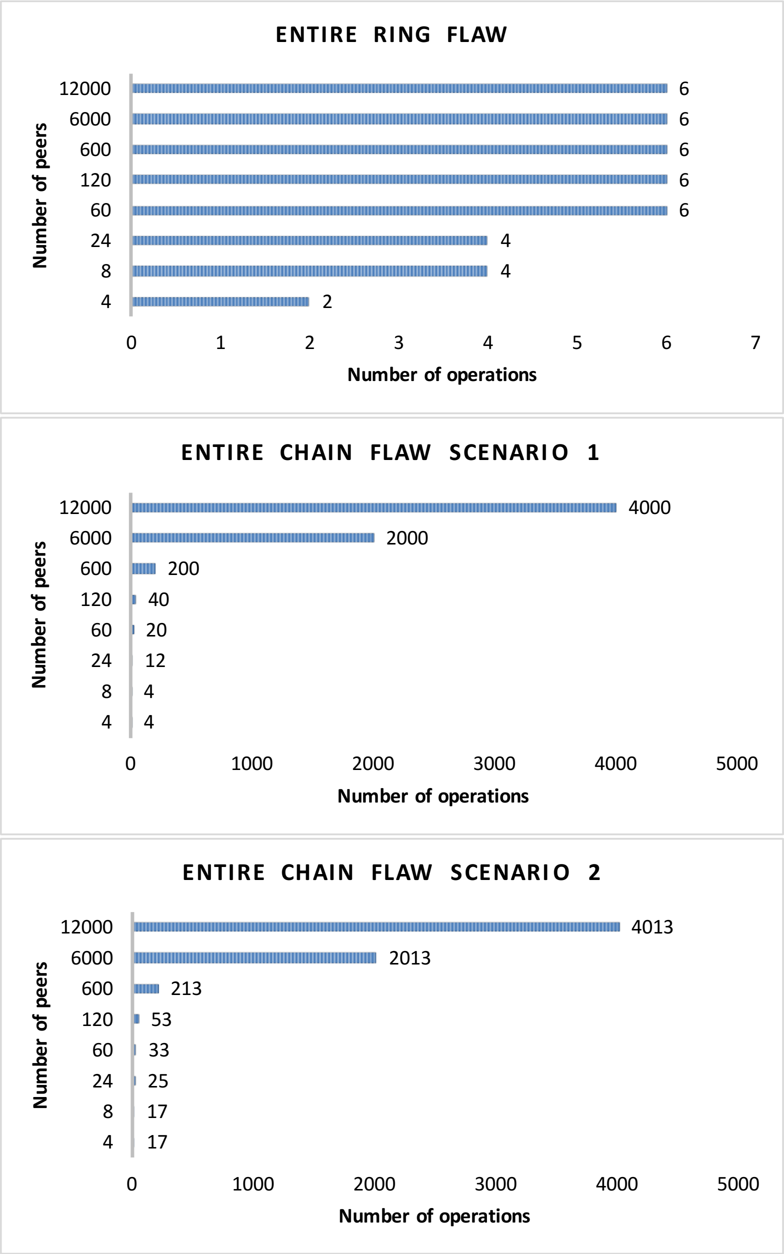

Fig. 8. Number of operations needed for SPIDER self-adaption in case of global flaws.

presented 4 algorithms for fault self-recovery. We have stated the fact that local flaws does not affect the structure of the SPIDER overlay and the only actions

Table 2. Application type - flaw mapping

Application type	**Live video streaming**	**IoT Support**
One random node fault	Yes	Yes
Two nodes fall from the same chain fault	Yes	Yes
Two random nodes fault	Yes	Yes
Two nodes fall from the same ring fault	Yes	Yes
Entire chain fault - scenario 1	Yes	Yes
Entire chain fault - scenario 2	No	No
Entire ring fall	No	Yes

that must be executed is the update of the neighbor tables. In case of global flaws the structure of the overlay is affected and it must self-reconstruct. In case of entire chain fall the overlay will be reconstructed with a decremented number of chains by 1. The same behavior is happening when an entire ring falls, but the recovery process is more efficient.

In future research we will focus on researching redundancy and replication techniques for SPIDER overlay in order to achieve availability.

