Annex B (normative):
NFV ISG PoC Report Template

The following normative disclaimer shall be included on the front page of a PoC report:

> *Submission of this NFV ISG PoC Report as a contribution to the NFV ISG does not imply any endorsement by the NFV ISG of the contents of this report, or of any aspect of the PoC activity to which it refers.*

B.1 NFV ISG PoC Report

B.1.1 PoC Project Completion Status

- Overall PoC Project Completion Status: **Completed**

B.1.2 NFV PoC Project Participants

Specify PoC Team; indicate any changes from the NFV ISG PoC Proposal:

- PoC Project Name: Availability Management with Stateful Fault Tolerance.
- Network Operator/ Service Provider A: AT&T
  Contact: Percy Tarapore, pt5947@att.com; Al Morton, acmorton@att.com
- Network Operator/ Service Provider B: iBasis – part of KPN
  Contact: Richard Xu, RXu@ibasis.net
- Network Operator/ Service Provider C: NTT
  Contact: Eriko Iwasa iwasa.eriko@lab.ntt.co.jp; Atsuyoshi Shirato shirato.atsuyoshi@lab.ntt.co.jp
- Manufacturer A: Stratus Technologies, Inc.
  Contact: Ali Kafel, Ali.Kafel@Stratus.com; Pasi Vaananen, pasi.vaananen@Stratus.com (technical)
- Manufacturer B: Aeroflex
  Contact: Mark Lambe, Mark.Lambe@aeroflex.com
- Manufacturer C: Brocade
  Contact: Yue Chen, cheny@brocade.com
- Manufacturer D: Allot
  Contact: Adi Mendel amendel@allot.com

Due to project time constraints, we were unable to complete the successful on-boarding and integration of the Allot DDoS security solution. Therefore, it was mutually decided that this part of the PoC was altered with another VNF option, and further integration efforts would be postponed to potential future collaboration.

We attempted to identify a comparable DDoS application that was deployable in NFV cloud environment, and/or virtualised on KVM, but could not locate any that was or could have been made available within the PoC schedule, as most DDoS application implementations were only available as physical boxes. As an alternative route, we utilized two chained instances of Brocade router application, with one of them providing the stateful firewall function, and another performing per client maximum access rate enforcement to the video server application (which was acting as a reduced functionality proxy for the intended DDoS detection and enforcement function). We also had the Aeroflex virtualised test application for availability related path performance tests and demonstrations, and virtualised open source video server for provisioning of the demonstration streaming video traffic in the PoC service chain, for total of four virtualised functions in each of the three different availability mode service chains.

The VNF substitution allowed the project to stay on the committed schedule. The POC team does not consider the key objectives of demonstration and qualification of availability characteristics of fully stateful fault tolerance without any application level mechanisms (“FT” mode) and VM instance high-availability (“HA” mode) to be compromised by the change. In fact, some additional tasks that were not within the initial scope of this PoC were also implemented - we

were also able to capture detailed measurements to better characterize the time spent in associated fault management sub-processes and also characterized the repair times (i.e. time to restore redundancy).

B.1.3 Confirmation of PoC Event Occurrence

The PoC was demonstrated at two separate events, as well as at many ad-hoc meetings and events.

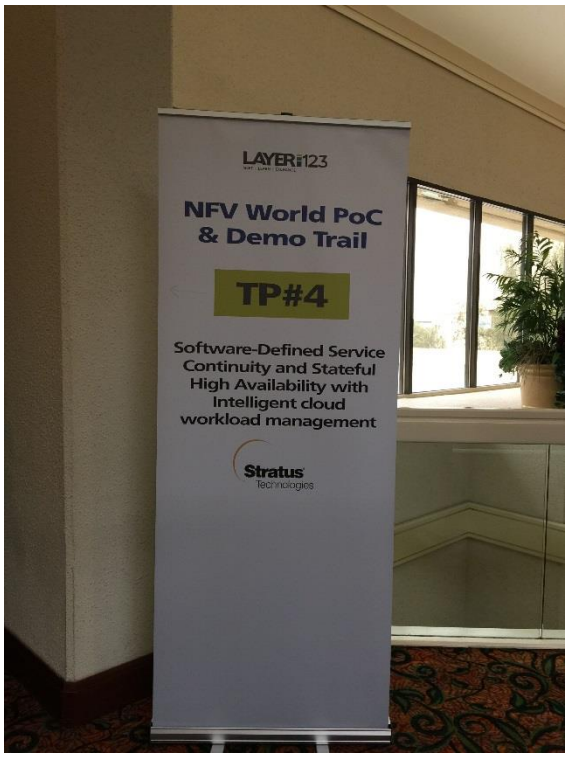
The two events at which the PoC was demonstrated were:

Demo Event #1: NFV World Congress 2015

The first public demonstration of the PoC took place in the NFV World Congress, which took place on May 6-8, 2015 in San Jose, CA. Prior to the event, we issued a press release inviting any and all attendees to come see the demonstration. The demonstration of the PoC along with the associated explanations of the configurations and what was being demonstrated took ~30 minutes per demo session. Over 20 demo sessions were conducted, and over 100 participants representing diverse audience consisting of representatives of service providers, press, NFVI vendors, HW vendors, MANO vendors, VNF vendors and our PoC partners show the demo.

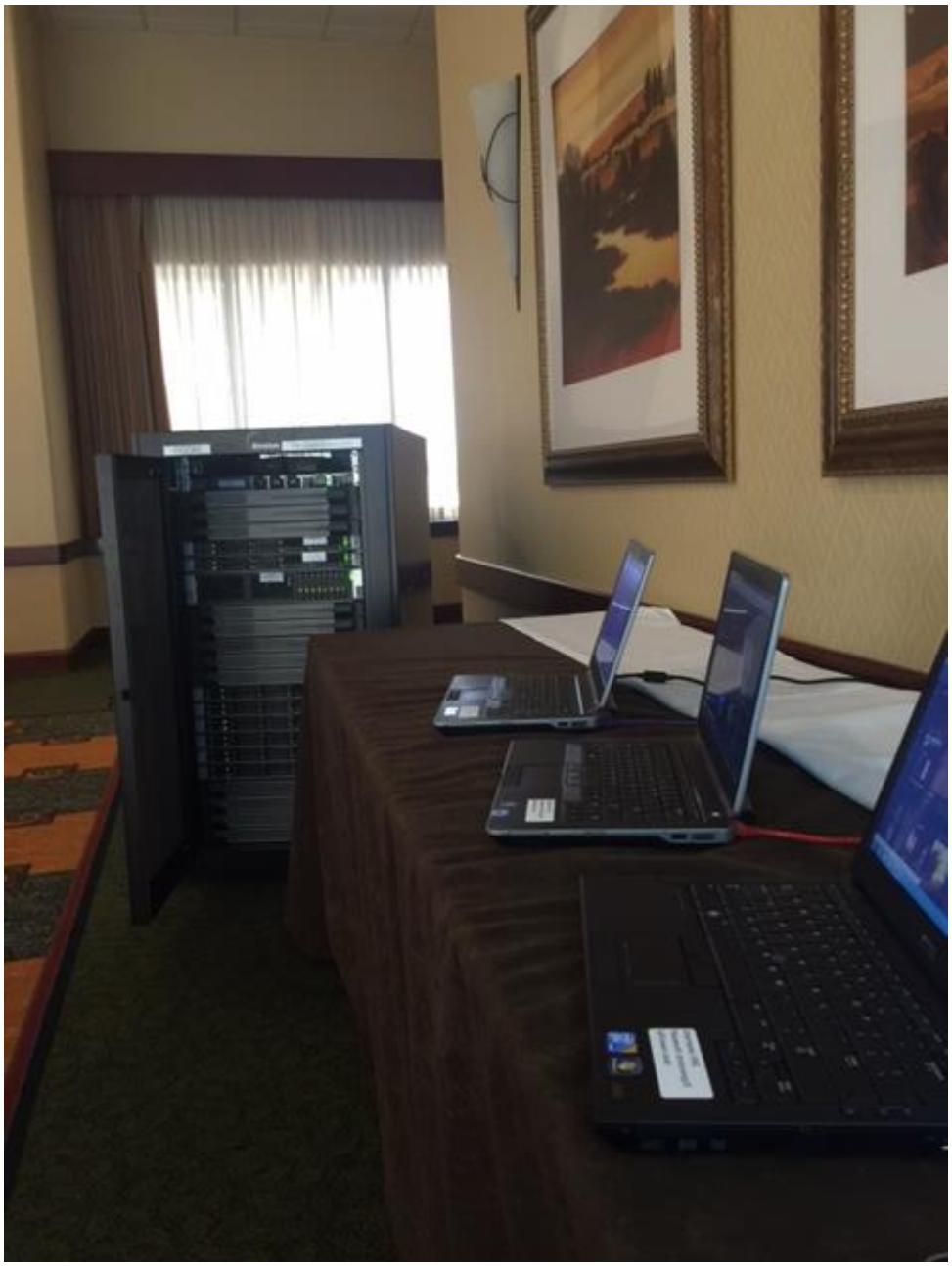
As per the PoC proposal, the demonstration consisted of qualitative demonstration of the streaming video application with server deployed in virtualised infrastructure with access through the service chains in different availability modes. Demo observers could see what the impact of the various resiliency services was to the video application, and we also demonstrated the measured performance characteristics of each of the three separate service chains primarily to show the quantitative values of the observed faults impacts on the service chain basis. The quantitative part was demonstrated using the Aeroflex test application and was observable in real time through the Aeroflex GUI application, which was observable on the projector screen along with OpenStack (Horizon), Stratus workload services and other associated monitoring GUIs to help observe the system state transitions when the simulated faults were injected on the configuration.

