Building the liquid Internet

Collaborative project

FP7 ICT-2011.1.1 Future Networks

D 2.3 – Final Liquid Net Architecture

Addendum

Project co-ordinator: Marcelo Bagnulo, UC3M

Start date: 1st January 2013 **Duration**: 36 months

Project officer: Remy Bayou

This project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no 317756

1 INTRODUCTION

The Internet today is composed of a heterogeneous set of resources and distributed functions with complex interconnections spanning from traditional network services up to virtualized ones. It includes hardware infrastructure to provide classic network and security functions (enhanced forwarding, Quality of Service (QoS), firewalls, Network Address Translators (NAT), Deep Packet Inspectors (DPI), traffic scrubbing, load balancing, etc.), as well as virtualized infrastructures (servers and storage technologies), platforms (OS and middleware functions/primitives), and software layers (Software as a Service - SaaS) that are typically deployed within data centres.

Multiple existing technologies and solutions allow network operators and service providers to easily launch a variety of on-demand services for the dynamic pooling of processing, storage, or bandwidth resources. Initial investigative work is also being carried out for the joint orchestration of processing and storage, however this is mostly within a single administrative entity and generally within a single physical data centre.

Key for the evolution of these technologies and services is the possibility to treat the different resources as belonging to the same pool and consequently orchestrate their allocation and usage to better match the complex, multi-layer service chaining. Such a resource pool is the basis of the Liquid Network Architecture developed in Trilogy2. Within Trilogy2 the term resource pool describes a collection of independent resources, which together act as a single more capable and more tangible resource. Once a resource pool has been created, it becomes a source of liquidity, as the elements of the pool are interchangeable. This does not however imply that the elements of the pool are homogeneous - they will typically vary in capability and location, so moving demand between them affects performance and redistributes costs.

Based on the concept of resource pool, different types of liquidity can be identified depending on the specific service aspects and involved technological areas. In Trilogy 2 the following liquidity scopes are identified:

- Cross-provider liquidity, which points to the pooling techniques for orchestrating and controlling bandwidth, processing and storage resources across different providers / resource owners.
- Cross-layer liquidity, which refers to the possibility of more flexible access to those network liquidity functions that are operating in different layers of the stack e.g. MPTCP creates bandwidth liquidity at the transport layer and MPLS traffic engineering creates liquidity at the sub-IP layer.
- Cross-resource liquidity, which aims to understand the trade-offs between different types of resources. For example, would it be possible to trade some storage resources in exchange for lower latency and higher bandwidth?

In this document we describe the Trilogy 2 architecture, which serves as a framework for all the liquidity tools developed within the project.

The overall framework is depicted in Figure 1. The framework includes the different resources pooled (storage, processing and network resources) and describes how the resources are pooled and in some cases traded. Moreover, it also describes if the different liquidity tools are cross layer or cross provider.