Acknowledgment

The research presented in this paper is supported by projects: *DataWay*: Real-time Data Processing Platform for Smart Cities: Making sense of Big Data - PN-II-RU-TE-2014-4-2731; *MobiWay*: Mobility Beyond Individualism: an Integrated Platform for Intelligent Transportation Systems of Tomorrow - PN-II-PT-PCCA-2013-4-0321; *clueFarm*: Information system based on cloud services accessible through mobile devices, to increase product quality and business development farms - PN-II-PT-PCCA-2013-4-0870. The research presented in this paper is also supported by project Data4Water, H2020-TWINN-2015 ID. 690900.

We would like to thank the reviewers for their time and expertise, constructive comments and valuable insight.

References

1. Alghazawy, B.A., Fujita, S.: A scheme for maximal resource utilization in peer-to-peer live streaming. arXiv preprint arXiv:1510.02138 (2015)
2. Andersen, M.P., Fierro, G., Culler, D.E.: Enabling synergy in iot: Platform to service and beyond. In: 2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI). pp. 1–12. IEEE (2016)
3. Bojan, V.C., Raducu, I.G., Pop, F., Mocanu, M., Cristea, V.: Architecture design of pattern detection system for smart cities datasets. In: Intelligent Computer Communication and Processing (ICCP), 2016 IEEE 12th International Conference on. pp. 403–408. IEEE (2016)

4. Brienza, S., Cebeci, S.E., Masoumzadeh, S.S., Hlavacs, H., Özkasap, Ö., Anastasi, G.: A survey on energy efficiency in p2p systems: File distribution, content streaming, and epidemics. ACM Computing Surveys (CSUR) 48(3), 36 (2016)
5. Chilipirea, C., Ursache, A., Popa, D.O., Pop, F.: Energy efficiency and robustness for iot: Building a smart home security system. In: Intelligent Computer Communication and Processing (ICCP), 2016 IEEE 12th International Conference on. pp. 43–48. IEEE (2016)
6. Coradeschi, S., Cesta, A., Cortellessa, G., Coraci, L., Gonzalez, J., Karlsson, L., Furfari, F., Loutfi, A., Orlandini, A., Palumbo, F., et al.: Giraffplus: Combining social interaction and long term monitoring for promoting independent living. In: 2013 6th International Conference on Human System Interactions (HSI). pp. 578–585. IEEE (2013)
7. Magharei, N., Rejaie, R., Rimac, I., Hilt, V., Hofmann, M.: Isp-friendly live p2p streaming. IEEE/ACM Transactions on Networking 22(1), 244–256 (2014)
8. Mocanu, B., Mureean, V., Mocanu, M., Cristea, V.: Cloud live streaming system based on auto-adaptive overlay for cyber physical infrastructure. In: Proceedings of the Third International Workshop on Adaptive Resource Management and Scheduling for Cloud Computing. pp. 52–58. ACM (2016)
9. Mocanu, B., Pop, F., Mocanu, A.M., Dobre, C., Cristea, V.: Spider: A bio-inspired structured peer-to-peer overlay for data dissemination. In: 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC). pp. 291–295. IEEE (2015)
10. Mocanu, M., Cristea, V., Negru, C., Pop, F., Ciobanu, V., Dobre, C.: Cloud-based architecture for farm management. In: Control Systems and Computer Science (CSCS), 2015 20th International Conference on. pp. 814–819. IEEE (2015)
11. Morosan, A.G., Pop, F., Arion, A.F.: Electric mobility in green and smart cities. In: International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. pp. 173–184. Springer (2016)
12. Palmieri, F., Ficco, M., Pardi, S., Castiglione, A.: A cloud-based architecture for emergency management and first responders localization in smart city environments. Computers & Electrical Engineering 56, 810–830 (2016)
13. Pop, F., Dobre, C., Mocanu, B.C., Citoteanu, O.M., Xhafa, F.: Trust models for efficient communication in mobile cloud computing and their applications to e-commerce. Enterprise Information Systems pp. 1–19 (2015)
14. Wichtlhuber, M., Richerzhagen, B., Rückert, J., Hausheer, D.: Transit: Supporting transitions in peer-to-peer live video streaming. In: Networking Conference, 2014 IFIP. pp. 1–9. IEEE (2014)
15. Xu, K., Wang, X., Wei, W., Song, H., Mao, B.: Toward software defined smart home. IEEE Communications Magazine 54(5), 116–122 (2016)
16. Yan, Y., Qian, Y., Sharif, H., Tipper, D.: A survey on smart grid communication infrastructures: Motivations, requirements and challenges. IEEE communications surveys & tutorials 15(1), 5–20 (2013)