For this demonstration, we brought the whole demo rack to the show, which allowed us to pull power cords etc. to simulate faults in the NFVI server components hosting the VNFs in the service chains. Due to power and physical size needs for the demo equipment as well as to accommodate the participants, we had a separate room for the PoC demonstration purposes in NFVWG venue in the Silicon Valley Room, 2nd floor, DoubleTree Hilton, San Jose, California, instead of showing the demonstration in Stratus booth.



Picture #1, above shows the sign of the PoC demo immediately next to our PoC demo room entrance in NFVWG.

Picture #2, Aaron Smith from Stratus Technologies conducting one of the demo sessions in VNFWG venue. This view also shows the control and demo laptops used for qualitative demo, and server rack hosting the demo cloud.

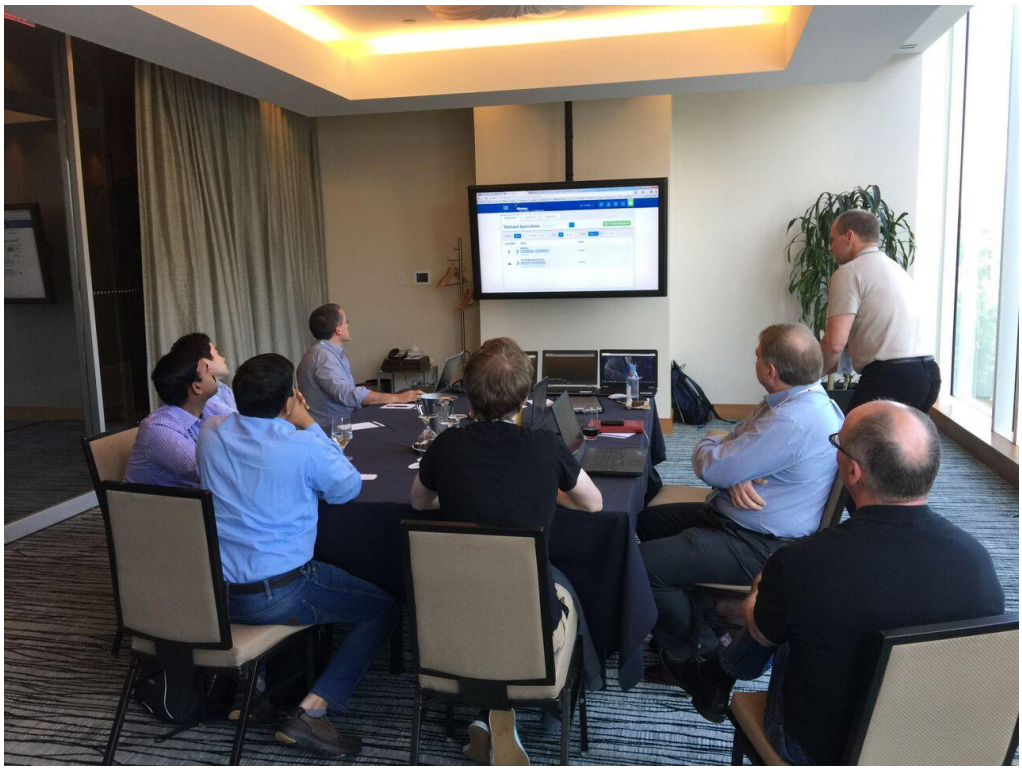


Picture #3 – demo setup HW showing details of the rack with controller, storage and compute nodes for demo clouds in the background, and three laptops used to show what happens to streamed video sessions in each of the three availability modes (FT, HA, and GA) that were demonstrated as part of the PoC.

Picture #4, demo session participants observing the demonstration during one of the many demonstration sessions in NFVWG.

In addition to the demonstrations, the PoC was subject to many in-depth discussions on the Stratus booth, Ali Kafel from Stratus also presented an associated talk on day 3 of the congress, titled “Ensuring Availability and Resiliency Amid the ‘cloudification’ of Telco”.

Demo Event 2: OpenStack Summit Vancouver, 2015



Picture #5: demo session in progress in the OpenStack summit.

For the OpenStack summit, we had a demonstration room set up in the hotel immediately next to the OpenStack summit conference venue. Due to constraints of the room, instead of bringing the demo rack to the venue, we conducted the demonstration utilizing the PoC setup in the Stratus lab in Maynard, MA remotely through the secure internet connection. To simulate faults, we utilized remotely controlled power strip to simulate node power failures, along with the remote access to compute resources to simulate other failure modes (such as hypervisor, VNFC, etc. faults). The access to the test application and other GUIs for monitoring was also through the remotely access connection from laptops on site.

B.1.4 PoC Goals Status Report

This PoC focuses on demonstration of how multiple VNFs from multiple vendors can be deployed in an NFVI+MANO environment that enables deployment, monitoring and control of these VNFs in a variety of availability modes including Fault Tolerant (FT), High Availability (HA) and General Availability (GA).

The salient characteristics of the availability modes in terms of the key fault management cycle phases are as follows:

FT (Fault Tolerant): redundant VM instances, full-VM state checkpointing, NFVI fault detection, fault recovery with fully stateful VM failover to redundant entity. Repair (Redundancy Restoration) through re-instantiation of redundant instance & warm-up (state replication), followed by a move of the new redundant instance to standby state.

HA (High Availability): no VM instance redundancy, no infrastructure provided VM state checkpointing, NFVI fault detection, fault recovery with re-instantiation. Repair is not applicable (recovery & repair operations are the same).

GA (General Availability): no VM instance redundancy, no infrastructure provided VM state checkpointing, NFVI fault detection, fault recovery through clean up only (i.e. no automatic recovery, will require external intervention). Repair is not applicable (requires either external SW agent request or manual intervention to re-instantiate).

The list of the specific, enumerated PoC goals is given below.

- **PoC Project Goal #1:** Demonstrate instantiation of VNFs from multiple vendors onto a simple static NFV Forwarding Graph (emulating the configuration that is analogous to instantiation of corresponding physical NFs onto functionally equivalent service chain
  - **Goal met & demonstrated**; however, as stated in the section B.1.2 it was agreed to replace the VNF of the DDoS solution with a different VNF, implementing two instances of Brocade router in differing configurations, but in addition we had Aeroflex test application and video server VNFs in the configuration, so we consider that this PoC still meets the "multiple vendors" part of the objective as stated.