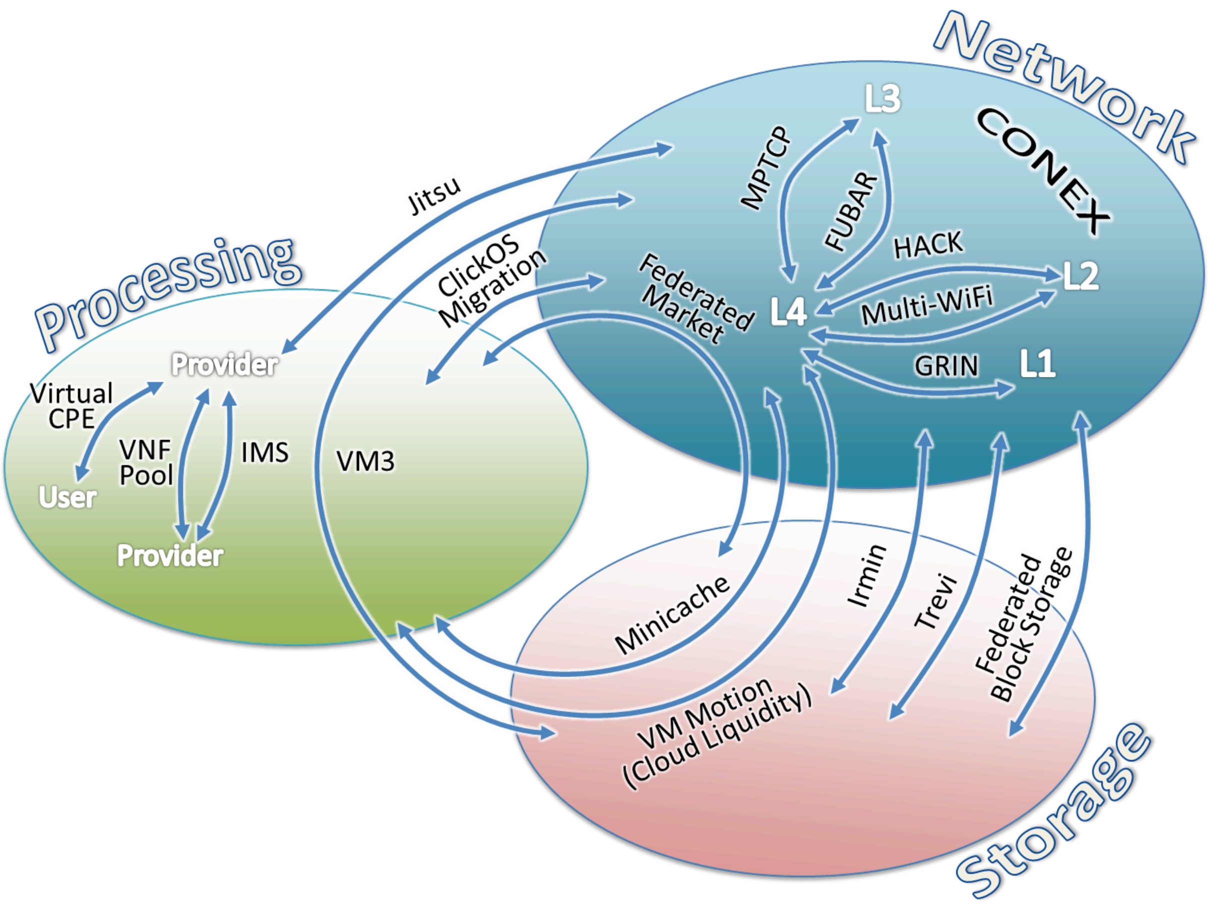

Figure 1. Trilogy 2 overall framework

2 RESOURCE POOLING MECHANISMS

We next include a description of the different mechanisms detailing how they provide resource pooling and what type of pooling they provide. The mechanisms described are the following ones:

- MPTCP-enabled ClickOS Migration
- Flow Utility Based Routing (FUBAR)
- GRIN
- TCP/HACK
- Minicache
- Multipath TCP
- MultiWiFi
- SwBRAS
- Resilient IMS
- Virtual CPE
- VNFPool
- Federated Block Storage
- Cloud Liquidity
- Federated Market
- vM3
- Jitsu
- Trevi

MPTCP-enabled ClickOS Migration	
Description	This work targets the ability to migrate virtualized middleboxes across networks and even WANs (i.e., potentially changing IP addresses) transparently, meaning that end-to-end connections do not notice the migration (TCP connections are not broken). To achieve this, we instrument ClickOS with the ability to create a new MPTCP subflow for the host which is the target of the migration. When the migration is finished, the subflow for the initial host is terminated.
Resources pooled	Processing and network
Cross liquidity considerations	The tool is cross-resource, allowing users to trade processing for network (e.g., by migrating to a more optimally-placed server so that bandwidth is saved)
Control tool	The ClickOS migration tool contains a control tool that receives migration requests and takes care of both migrating the ClickOS virtual machine and signalling the need for a new subflow at the target host.
Deliverable where it is described	D3.1 (Section 3.1) + D3.2 (Section 3.1)
Paper (or standard) where it is described	Towards the Super Fluid Cloud. Filipe Manco (NEC Europe Ltd.); Joao Martins (NEC Europe Ltd.); Felipe Huici (NEC Europe Ltd.). SIGCOMM 2014 Demo. See http://conferences.sigcomm.org/sigcomm/2014/program.php