- **PoC Project Goal #2**: Demonstrate the correct (i.e. constraints compliant) placement of redundant VNF (or VNFc) instances based on specified deployment anti-affinity attributes by MANO/VIM services.
  - **Goal met & demonstrated**

- **PoC Project Goal #3**: Demonstrate simultaneous instantiation of multiple VNFs in a mix of three different availability modes: FT, HA and GA (as described in the introductory portion of this section).
  - **Goal met & demonstrated**

- **PoC Project Goal #4**: Demonstrate the feasibility of support for VNF sparing / redundancy and full VM state checkpointing as NFVI + MANO provided services (transparent state checkpointing for FT availability mode only, storage-based state replication for any availability mode).
  - **Goal met & demonstrated**

- PoC Project Goal #5: Demonstrate automated detection and recovery of failed VNFs, including associated NFVI network reconfiguration mechanisms as NFVI + MANO services (for FT and HA availability modes).
  - **Goal met & demonstrated**

- PoC Project Goal #6: Quantify the service availability performance metrics demonstrating both service accessibility and service continuity aspects for each of the different availability modes (downtime performance metrics per event, measured from service client's perspective).

  - **Goal met & demonstrated**

- PoC Project Goal #7: Demonstrate the qualitative service availability performance aspects with different availability modes through the use of suitable end-to-end applications over the instantiated service chains upon injection of failures (e.g. failure event impact to streaming server-client audio/video application for each availability mode).

  - **Goal met & demonstrated**

B.1.5 PoC Feedback Received from Third Parties (Optional)

Here are some of the select quotes we have received as part of the PoC demo's and associated discussions with both the operators as well as with the VNF software vendors:

- "One of the reasons we have not done network wide deployment of NFV (just trials so far) is because of the stateful HA problem that we need to solve. It looks like you have solved this for us." Head of Core & Service Networks, A Tier 1 European Telco
- "All of our customers require redundancy and expect it. We are struggling as a group to make this a reality into clouds and we lack the time and the skillsets to create this. The existing redundant capability (in our non-cloud products) took years to get right and is not portable to the the Cloud. You have saved us a lot of time and complexity" VP of Engineering, IMS vendor
- "I am not saying you are the Jesus, but you are the only one that is addressing this issue. I have been talking to several big name vendors and nobody else has solved this problem that you have solved. I am very interested!". Head of Network Strategy, Network Operation, A Tier 1 European Telco
- "Of all of the things I have seen here, this (your software-based fault tolerant cloud product) is the most impressive" by a leading services standard member of a Tier 1 US Telco, at the NFV World PoC & Demo Trail

B.2 NFV PoC Technical Report (Optional)

PoC Teams are encouraged to provide technical details on the results of their PoC using the PoC Scenario Report template below.

B.2.1 PoC Scenario Report

Objective Id:	**Scenario #1.1** – VNF description at different availability levels	
Description:	In this scenario, VNF descriptions are created to describe the VNF on-boarding related requirements and constraints, including Availability Modes. This involves generation of the VNFD which includes the information required to onboard the VNFs and their constituent components in the instantiation phase (see following scenario). VNFD's that are otherwise equivalent in functionality will be created for the FT, HA and GA availability modes. In the instantiation step (next scenario) these VNFDs are used to instantiate VNFs on the NFVI resources.	
Pre-conditions	OpenStack environment up with Stratus availability services installed in compute nodes, and Stratus workload services installed on top of OpenStack. VNF binaries available in the compatible deployment format (or converted to compatible format).	
Procedure:	**1**	VNF binaries were brought on-board to the cloud image storage for each of the VNF components.
	2	VNF description package was created through the Stratus authoring tool, which allows modelling of the workload aspects. Items that were configured for each VNF instance include VNF binaries, configuration files, internal interconnects and external interfaces, cloud instance types and other resource requirements, and availability modes. The resulting description forms the VNFD for each component instance, including the availability mode configuration ("FT", "HA", and "GA" for this PoC). One description was created for each VNF in each of the availability modes it was intended to be deployed in.
	3	Resulting VNF descriptions were stored on VNF catalog for use in the deployment phase.

Results Details:	Objective was successfully demonstrated.
Lessons Learnt & Recommendations	While it is currently possible to model VNF internals in various ways, the issue is the lack of the agreed on **standard** for the vendor neutral VNFDs and VNF packaging in general. This requires VNF vendors and infrastructure SW vendors to work co-operatively to construct deployment package to each vendor specific environment (of which there are at least 10 different versions around), which is both labor and cost intensive process. We view this as one of the largest currently remaining obstacles toward the availability of interoperable, MANO vendor neutral VNFs. Overall, we had to perform relatively significant amount of engineering effort with participating VNF vendors to get to the deployable packages, while this should be eventually limited to addressing any issues encountered from onboarding the vendor supplied deployment packages. This has been confirmed to be a common issue by the operators participating on the PoC, as well as in the operator and VNF vedndor discussions in general. One operator stated that if VNF has not been on-boarded and described on a specific vendor's NFVI+MANO environment, it can take up to 6 months to get it in, per associated MANO vendor's statement related to deployment of "non-supported" VNFs. While phase 1 MANO document did a good work in laying groundwork to many key aspects of the VNFDs, it is not at the level that is required for the interoperable deployable standard packages. NFV ISG IFA working group is still working on the associated requirements, while there is also parallel work in OASIS TOSCA to start addressing this, and Open Source projects such as Tacker and template translator are also working on some relevant aspects. Unfortunately addressing this is not currently in the formal scope of the OPNFV project, which stops at the VIM northbound interface. TOSCA NFV profile is starting to address some reliability & availability related aspects, but much more work is required in VNF deployment packages in general and in aspects related to be able to fully express the redundancy configurations and other R&A mechanisms in particular.

Objective Id:	**Scenario #1.2** – VNF instantiation at different availability levels	
Description:	In this scenario, VNFD's created in the scenario #1.1 above are used to place and start the VNFs (and their associated services) within NFVI infrastructure. The VNFs are placed according to their deployment requirements, including the constraints and requirements associated with the support of the availability modes.	
Pre-conditions	Successful completion of Scenario 1.1.; VNF descriptors and binaries available in the cloud.	
Procedure:	1	Descriptions for each VNF deployment package created in scenario 1.1 are identified and deployed
	2	Inter VNF and external connectivity topologies are modelled through the Stratus Workload Services GUI and instantiated. Note: while this functionality is intended to be performed by orchestrator through northbound API, as we did not choose to do orchestrator integration on this PoC, we used the GUI to exercise the associated API calls, which essentially means that the GUI application was acting as a proxy for the full orchestrator in this PoC. All of the associated functionality is supported through APIs.
	3	After all the topologies are described and all VNFs are instantiated, the PoC is fully deployed and ready to be brought in service.
Results Details:	Objective was successfully demonstrated. After installation all connectivity is in place and VNF component instances were instantiated in the test cloud in anti-affinity configurations as required to support their specific availability mode level (e.g. fault tolerant instances are placed in the different physical servers).	
Lessons Learnt & Recommendations	As in the scenario 1.1., standards and/or agreed on open source implementations are required for the APIs in this area. However, this gap is not as severe as the gap in 1.1., as this only primarily affects VNFM/Orchestrator interfaces, so there is smaller amount of combinations affected by this gap.	

Objective Id:	**Scenario #1.3** – VNF configuration and configuration store to externalized persistent storage	
Description:	In this scenario, VNF instances are configured sufficiently to provide the associated VNF service through its VNF specific element management interface(s). VNF configuration is written to persistent storage, which is associated with the VNF instance, and this configuration information is expected to be available and used when the VNF is re-started or re-instantiated (i.e. restart should not require re-configuration). This demonstrates partial VNF state (i.e. configuration / management state only) persistence through externalized state storage (in disk).	
Pre-conditions	Scenario 1.2 successfully completed, VNFs instantiated and started	
Procedure:	**1**	For each application, the configuration to the associated target deployment model / mode in the context of the PoC was created and stored through the VNF specific management interfaces (which were all different, and utilized vendor specific GUIs, CLIs, configuration files and APIs). As expected these processes were different for each vendor (as they would be for the currently available physical or SW elements), and as such detailed descriptions here are likely not beneficial in the context of the overall PoC objectives, and are therefore details are omitted from here.
	2	The external physical network elements and associated overlay networks were configured to statically associate the traffic from each of the client laptop and virtualised video server entity to three different availability mode VNF chains to ensure that the availability performance could be simultaneously demonstrated for each of the three deployed availability modes. This would typically be accomplished through the authentication based policies in real network, but static configuration was simpler to configure and use, and was determined to be adequate for the objectives of the PoC. Similarly, the Aeroflex Virtual test application instances were configured to force the test traffic through each of the three availability mode paths, enabling simultaneous and independent measurements to be taken along each of the three distinct paths.
Results Details:	Objective was successfully demonstrated; after all elements were properly configured in the PoC environment context, we had a set of working instances, which were deployed in our PoC cloud.	
Lessons Learnt & Recommendations	As expected, this was a labour intensive process due to variety of complex configuration models and interfaces needed. Some configurations required combination of instantiation specific configuration files, scripts etc. to be created. While this is not necessarily different to the state of network element configurations in the present physical deployments, more standardization of interfaces and APIs for the configuration modelling could be greatly beneficial to streamline the process in NFV environments. Less vendor specific Interfaces / APIs would be a great start for making progress here, but those alone are not sufficient to fully address the issues (configuration object level standards with standardized syntax and semantics would be additionally required to reduce the differences in similar configuration tasks). In the NFV environment, it should also be possible to deploy initial instantiation specific configuration through the configuration specified (i.e. called in rather than included in, as this is instantiation specific rather than related to the generic VNF instantiation process and therefore can be different for each VNF deployment instance) in the VNF deployment packages. VNF vendor support was key to success in this phase, and we thank our PoC vendor participants for their help here.	

Objective Id:	**Scenario #2.1** – Resiliency of the network service in a Fault Tolerant (FT) VNF deployment model with fully stateful VNF VM redundancy. This means keeping a duplicate copy of full virtual machine state (and implicitly all session state) so that if a failure happens, the secondary VNFC will have a replica of the full VM state and will continue providing the service as if a failure never happened.	
Description:	● In this scenario two VNFs are deployed in FT mode and associated VNF VM state is maintained at all times ● Resiliency is demonstrated in three different scenarios a. Kill the individual VM process (VNFs/VNFCs) b. Kill the OS and/or hypervisor (all instantiated VNFs/VNFCs) c. Kill the hardware server (all instantiated VNFs/VNFCs) ● Availability metrics (accessibility and continuity metrics) are automatically monitored by test application through the injected fault scenarios and associated performance parameters are recorded. ● For qualitative availability performance monitoring, the quality of the application performance is monitored and any observed performance impacts (including no noticeable impact) are recorded.	
Pre-conditions	VNFs deployed in "FT" mode and passing traffic, after completion of scenario 1.3.	
Procedure:	**1**	For qualitative assessment, we used video server and client pair, which allowed observation of the effect on this demo application in the PoC demonstrations, along with the Aeroflex traffic monitoring application instances configured to send test traffic through this path to be able to monitor the path downtime performance in real time.
	2	For quantitative assessment of the downtime, we passed test packets through the path at the rate of 10k packets per second (giving us 0.1ms time granularity for the associated outage measurements), and used the time from the 1st lost packet to the 1st subsequent packet received after the fault and associated recovery processes to measure the time the path was out of service for each injected fault.
	3	For qualitative tests in demo configuration instances / nodes deployed in "FT" configuration, each of the three scenarios outlined in the Description above were performed in some of the demos. In public PoC demo scenarios, we were mostly using the approach c.) – i.e. kill the node(s) through power cut-off, which was a convenient way to allow observation of the effects on the video application as well as the path packet transfer outage performance simultaneously, allowing the PoC demo audiences to get an idea of relative outage performance of each of the three modes demonstrated.
	4	For quantitative tests in demo configuration instances / nodes deployed in "FT" configuration, each of the three scenarios outlined in the description above were performed, and downtime performance results were recorded.

Results Details:	Objectives were successfully demonstrated. Going beyond the initially stated PoC objectives, we also added some more detailed measurements to better characterize the time spent in associated fault management sub-processes and also characterized the repair times (i.e. time to restore redundancy). **Qualitative assessment:** For the video application demonstration, we used a laptop with video playback application to provide a convenient and hands-off way to demonstrate the outage impact in the context of the real-world application that most observers are familiar with. As expected, the video stream playback continued without perceptible degradation through the brief duration of the chain outage associated with the interruption. While there was measurable transfer interruption during the failover process and associated reconfiguration, the outage period was brief enough that it was not perceivable (even though we purposely minimized the playback buffer sizes on the client applications). As expected, the TCP sessions did persist over this outage, as the session state that was dynamically instantiated on the session stateful elements (primarily stateful firewall) was retained through the associated VNF component instance failure in its associated redundant stand-by instance. This met our PoC objective to demonstrate that both the very high *service accessibility* can be retained in this availability mode, *service continuity* can also be fully ensured with these mechanisms for the stateful VNF component instances. **Quantitative assessment:** We measured the outage performance for each of the three scenarios, as listed in description above. In addition to measuring just pure outage length which was the stated objective of the PoC, we did analyse the key phases of the recovery process to gain insight on the time allocation of these phases in the PoC configuration. Specifically, we characterized the following aspects of the Fault Management (FM) cycle: 1. fault detection time (specified here as the time from fault activation to the first fault indication from the fault detectors), 2. fault manager VM instance recovery time (time elapsed from the fault indication to standby instance being ready to provide service), and 3. network reconfiguration time: time elapsed from the network reconfiguration request to completion of the associated NFVI physical + overlay network connectivity reconfiguration to switch from previously active to the new active (previously standby) VM instance. The results for each of the three scenarios for FT mode were determined to be as follows: a.) **kill process** (kill -9 <qemu PID>): total downtime 416.5ms; detection time 123.6 ms; fault handler time 107.6ms; network reconfiguration time 185.2ms b.) **host OS crash** (echo c >/proc/sysrq/-trigger): total downtime 450.9ms; detection time 123.6 ms; fault handler time 113.6ms; network reconfiguration time 213.6ms **c.) Host Fail** (power removed): total downtime 439.7ms; detection time 123.6 ms; fault handler time 98.4ms; network reconfiguration time 217.6ms For the direct link failure, which was carrying VM state checkpointing traffic, we observed fault detection time of 26ms (and associated reduction of overall total FM cycle time to 293ms), which is an example

	indication on processes that can further accelerate the FM cycle. Ideally asynchronous interrupt driven detectors can be utilized for many of the infrastructure HW fault detection cases, as is the common case in the present “box” designs. In “FT” mode, instance’s persistent storage (if used) is considered to be part of the protected, full VM state. We used node-local storage for the “ephemeral storage” (i.e. storage which lifecycle is expected to persist through the associated VMs lifecycle) in PoC, which becomes *protected* ephemeral storage in this mode, and therefore will affect the time it takes to instantiate a new node (for fault scenarios that require it) in a new server and bring it to the fully synchronized state (including any associated storage state) with the instance that is providing service after failure. To characterize how long this repair process takes, we also measured the time for the repair (i.e. re-instantiation of redundancy) process, including full node-local storage synchronization. The associated times are: a.) **Kill process**: 63 seconds b.) **Host OS crash**: 548 seconds c.) **Host fail**: 525 seconds It is important to understand that these are NOT outage related times, but rather **repair** times – i.e. service is not affected, but this is the maximum time the system is subject to second failure that cannot be fully recovered. This time can be largely eliminated from node level consideration in the deployments that utilize shared storage for all purposes, including for the ephemeral storage. Obviously, if the instances do not utilize storage, this time can also be eliminated. The short repair time on the “process kill” scenario above is attributable to the fact that as the failure mode was determined by the fault manager to be constrained to a single VM instance rather than full node, the fastest repair action available (which was taken) was the re-instantiation of the single failed instance on its original location, which did not require hypervisor reboot and simultaneous instantiation of multiple affected VMs, which would be needed on full node failure case.