Flow Utility Based Routing (FUBAR)	
Description	FUBAR is a system that reduces congestion and maximizes the utility of the entire network by installing new routes and changing the traffic load on existing ones. FUBAR works offline to periodically adjust the distribution of traffic on paths. It requires neither changes to end hosts nor precise prior knowledge of the traffic matrix.
Resources pooled	Network
Cross liquidity considerations	FUBAR is a routing system that increases overall network utility by exploiting path diversity. This results in both bandwidth gains for bandwidth hungry applications and reduced delay for delay sensitive traffic, increasing overall network liquidity. The tool is cross layer as congestion control involves the transport layer and routing involves the network layer and FUBAR harmonizes their behaviour

Control tool	FUBAR allows network operators to define the utility of end-user applications and express high-level traffic engineering traffic policies.
Deliverable where it is described	D2.4
Paper (or standard) where it is described	Nikola Gvozdiev, Brad Karp, and Mark Handley. *Fubar: Flow utility based routing.* In the 13th ACM Workshop on Hot Topics in Networks (HotNets '14), page 12. ACM, 2014

GRIN	
Description	GRIN is an overlay network designed to increase utilization for almost any existing topology, especially in a full-bisection datacenter setting. It relies on MPTCP and the assumption that servers often posses (or may support) multiple network ports (at least some of which are available). We use any free network port to interconnect servers directly. In this situation, a single server may transmit or receive data both using its uplink, and via any of its neighbors. In the best case scenario, the performance scales linearly with the number of extra ports. The performance improvements also prove to be significant for many real world use cases.
Resources pooled	Network
Cross liquidity considerations	GRIN is cross layer, as it enables liquidity by augmenting already existing datacenter topology (physical layer) and providing a transparent interface at the transport layer
Control tool	No
Deliverable where it is described	D1.2
Paper (or standard) where it is described	A. Agache, R. Deaconescu and C. Raiciu "Increasing Datacenter Network Utilisation with GRIN", NSDI 2015

TCP/HACK	
Description	TCP/HACK is a cross-layer mechanism that increases the throughput of TCP operating over wireless networks. TCP/HACK, at its core, encapsulates TCP ACKs within WiFi's link-layer acknowledgments, therefore eliminating all medium acquisitions for TCP ACKs in unidirectional TCP flows and increasing the achievable capacity for TCP workloads
Resources pooled	Network
Cross liquidity considerations	TCP/HACK is a cross-layer optimization. By allowing WiFi's link layer to retain some knowledge about TCP, TCP/HACK achieves significant throughput gains
Control tool	No
Deliverable where it is described	D1.3
Paper (or standard) where it is described	Lynne Salameh, Astrit Zhushi, Mark Handley, Kyle Jamieson Brad Karp. "HACK: Hierarchical ACKs for Efficient Wireless Medium Utilization", Usenix ATC, 2014

Minicache	
Description	Minicache is a specialized, virtualized content cache that allows for building on-the-fly CDNs (Content Distribution Networks) with high performance. Thanks to features like a minimalistic OS, a specialized filesystem, and optimized I/O, Minicache instances can be quickly instantiated (in 100 milliseconds or less), and can stream video at rates of several Gb/s. These features allow it to quickly build large distribution networks for live streaming or Video-on-Demand
Resources pooled	Processing, network and storage
Cross liquidity considerations	The tool is cross-resource, since it can trade-off processing (i.e., a content cache) for bandwidth (by placing a content cache close to its clients); or trade-off storage against bandwidth, by storing content locally at the edge of the network and thus keeping traffic away from the core. Further, it is cross-provider, since it allows virtual CDNs to be built on infrastructure owned by different providers

Control tool	Minicache includes a small control plane tool to be able to manage Minicache instances, but is not itself a control tool per se
Deliverable where it is described	D3.1
Paper (or standard) where it is described	Towards Minimalistic, Virtualized Content Caches with Minicache. Simon Kuenzer, Joao Martins, Mohamed Ahmed, and Felipe Huici. CoNEXT 2013, Hot Middlebox Workshop

Multipath TCP	
Description	Multipath TCP is a TCP extension that enables TCP connections to use multiple IP addresses on the communicating endpoints. Multipath TCP enables different use cases including mobility (where the set of IP addresses used for a connection changes as the hosts move), bonding (where the capacity of several interfaces/paths is aggregated), increased performance (e.g. by using the path with the lowest delay), … Multipath TCP has already been used for various applications ranging from datacenters to smartphones
Resources pooled	Network
Cross liquidity considerations	Multipath TCP makes the network more liquid and enables new use cases such as ClickOS migration, which can leverage the cross liquidity. MPTCP is cross layer as it enables the transport layer to exploit the network layer path diversity
Control tool	The Multipath TCP path manager, briefly described in D1.3, exposes the main hooks of the Multipath TCP implementation in the Linux kernel to enable applications to control how the different subflows are used
Deliverable where it is described	Implementation is described in D1.1 Path manager is described in D1.3 Security issues are discussed in D2.4 and in D3.2
Paper (or standard) where it is described	FORD, A., RAICIU, C., HANDLEY, M., AND BONAVENTURE, O. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6824, January 2013.

MultiWiFi	
Description	MultiWiFi is a combination of tools that enables simultaneous use of multiple access points by kernels that have MPTCP functionality. A client can use duplicate identities, one for each AP it needs to connect to. The APs can belong to the same or to different networks, but a duplicate identity seems a different client from the point of view of the network. One tool implements channel switching on the client so that it can attach to APs on different channels. A second one connects to APs on the same channel preferring the one with more efficient air use
Resources pooled	Network
Cross liquidity considerations	MultiWiFi makes the wireless infrastructure more liquid and enables seamless transport layer mobility by coordinating the link layer with the transport layer
Control tool	MultiWiFi modules that are briefly described in D1.3, use functionality in the 802.11 common part of the stack in the Linux kernel to regulate time spent on APs depending on their efficiency in the air
Deliverable where it is described	Mechanism and implementation are described in D1.3
Paper (or standard) where it is described	A. Croitoru, D. Niculescu, and C. Raiciu “Towards Wifi Mobility without Fast Handover”, NSDI 2015.

Software BRAS	
Description	This tool leverages the ClickOS framework to build high performance, virtualized, software-based Broadband Remove Access Servers (BRAS). The advantages to operators from having a software BRAS are numerous, including lower capital expenses, no vendor lock-in, the ability to easily upgrade functionality, and the scalability and ease of migration that come with virtualization technologies
Resources pooled	Processing and network
Cross liquidity considerations	The tool is cross-resource: among their broad functionality, a software BRAS handles functions such as filtering, rate limiting and even compression, which trade computing cycles (to execute those functions) versus reduced load on the network

Control tool	No
Deliverable where it is described	D1.1
Paper (or standard) where it is described	Enhancing the BRAS through Virtualization. Thomas Dietz, Roberto Bifulco, Filipe Manco, Joao Martins, Hans-Joerg Kolbe, Felipe Huici. Netsoftt 2015

Resilient IMS	
Description	Resilient IP Multimedia Subsystem (IMS) allows an operator to dynamically reallocate users' sessions among their (probably virtual) IMS functional elements, which can also be used to recover after failures of those elements
Resources pooled	Processing
Cross liquidity considerations	This tool provides the functionality to transfer the state of IMS functional elements after virtual machine migrations, so we can consider it a cross-layer tool
Control tool	Resilient IMS extends the IMS control plane to improve sessions' setup
Deliverable where it is described	D3.1
Paper (or standard) where it is described	Under review

Virtual Customer Premise Equipment (CPE)	
Description	The Virtual CPE allows the execution in virtual environments of those network functions traditionally integrated in hardware gears at customer premises. Within the project, a set of integrated and orchestrated components for the virtualization of CPE functions in the operator's datacentre is implemented to let business customers incorporate new virtual assets (e.g. Virtual Machines) in their MPLS Layer 3 VPNs
Resources pooled	Processing and network
Cross liquidity considerations	The Virtual CPE in the operator's datacentre is a cross provider tools, in the sense that pools resources from different administrative domains, namely resources from the client and resources from the

	provider
Control Tool	The Virtual CPE leverages on MPLS control plane functionalities to provide isolation and separation of tenants within the virtualized infrastructure. In particular, the Virtual CPE in the datacentre integrates an extended Quagga BGP routing suite with full support of Multi-Path protocol capabilities and MPLS Layer 3 VPNs
Deliverable where it is described	D1.3
Paper (or standard) where it is described	Giacomo Bernini, Gino Carrozzo, Pedro A. Aranda Gutierrez, Diego R. Lopez "Virtualizing the Network Edge: Virtual CPE for the datacenter and the PoP", EUCNC 2014 conference, June 23-26 2014, Bologna, Italy