Lessons Learnt & Recommendations	As per the objectives of this scenario, we have successfully demonstrated that the VM based state replication can be used to ensure both high *service continuity* and *service accessibility* as NFVI services, and without the application awareness. This also implicitly demonstrates that all phases of the fault management cycle (fault detection, fault localization, fault isolation, fault recovery and fault repair) **can** be provided as infrastructure services (i.e. using a combination of NFVI and MANO level mechanisms). In addition to the fault management processes, we also demonstrate that the full VM stateful replication and recovery, which are required to ensure service continuity can also be provided by infrastructure – all without any application (i.e. VNF) level support mechanisms. Fully stateful VM replication implicitly enables fully stateful failovers for all aspects of the protected VMs, including but not limited to the guest OS state, protocol session adjacencies including TCP sessions, and full protected VNFCI guest application state. In addition to VM state protection (including block storage protection, when used), implementation guarantees globally consistent state through queuing and gating the release of outgoing protocol packets until acknowledgement is received from the standby instance that associated VM state checkpoint has been successfully received. Associated mechanisms and potential implementations are discussed in more detail in REL002 work item. We'll examine the various aspects related to the VM stateful fault tolerance service below, and propose further future study items. **Downtime budgets:** The service availability for an associated instance pair can be calculated to be over 7NINES, assuming the generally very conservative (i.e. high) failure rate of once per year for all causes, and 500ms maximum outage time. Due to comparatively very rapid repair processes enabled by homogenous resource pools in cloud (in seconds instead of hours), the second failure exposure risk can also be effectively minimized, while at the same time relaxing the constraints of physical repair times, allowing deferred/batched maintenance processes instead of critical 24x7 maintenance with tight (e.g. 4hour) total repair cycle time targets. We are also confident that the recovery and repair (i.e. full redundancy restoration) times can be both improved over the values measured in this PoC. 500ms failovers are expected to be sufficient for many (or even most) telco applications – recall that "5NINES" availability with one incident per 12 month period would mean ~315 seconds of **service** (or often network element) un-availability ceiling target per year. Even with many VNFs in chain and many VNFCIs in each of the VNF's chained internally (which is expected to be typical of service composition in VNF), 500ms is just a small fraction of the un-availability "budget", as it needs to be for supporting service level objectives. There are also many systems and services that VNFCIs depend on, all of which are subject to the limits of the total available downtime budget, and no single system or subsystem can assume availability of ALL of the downtime (examples of such systems and subsystems that VNFCIs implicitly or explicitly rely on are facility power and cooling, node power and cooling, NFVI infrastructure physical and overlay networks, operating systems, hypervisors, storage systems, SDN controllers, messaging and management services, etc.), in addition, applications (VNFs) themselves as well as operations related incidents need an allocation for a portion of total downtime. Additionally, one of the stated goals in NFV (as described in the phase-1 SWA document) is that infrastructure services perform "fault masking", meaning that the fault recovery processes take place autonomously, and without application awareness. Fault masking

	generally requires that the FM cycle will complete in the timelines that are lower than the application response to the fault. While timelines of application reaction are clearly application specific, there are known applications that require 100ms and less failover times. We believe that this is possible from the detection and baseline fault manager perspective, but getting to 100ms total FM cycle time may require tighter integration (possibly with pre-computed and/or pre-instantiated failover paths with fast switchover) with the NFVI network to optimize the network reconfiguration times. We will therefore plan to keep the optimization of the FM cycle for even faster cycle time as a future work item. It should be noted that the many types of fault detectors will have a potential risk to become more susceptible for the false positives when the detection time is reduced, therefore the optimization approach is to minimize the time for the subsequent phases of the fault management cycle, and have detectors to be the primary determinant on the total cycle time. This allows the detectors to be selected and/or configured to meet the application downtime targets while allowing application developers to make the appropriate tradeoffs on the false positive sensitivity. **Recommendations for future work (downtime):** 1.) NFV REL working group should study the downtime allocations in the realistic NFV configurations, and provide guidelines for overall downtime as well as key subsystem allocations (including allocations for the applications and maintenance actions). This could be based on the modelling efforts of REL003, and associated performance metrics should be described on the newly established REL QAF work item. Ideally, in addition to defined metrics and use of prescriptive models we would establish target downtime allocations / reaction times for key subsystems that can be used as a guidance for subsystem developers (such as in OPNFV project). 2.) Present NFV work assumes that the VNFM will be primarily responsible for the fault management actions. This is good approach for the VNF related aspects of the FM cycle, but in real implementations there will be multiple layers of co-operating fault managers, with increasing scope of actions that can be taken by layer, but with also increasing time to take actions, so we will have a hierarchy of fault manager layers. This structure allows the actions to be taken on the lowest layer that can perform the recovery, which generally results in the best performance (i.e. fastest recovery). What is missing from the current NFV documentation is state change events not only faults (this is covered) but also for the fault management actions – this is required mechanisms for both avoiding possibly conflicting multi-layer actions at the various layers, as well as an escalation mechanism when lower layer cannot successfully perform the recovery action. These state change notifications should be added to the present FM messaging requirements. This may be potentially coupled with multi-layer policy management efforts, if the assumed implementation is expected to rely on FM policies implemented at multiple layers, which has been suggested by some. **State Replication and resource use** The key benefit of full VM state protection mechanism is that it protects complete machine state, and does not require application modifications to do so (i.e. application level checkpoints are not needed as everything gets replicated). To minimize the resource use of the state replication mechanisms, there are multiple possible avenues of varying complexity tradeoffs, such as:

	➢ **De-clustered redundancy**: - this approach takes an advantage of a specific property of the replication scheme, which is that the standby instances are not running, and only receive state updates, which consumes substantially less resources than the corresponding active component. In the cloud, with controlling standby instance placement to physical nodes, we can convert the sets if individual 1:1 redundancy associations to N+k association, and avoid the dedicated processing resources to be allocated for each active instance, which reduces the processing resource overhead (while full memory footprint to support replicas is still required). This redundancy resource gain was described in Ali Kafel's talk in the NFV World Congress presentation, which is available on line. ➢ **Application decompositioning** – this means that the application is partitioned to stateful and "stateless" (i.e. externalized state) components, and only stateful VNF component instances are protected. As an example, it is possible to implement a router where the control plane instance is protected with full VM state replication and dataplane instances are protected against failures, possibly in n-way active or n+1 redundancy configuration, but without state replication. For session-stateful applications (e.g. NAT, firewall, etc.), state protection can be confined to transactions that modify the session state records instead of to every packet of the session. This is analogous to most dataplane intensive implementations from the physical world, and can be easily implemented. There are additional benefits for the decompositioning, such as additional freedom to optimize VM (VNFCI) placement, as well as improved VM live migration performance (including improved outage times on migration), which is important for ensuring that the infrastructure maintenance operations as well as the infrastructure power management (through dynamic workload consolidation) operations do not become sources of service outages. **Recommendations for future work (state replication and resource use):** Describe methods and mechanisms to minimize the resource requirements of the state replication in the REL working group documents (preferably in REL003, or possibly in a future work item depending on the schedule). **Latency** The property of the "FT" mode that ensures the globally consistent state over instance failover (i.e. the surrounding entities are unaware of the failover had taken place) is guaranteed through buffering the I/O until the full VM checkpoint acknowledgement from the standby instance has been received. This buffering introduces extra latency that is proportional to the length of the VM state checkpoint interval. There are multiple ways to improve on this property: 1.) Faster VM state checkpoint intervals – we are in control of the interval bounds, so this is easy to do; however its effect on the replication effectiveness needs further study. We have so far performed initial measurements and we can support sub-milisecond VM state checkpoint times (and corresponding sub-millisecond maximum latency penalty), but we need to perform more work to fully characterize the system with such short intervals. 2.) Application decompositioning – as above; only use the VM state checkpointing for the control elements rather than forwarding elements; this eliminates the added latency on the datapath, and constraints it on the control element transactions only, where the latency is not generally as big issue due to more complex (and therefore more time

	consuming) processing associated with those transactions. This is again analogous to current implementation – e.g. one would not replicate the internal state of the forwarding ASIC in the HW implementation (and implicitly the whole ASIC), but might want to ensure that associated control state is protected. **Recommendations for future work (latency):** Latency behaviour of the VM state checkpoint mechanisms was studied in the REL002 work item in the context of set of implementations, which concluded that more work is needed to reach conclusion in this area. One important study aspect is to understand the latency targets for various application categories. We are planning to document the associated implementation tradeoffs and provide this as input to REL working group in future contributions. Unfortunately while we started on this aspect, we did not have time to provide full latency performance qualifications in the PoC timeframe. **Other proposed future work items:** It should be re-iterated that while "FT" mode demonstrates that **all** aspects of the fault management, including the full VM state replication can be performed as infrastructure service, it is not necessary for the infrastructure to do everything as a service – it is also possible to do any selected subset, e.g. full FM, with the application level state protection. Further discussion on the supported FM models and mechanisms as infrastructure services should be included in the REL working group efforts. REL003 Work Item is a good start on this path, but more work is expected to remain. There are certain FM processes that clearly need to be performed as infrastructure services (placement, certain types of fault detection such as HW fault indications that are masked from the applications due to lack of direct HW access, fault correlation and associated user VNF(M) notifications (including associated messaging infrastructure and its performance and reliability aspects), fault containment mechanisms and infrastructure network service recovery mechanisms to name a few. Some of the other processes can be utilized on demand by VNFs/VNFM's but they will necessarily need to rely on the services as exposed by associated infrastructure APIs, primarily through the VIM northbound APIs in the NFV MANO architecture. Other set of services that can be provided either by infrastructure or application include state protection services, and FM decision making services. Standardized fault indications from NFVI components and subsystems will be an essential requirement if it is assumed that the VNF or its VNFM is to be able to take full responsibility of the associated fault recovery handling. Such standards should be established either in NFV ISG effort, or in some associated standards development organization. It would also be beneficial to identify the key past domain specific and general standards on this area and review them for the gaps.

Objective Id:	**Scenario #2.2** – Service resiliency upon VNF failure using hypervisor HA. When the failure is detected we immediately respawn the VM. This is defined as Hypervisor High Availability ("HA")	
Description:	• In this scenario two VNFs are deployed in "HA" mode, without state replication (recovery is based on restart) • Resiliency is demonstrated in three different scenarios a. Kill the individual VM process (VNFs/VNFCs) b. Kill the OS and/or hypervisor (all instantiated VNFs/VNFCs) c. Kill the hardware server (all instantiated VNFs/VNFCs) • Availability metrics (accessibility and continuity metrics) are automatically monitored by test application through the injected fault scenarios and associated performance parameters are recorded. • For qualitative availability performance monitoring, the quality of the application performance is monitored and any observed performance impacts (including no noticeable impact) are recorded	
Pre-conditions	VNFs deployed in "HA" mode and passing traffic, after completion of scenario 1.3.	
Procedure:	**1**	For qualitative assessment, we used video server and client pair, which allowed observation of the effect on this demo application in the PoC demonstrations, along with the Aeroflex traffic monitoring application instances configured to send test traffic through this path to be able to monitor the path downtime performance in real time.
	2	For quantitative assessment of the downtime, we passed test packets through the path at the rate of 10k packets per second, and used the time from 1st lost packet to 1st subsequent packet received after the fault and associated recovery processes to measure the time the path was out of service for each injected fault.
	3	For qualitative tests in demo configuration instances / nodes deployed in "HA" configuration, each of the three scenarios outlined in the Description above were performed in some of the demos. In public PoC demo scenarios, we were mostly using the approach c.) – i.e. kill the node(s) through power cut-off, which was convenient way to allow observation of the effects on the video application as well as the path packet transfer outage performance simultaneously, allowing the PoC demo audiences to get an idea of relative performance of each of the three modes demonstrated.
	4	For quantitative tests in demo configuration instances / nodes deployed in "HA" configuration, each of the three scenarios outlined in the Description above were performed, and downtime performance results were recorded.

Results Details:	Objectives were successfully demonstrated. **Qualitative assessment:** For the video application demonstration, we used laptop with video playback application to provide convenient and hands-off way to demonstrate the outage impact in the context of the real-world application most observers are familiar with. As expected, the video stream playback stopped approximately for the duration of the outage. As expected, the TCP sessions did not persist over this outage, as the session state that was dynamically instantiated on the session stateful elements (primarily stateful firewall) was lost upon associated VNF component instance failure. This met our objective to demonstrate that while the *service accessibility* can be restored in this availability mode after the associated outage period, *service continuity* cannot be supported with these mechanisms alone for the stateful service elements. **Quantitative assessment:** Repair times (from fault occurrence through detection to the completion of the Fault Management processes, which in this scenario included re-instantiation of the failed VNF instance) varied from 120 to 350 seconds, with average at ~250 seconds. The large variation in this case was mostly attributable to the OpenStack related processes, where the key variations included variations in instance fault detection time (~60 to 120 second range), as well as very wide variations in the time to re-instantiate and start a new instance (10s to as much as 250s from the request).

Lessons Learnt & Recommendations	The key advantage of the “HA” mode is that it does not require pre-commitment of the resources for the protection, because in this mode the instances are re-instantiated upon detected failure. Therefore, the main benefits would translate primarily to cost saving due to no need to pre-commit resources (CapEx) and no need to power and cool the pre-committed resources (OpEx). The obvious disadvantages are the long recovery (repair) times, and the fact that this mode cannot support the direct state replication between the deployed entities (however, it would be possible to retain critical state through “externalized” state retention mechanisms where either persistent storage facilities or memory based checkpointing mechanisms are utilized for applications that do not have extremely high state modification rates). Given that the recovery times in this deployment model were identified to be almost entirely attributable to the OpenStack processes, the utility of this mode could benefit on any work to make the both fault detection and VM deployment processes faster and/or more deterministic (i.e. to lower the maximum time bounds of these processes). It should be noted that we did not make an attempt to optimize these processes as part of the PoC effort, and this is subject to the future efforts – we see three key paths to improve on this: a.) use performance optimized controller and storage configurations, b.) tune the associated baseline OpenStack services as currently implemented to faster detection as much as possible, and c.) changes on the relevant OpenStack processes themselves to optimize the timelines and to reduce presently large variations. Other potential avenue that might help to overcome the long recovery times of VMs could be to utilize containers (possibly deployed within VMs), which are commonly stated to support “very fast” start times by container proponents. To assess the feasibility and performance of this approach would require testing of container deployments in OpenStack context, and might be something that should be undertaken after EVE working group makes more progress on their container use cases effort in the context of NFV. As it stands currently, we conclude that this mode is not applicable to most of the telco applications that require “5NINES” levels of availability performance. Although the performance is marginally under the 315sec 5NINES requirement for 12 month period, whether this would be useable to provided 5NINES capable **services** is dependent on the number of instances in the service chain, their MTBFs, and other downtime sources. As tested, a single instance failure would take almost all of the yearly downtime budget, just for one fault event occurrence per year – our general assessment is that this would not be generally sufficient for supporting 5NINES services. However, there are potential application processes even in telco internal systems (e.g. batch processes for billing, analytics etc.) as well as enterprise workloads (both of which some operators are planning to support in same converged hybrid cloud infrastructure) where this level of availability performance can be sufficient. Generally, this mode may be appropriate for the workloads that do not support application state replication and therefore would be candidates for restart based recovery, provided that such applications do not require very high availability levels. In the context of high-availability telco applications, this mode may also be sufficient building block for cases where stateless (or externalized state protection) applications are deployed in parallel structures such as N-way active, and instance failures do not result on service outage, but rather 1/N level of throughput degradation during the recovery processes. To address such cases, faster failure detection mechanisms would be required to avoid perceptible outages associated with the failed instances, even if the recovery (i.e. re-instantiation) related mechanisms would be left to their current performance levels.

	To overcome the issues on the long recovery times, it is necessary to support availability modes where the stand-by instances are pre-instantiated. If the faster detection and recovery times will be implemented over time, then the usability of this mode will expand to cover more of the application scenarios.