VNF Pool	
Description	VNF Pool allows virtual network functions (VNFs) to be pooled, providing scale-in and scale-out and resilience to them
Resources pooled	Processing and storage
Cross liquidity considerations	The tool provides liquidity *within* a provider domain.
Control tool	The VNFPool activities will define the control protocol to implement resilience mechanisms for VNFS and to allow VNFs to scale-in and scale-out. Therefore, VNFPOOL is the foundation for a tool to control these functions
Deliverable where it is described	D2.3
Paper (or standard) where it is described	https://datatracker.ietf.org/wg/vnfpool

Federated Block Storage (FBS)	
Description	The Federated Block Storage (FBS) system supports the use-cases 'Disaster Recovery as a Service' and 'Cloud Liquidity' – D3.2 'Use case Development' Section 4.2. The underlying storage associated with Virtual Machines (VMs) can be replicated across the wide-area using FBS. The key differentiator of FBS is that it establishes permanent network connections between the source and target sites at the block storage level. Caching is used to reduce the apparent latency affect of working on remote paths but under extreme load the performance will be limited by the characteristics of the wide-area (high latency) path
Resources pooled	Storage is the primary resource that is traded, utilising space in multiple locations. To transfer the blocks, the system needs to utilise network bandwidth to transfer the blocks between locations. The decision logic for where the resources should be located uses a small amount of CPU (replication paths are decided a-priori).
Cross liquidity considerations	FBS can be considered cross-resource in that the two resources being utilised are storage and network bandwidth. To maintain an adequate quality of service/experience (QoS/QoE) the network bandwidth should be suitable to support the expected storage I-O load ,otherwise the QoS and QoE will be adversely affected. FBS can also be considered to be cross-provider. The model as described in the Cloud Liquidity and DRaaS use-cases' is that a second Cloud location is used as the target site for block replication. In developing the system we have had to improve the various control tools to allow for cross-provider interactions. FBS has been architected such that the storage resources at the different sites and the network connectivity do not have to belong to the same entity. FBS has been implemented for the DRaaS use-case and based on customer feedback it appears that connection performance and security could be improved. In this context, there is potential scope to combine FBS with TCPcrypt for security and MPTCP to handle the connection hand-over (see §2.2 of D2.4 for details of SMTCP)
Control tool	Although not a control in itself, FBS relies on control logic to allow for cross-provider replication. To assist the setup of DRaaS between end-points there is a DRaaS dashboard that links the source and destination together and handles the AAA. Each cloud location/owner can also decide to disable or enable FBS but there are some SLAs in place that govern when and how often these can be (en/)disabled.
Deliverable where it is	D1.3

described	
Paper (or standard) where it is described	www.onapp.com/federation

VM Motion (Cloud Liquidity) (CL)	
Description	N.B. to differentiate from the Intel use-case we will refer to Virtual Machine Motion as Cloud Liquidity (CL). Cloud Liquidity augments the system of Federated Block Storage described in D1.3 'Advanced Cross Liquidity Tools' – Section 2.2, with the ability to migrate processing. The basic use-case is described in D3.2 'Use Case Development' – Section 4.2 'Cloud Liquidity'. CL allows for resources to be migrated between Cloud locations and providers at the granularity of Virtual Machines. VMs are made up of storage, network and processing as well as their configuration and state. Most current best of breed solutions allow VM migration across local infrastructure. Utilising the wide-area Federated Block Storage, CL can replicate the content across Cloud locations and then at the required time transfer the running state between the source and the destination. The difference between FBS as used in DRaaS and CL is that DRaaS requires a permanent helper VM and a dedicated connection between end-points. In CL there may be no prior connection, so all state has to be transferred over at the time of motion. The other main differentiator is that the remote network resources should be configured to replicate the network set up on the original site.
Resources pooled	Storage, Network, Processing
Cross liquidity considerations	This tool trades Virtual Machine (VM) resources between providers and as such is both cross-resource and cross-provider
Control tool	Uses the OnApp Federated Market and is currently under development. It also utilises the work carried out for Federated Block Storage
Deliverable where it is described	D3.2
Paper (or standard) where it is described	www.onapp.com/federation http://onapp.com/platform/onappdraas/