Objective Id:	**Scenario #2.3** – Service resiliency upon VNF failure in GA mode. When the failure is detected, operator is notified but operator (or NFVI external agent) intervention is required to respin the VM to restore the service. This is defined as General Availability (GA)	
Description:	● In this scenario two VNFs are deployed in GA mode, without state replication (recovery is based on restart). ● Resiliency is demonstrated in three different scenarios a. Kill the individual VM process (VNFs/VNFCs) b. Kill the OS and/or hypervisor (all instantiated VNFs/VNFCs) c. Kill the hardware server (all instantiated VNFs/VNFCs) ● Notification of the failure to operator is demonstrated, followed by the use of MANO interfaces to re-instantiate the failed VNF to restore service. ● Availability metrics (accessibility and continuity metrics) are automatically monitored by test application through the injected fault scenarios and associated performance parameters are recorded. ● For qualitative availability performance monitoring, the quality of the application performance is monitored and any observed performance impacts (including no noticeable impact) are recorded.	
Pre-conditions	VNFs deployed in "GA" mode and passing traffic, after completion of scenario 1.3.	
Procedure:	**1**	For qualitative assessment, we used video server and client pair, which allowed observation of the effect on this demo application in the PoC demonstrations, along with the Aeroflex traffic monitoring application instances configured to send test traffic through this path to be able to monitor the path downtime performance in real time.
	2	For quantitative assessment of the downtime, we passed test packets through the path at the rate of 10k packets per second, and used the time from 1st lost packet to 1st subsequent packet received after the fault and associated recovery processes to measure the time the path was out of service for each injected fault.
	3	Faults, as outlined in description above were injected into system, and results were observed.
	4	GUI features were utilized to eventually manually restore the failed instances back to service.

Results Details:	Objectives were successfully demonstrated. As this scenario was mostly to demonstrate the baseline performance of the OpenStack based VIM environment “as-is”, i.e. without any additional mechanisms implemented to support instance availability, there was no point on attempting to characterize in detail the performance – i.e. as expected, when something fails, it stays failed until external repair action is take (whether manual or using automated fault management agent). An example scenario with automated FM agent deployed on top of the OpenStack to perform recovery actions is “HA” mode. **Qualitative assessment:** In demo’s, manual actions were taken to bring the GA instances back to service. This process varied in time depending on the factors such as operator attentiveness to fault indications as well as the OpenStack related new instance deployment performance comparable to and as quantified in the “HA” mode, described in scenario #2.2 above. In the context of PoC (or in any real deployment for such mode relying on the operator intervention for service restoration), the mean reaction and interaction times would need to be added to “HA” scenario numbers to get the resulting expected outage performance. As expected, the demo stopped when faults were injected, and the affected IP packet transfer path was unavailable until the failure was manually rectified. Further, as the session state was also not protected in this mode, service continuity could not be ensured without the end-user intervention. Therefore, this mode is not adequate to support neither high *service accessibility* nor any *service continuity* objectives.
Lessons Learnt & Recommendations	This demonstrates that OpenStack based VIM mechanisms alone are insufficient for supporting the “carrier-grade” availability objectives. Baseline functionality is only adequate for supporting development scenarios and non-resilient workloads. Further, the instance fault detection mechanisms that are currently native to OpenStack are themselves too slow to support the fast failure recovery (implicitly, you cannot recover in time less than the detection time and associated fault event message delivery processes), even if the recovery mechanisms would otherwise be deployed on top of the baseline OpenStack based VIM installation. We do not recommend that this mode is utilized for any mission critical applications, where failure induced outage performance is considered to be important.

B.2.2 PoC Contribution to NFV ISG

Use the table below to list any contributions to the NFV ISG resulting from this PoC Project.

Overall, as was the PoC objective, we have demonstrated the feasibility of implementation of complete fault management cycle and state protection mechanisms as NFVI+MANO services, which aligns with the proposition on REL002 and REL003 work items, and identified several aspects of the future specification and study items to progress the standardization of the fault management services in the NFV infrastructure.

Contribution	WG/EG	Work Item (WI)	Comments
Email and WG meeting discussions	REL	ETSI GS NFV-REL 002	Various contributions during meetings and through emails; this PoC demonstrates the availability performance aspects of a specific implementation of application transparent methods (in "FT" mode) which is referred to as "checkpointing with buffering" in REL002. We will continue work to characterize the latency/performance tradeoffs on this scheme (as discussed on the action items and REL002 proposed future work items), and report on these aspects on future contributions and/or PoCs.
NFVREL(15)000076	REL	ETSI NFV-REL 003	A section in REL003 WI discussing resiliency mechanisms as infrastructure services; we will update the initial contribution based on the PoC findings and other discussions. This work is ongoing as per writing of this report, and referenced contribution should be considered just as a "snapshot", subject to subsequent updates on relevant discussion topics rather than final conclusive report.
NFVREL(15)000093	REL	ETSI NFV REL-003	Proposed standardized / harmonized terminology for Fault Management cycle phases for NFV

B.2.3 Gaps identified in NFV standardization

See the list of suggested action items below. We will work to identify specific existing or new work items to address them in subsequent work. Unfortunately we did not have time to review the state of all of the related IFA work items in the timeframe of the PoC, but we do believe that there will be reliability and availability related work required that will necessarily manifest on requirements on various interfaces that are in the scope of the IFA working group. We will address these in future contributions in REL and/or IFA working groups, after the baseline IFA deliverables will become available, potentially as a R&A gap analysis for the proposed baseline set of specifications.

We have reviewed the OASIS TOSCA NFV draft document, and as a specific comment to that work item, we would like to suggest that more comprehensive treatment of the availability modes (including redundancy models, certain fault management aspects and potentially state replication methods) needs to be explicitly included in the specification to be able to address the resiliency requirements of the VNFCIs through infrastructure services (whether implemented in VNFM and/or lower layers of the NFV MANO stack). We will look for ways to provide more detailed feedback to OASIS on these matters, and determine whether this should be done through NFV ISG (which we would prefer) or directly to OASIS.

B.2.4 PoC Suggested Action Items

- While it is currently possible to model VNF internals in various ways, the issue is the lack of the agreed on **standard** for the vendor neutral VNFDs and VNF packaging in general. This requires VNF vendors and infrastructure SW vendors to work co-operatively to construct deployment package to each vendor specific environment (of which there are at least 10 different versions around), which is both labour and cost intensive process. We view this as one of the largest currently remaining obstacles towards the availability of interoperable, MANO vendor neutral VNFs. Overall, we had to perform relatively significant amount of engineering effort with participating VNF vendors to get to the deployable packages, while this should be eventually limited to addressing any issues encountered from onboarding the vendor supplied deployment packages.

- As in the scenario 1.1., standards and/or agreed on open source implementations are required for the APIs in the area of the inter-VNF network connectivity configuration.

- While the lack of the preferred interface standards and configuration data models is not necessarily different to the state of network element configurations in the present physical deployments, more standardization of associated management interfaces and APIs for the configuration modelling could be greatly beneficial to streamline the deployment and configuration processes in NFV environments. Less vendor specific Interfaces / APIs would be a great start for making progress here, but those alone are not sufficient to fully address the issues (configuration object level standards with standardized syntax and semantics would be additionally required to reduce the differences in similar configuration tasks).

- NFV REL working group should study the downtime allocations in the realistic NFV configurations, and provide guidelines for overall downtime as well as key subsystem allocations (including allocations for the applications and maintenance actions). This could be based on the modelling efforts of REL003, and associated performance metrics should be described / specified based on the newly established REL QAF work item. Ideally, in addition to defined metrics and use of prescriptive models we would establish target downtime allocations and/or reaction times for key subsystems that can be used as a guidance for subsystem developers (such as in OPNFV project).

- Present NFV work assumes that the VNFM will be primarily responsible for the fault management actions. This is good approach for the VNF related aspects of the FM cycle, but in real implementations there will be multiple layers of co-operating fault managers, with increasing scope of actions that can be taken by layer, but with also increasing time to take actions, so we will have a hierarchy of fault manager layers. This structure allows the actions to be taken on the lowest layer that can perform the recovery, which generally results in the best performance (i.e. fastest recovery). What is missing from the current NFV documentation is state change events not only faults (this is covered at the high level – i.e. such notifications are required) but also for the fault management actions – this is required mechanism for both avoiding possibly conflicting multi-layer actions at the various layers, as well as an escalation mechanism when lower layer cannot successfully perform the recovery action. These state change notifications should be added to the present FM messaging requirements. This may be potentially coupled with multi-layer policy management efforts, if the assumed implementation is expected to rely on FM policies implemented at multiple layers, which has been suggested by some.