Federated Market	
Description	The Federated Market (FM) is a brokerage service that connects buyers with sellers or traders of compute and storage resources. Currently the FM only operates with OnApp Cloud sites but for commercial as well as research reasons, this will be expanded to allow control of other types of Cloud sites and the associated resources. The FM runs as a service that receives, and coordinates requests and instructions from different Cloud resource providers. The Federated Market is an enabling technology that allows the control of resources to be shared between sites and providers. Commands can be indirectly executed on another Cloud by sending instructions to the FM that are then in turn run on the destination Cloud. These resources are traded for real money. As of the current version the resource granularity is that of a Virtual Machine (VM). It is envisaged that over the course of the final year of T2 the FM will integrate light-weight containers and resource abstraction mechanisms such as Unikernel (UCAM) and ClickOS/MiniOS (NEC). This will increase the granularity of the market, creating pools of resources that are much more linked to one of the three resource types. The Federated Market has to handle requests from different entities and be able to manage complex interactions over the wide area. The FM also has a role in incentivising and enforcing fair resource trading by exposing resource utilisation and making the system as transparent to both sides as possible. To expose resources for virtual operators, the FM also has a dashboard that allows the purchasing of resources without owning infrastructure (www.cloud.net).
Resources pooled	Storage, Network, Processing
Cross liquidity considerations	The FM can be considered to be cross-resource, cross-provider and potentially cross-layer. It acts as a control tool (discussed next) and allows for resources (once they are described according to the information model), to be provisioned on remote sites and made accessible. Although no cross-layer mechanisms have currently been developed it is envisaged that techniques/protocols such as MPTCP can be utilised to improve cross-provider liquidity. Enabling seamless transmission of Cloud resources between sites and providers meets many of the objectives of the Trilogy 2 architecture
Control tool	The FM is a pure control tool in that it only arbitrates between resource providers and users. No resources from the FM are shared with either the supplier or trader. Monitoring, database requests/lookups and messaging systems are all self-contained

	within the FM and it acts as a connector between sites/providers
Deliverable where it is described	D1.1 'Software Platforms', showed the first incarnation of the Federated Market, as the Testbed platform. More detail is contained in D1.3 'Advanced Cross Liquidity Tools' – Section 4.1 'Federated Market'. As a pure control tool that manages resources and users within the platform, the Federated Market also touches on Incentives and Enforcement as well as the architecture; D2.3 'Final Liquid Net Architecture' and D2.4 'Advanced tools for controlling liquidity'.
Paper (or standard) where it is described	www.onapp.com/federation + www.cloud.net

Virtual Machine Migration for Mobile Devices (vM3)	
Description	The Virtual Machine Migration for Mobile (vM3) tool is responsible for creating liquidity on the end devices. It achieves this by migrating applications encapsulated into Virtual Machines. The migration process is a smooth and rapid transition between devices. This is accomplished by liquidizing the network resource of the devices using the MPTCP protocol
Resources pooled	Storage, Network, Processing
Cross liquidity considerations	The tool is cross-resource, allowing users to trade processing and storage for network.
Control tool	There is a control component that handles the VM migration process on both the sending and the receiving device. This component, among other things, ensures the creation of a new subflow at destination and the closing of the old subflow corresponding to the source.
Deliverable where it is described	D1.3
Paper (or standard) where it is described	No references other than deliverable at time of writing. There are plans however to open-source VM3.

Jitsu	
Description	Network latency is a problem for all cloud services. It can be mitigated by moving computation out of remote datacenters by rapidly instantiating local services near the user. This requires an