- Stratus will provide a contribution to describe methods and mechanisms to minimize the resource requirements of the state replication in the REL working group documents (preferably to REL003, or possibly in a future work item depending on the schedule).

- Latency behaviour of the VM statecheckpoint mechanisms was studied in the REL002 work item in the context of set of implementations, which concluded that more work is needed to reach conclusion in this area. One important study aspect is to understand the latency targets for various application categories. We are planning to document the associated implementation tradeoffs and provide this as input to REL working group in future contributions. Unfortunately while we started on this aspect, we did not have time to provide full latency performance qualifications in the PoC timeframe.

- Further discussion on the supported FM models and mechanisms as infrastructure services should be included in the REL working group efforts. REL003 Work Item is a good start on this path, but more work is expected to remain.

- There are certain FM processes that clearly need to be performed as infrastructure services (placement, certain types of fault detection such as HW fault indications that are masked from the applications due to lack of direct HW access, fault correlation and associated user VNF(M) notifications (including associated messaging infrastructure and its performance and reliability aspects), fault containment mechanisms and infrastructure network service recovery mechanisms to name a few. Some of the other processes can be utilized on demand by VNFs/VNFM's but they will necessarily need to rely on the services as exposed by associated infrastructure APIs, primarily through the VIM northbound APIs in the NFV MANO architecture. Other set of services that can be provided either by infrastructure or application include state protection services, and FM decision making services.

- Standardized fault indications from NFVI components and subsystems will be an essential requirement if it is assumed that the VNF or its VNFM is to be able to take full responsibility of the associated fault recovery handling. Such standards should be established either in NFV ISG effort, or in some associated standards development organization. It would also be potentially beneficial to identify the key past domain specific and general standards on this area and review them for the gaps.

- We see three key paths to improve on the performance of the "HA" type availability mode: a.) use performance optimized controller and storage configurations, b.) tune the associated baseline OpenStack services as currently implemented to faster detection as much as possible, and c.) changes on the relevant OpenStack processes themselves to optimize the timelines and to reduce presently large variations. This could be potentially appropriate project to consider in OPNFV.

- Other potential avenue that might help to overcome the long recovery times of VMs in "High Availability" type availability mode could be to utilize containers (possibly deployed within VMs), which are commonly stated to support "very fast" start times by container proponents (tens of milliseconds are often quoted in this context). To assess the feasibility and performance of this approach would require testing of container deployments in OpenStack context, and might be something that should be undertaken after EVE working group makes more progress on their container use cases effort in the context of NFV.

B.2.5 Any Additional messages the PoC Team wishes to convey to the NFV ISG as a whole?

- The PoC provided a great opportunity for joint-collaboration between VNF and NFVI vendors to showcase next-generation NFV solutions that provide High Availability and Resiliency in the Software infrastructure.

- This PoC demonstrates that OpenStack based VIM mechanisms **alone** are insufficient for supporting the "carrier-grade" availability objectives. Baseline functionality by itself is only adequate for supporting development scenarios and non-resilient workloads.

- The instance fault detection and associated event delivery mechanisms that are currently native to OpenStack are themselves too slow to support the fast failure recovery (implicitly, you cannot recover in time less than the detection time), even if the recovery mechanisms would otherwise be deployed on top of the baseline OpenStack based VIM installation.

- We do not recommend that native OpenStack VIM only mode without additional fault management mechanisms is utilized for any mission critical applications, where failure induced outage performance is considered to be important.

- As per the objectives of this PoC, we have successfully demonstrated that the VM based state replication can be used to ensure both high *service continuity* and *service accessibility* as NFVI + MANO services, and without the application awareness.

- The implementation of "FT" mode also implicitly demonstrates that all phases of the fault management cycle (fault detection, fault localization, fault isolation, fault recovery and fault repair) **can** be provided as infrastructure services (using a combination of NFVI and MANO level mechanisms). In addition to the fault management processes, we also demonstrated that the full VM stateful replication and recovery, which are required to ensure service continuity can also be provided by infrastructure – all without any application (i.e. VNF) level support mechanisms.

- The service availability for an associated "FT" instance pair can be calculated to be over 7NINES, assuming the generally very conservative (i.e. high) failure frequency of once per year for all causes, and 500ms maximum outage time. Due to very rapid repair processes enabled by homogenous resource pools in cloud (in seconds instead of hours), the second failure exposure risk can also be effectively minimized, while at the same time relaxing the constraints of physical repair process times, allowing deferred/batched maintenance processes instead of critical 24x7 maintenance with tight (e.g. 4hour) total repair time targets. We are also confident that the recovery and repair times can be both improved over the values measured in this PoC.

B.2.6 Any Additional messages the PoC Team wishes to convey to Network Operators and Service Providers?

- We would like network operators and service providers to consider a software infrastructure approach for HA and Resiliency mechanisms because of the findings in B.2.5

- There are many phases of the fault management cycle that are still required to support the "carrier grade" availability performance that have traditionally been implemented in the application software.

It is not possible to support everything that is required inside the application alone, the fault management in the cloud will always be co-operative set of processes between the VNFs, MANO and NFVI services. Examples of the processes that must move to the infrastructure include fast HW fault detection (typically in device drivers, cannot be done due to lack of direct HW access), fault correlation (no access to all events and associated physical infrastructure context), fault containment (no privileges to power off nodes, switches, etc. elements and various other containment processes).

- Similarly, cloud introduces new requirements for the resiliency mechanisms such as anti-affinity constrained instance placements to minimize or even eliminate common mode single points of failures. In "box" designs, it was typical to model the system configuration, in cloud we need to model the intended system configuration and it is the collective job of the infrastructure to first instantiate "system" in the virtualized environment and then help retain it in the desired availability / resiliency state.
- The key benefits of software infrastructure approach for HA and Resiliency mechanisms are:
    - Enables the deployment any KVM/OpenStack application with transparent and instantaneous Fault Tolerance for service accessibility and service continuity, without requiring code to implement these features within the VNFs.
    - Reduces time, complexity and risks of implementing HA and Resiliency mechanisms in every VNF application
    - Has the flexibility to deploy multiple availability modes with complete fault management (fault detection, localization, isolation, repair, recovery, and optional state protection) for all applications
    - State protection requirements and relevant availability modes vary by functional element, and software-defined availability approach can be applied for example to treat control and management plane processes differently from forwarding plane processes that are more latency sensitive and require accelerated packet processing through the "fast" path. This allows, for example, control elements to de deployed with stateful fault-tolerance while the associated forwarding elements may be deployed in "stateless" (i.e. without state replication) availability modes, leveraging DPDK and SR-IOV for highest possible performance and lowest latency processing
    - Enables standardized monitoring of the reliability and availability related events and performance metrics through the use of the common APIs (vs. different interfaces for each VNF vendor)
    - Enables implementation of new maintenance models, such as pooled and batched maintenance operations with relaxed timeframes (e.g. going from 4hour MTTR target to e.g. weekly batched physical repairs, assuming that the cloud is dimensioned to support sufficient resources to enable this). An example of how data gathered from previous item is to support this dimensioning process.
    - Clouds in-scale enable multiple new mechanisms to reduce the resource consumption to enable high service availability and continuity that are not possible in physical "box" designs due to limited scale, such as de-clustered redundancy.
    - Clouds will also enable new mechanisms for energy and performance management that were not feasible in physical boxes due to scale constraints that are removed in cloud. Implementation of such mechanisms will require standardization of the states beyond the current ISG efforts as well as coordination of the associated (likely policy based) management processes, e.g. fault management, power state management, and load / performance management. It is much easier (or potentially even only feasible) to enable such coordination when these processes are implemented as infrastructure software services (i.e. in NFVI and MANO).