	embedded cloud platform on which to deploy multiple applications securely and quickly. We present Jitsu, a new Xen toolstack that satisfies the demands of secure multi-tenant isolation on resource-constrained embedded ARM devices. It does this by using unikernels: lightweight, compact, single address space, memory-safe virtual machines (VMs) written in a high-level language. Using fast shared memory channels, Jitsu provides a directory service that launches unikernels in response to network traffic and masks boot latency. Our evaluation shows Jitsu to be a power-efficient and responsive platform for hosting cloud services in the edge network while preserving the strong isolation guarantees of a type-1 hypervisor.
Resources pooled	Network, processing
Cross liquidity considerations	Jitsu provides a dynamic scheduling interface for unikernel-based services, and ensures that (when provided with a suitable SLA policy engine) that only the minimal VM resources are used. It thus provides liquidity across CPU and networking resources, and could be extended to dynamically attach storage nodes in the future as well.
Control tool	Controls launching of unikernels, and reduces latency for launch time, and allows for low latency communications between VMs, plus provides memory safe properties.
Deliverable where it is described	Not yet described – To be described in Y3
Paper (or standard) where it is described	Anil Madhavapeddy, Thomas Leonard, Magnus Skjegstad, Thomas Gazagnaire, and David Sheets, Dave Scott, Richard Mortier, Amir Chaudhry, and Balraj Singh, Jon Ludlam, Jon Crowcroft and Ian Leslie, Jitsu: Just-In-Time Summoning of Unikernels, NSDI 2015.

Trevi	
Description	Trevi is a storage system that uses fountain coding to encode writes to multiple servers using multicast. This enables reads of data blobs from multiple servers in parallel. Trevi combines resilience to loss, to load balancing (resource pooling) and to slow (straggler) in a way that is largely oblivious to the topology of the network and the locations of the storage systems. This is far simpler than using systems that actively monitor load on links and stores and attempt to apply some explicit optimisation based approach. It is implicit in the codes we use. The trade off now moves to the design and implementation of rate-efficient and computationally affordable codes, and away from the need for any load balancer design at all. At the heart of Trevi is a simple receiver driven flow control mechanism that does away with the need for reliable transports such as TCP. This mechanism leverages the inherent rateless property of fountain codes and avoids many of the chronic problems that beset TCP within data centres (TCP incast, TCP outcast and timeouts to name three). Trevi is able to make use of multipath availability without the need for any end-host optimisations and without the need for complex flow-aware hashing of data across intermediate paths
Resources pooled	Trevi pools storage resources across the network. While it is not directly involved in processing the requirement for encoding of data at the end points does imply that it will need to draw on processing resources
Cross liquidity considerations	Explain if the tools is cross resource, cross layer or cross-provider. Trevi is a cross layer tool in that it combines encoding at the application layer with a simplified flow control at the transport layer and runs on top of multicast at the network layer.
Control tool	No
Deliverable where it is described	D1.3
Paper (or standard) where it is described	George Parisis, Toby Moncaster, Anil Madhavapeddy, and Jon Crowcroft. *Trevi: Watering down storage hotspots with cool fountain codes*. In Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks, HotNets-XII, pages 22:1–22:7, New York, NY, USA, 2013. ACM.

The following table provides a summary of all the different tools.

Name	Storage	Processing	Network	Cross-resource	Cross-Layer	Cross provider
ClickOS Migration		X	X	X		
FUBAR			X		X	
GRIN			X		X	
TCP/HACK			X		X	
Minicache	X	X	X	X		X
Multipath TCP			X		X	X
MultiWiFi			X	X	X	
SwBRAS		X	X	X		
Resilient IMS		X			X	
Virtual CPE		X	X			X
VNFPool	x	x				
Federated Block Storage	X		X	X		X
Cloud Liquidity	X	X	X	X		X
Federated Market	X	X	X	X	(X)	X
vM3	X	X	X	X		
Jitsu		X	X	X		(X)
Trevi	X	(x)	X		X	

3 CONCLUSION

The main objective of Trilogy2 is to unlock the value inherent in joining up the pools of liquidity in the Internet. This Deliverable addendum to D2.3 summarises the tools and techniques that have been developed over the course of Trilogy2 and indicates how they address the Liquid Network principles. As stated in the DoW we have selected a few challenges in cross-resource, cross-layer and cross-provider resource pooling to demonstrate the techniques for the three types of resources that Trilogy2 focuses on. We have suggested a few mechanisms for sharing and managing resource liquidity across providers and between resource types to help identify trends and suggestions for more general mechanisms and tools.

Several use-cases have been proposed, developed and evaluated in the course of Trilogy2 so far. The different use-cases have reached different levels of maturity with some being product-ready whilst others are more research related topics and therefore still being worked on. The variety and combination of use-cases have allowed different areas across the scope of Trilogy2 framework to be addressed with novel solutions for creating and managing liquidity between them. During the course of the final year some of the most promising use cases may be linked together to form a converged solution that show-cases the linkage of liquid resources in a more fully evolved manner.