A Scalable and Dynamic Application-level Secure Communication Framework for Inter-Cloud Services

Ali Sajjad[a,b,*], Muttukrishnan Rajarajan[a], Andrea Zisman[a], Theo Dimitrakos[b]

[a] *City University London, EC1V0HB London, UK*
[b] *British Telecom Ltd, Adastral Park, B62 Orion Building PP10, IP53RE Ipswich, UK*

Abstract

Most of the current cloud computing platforms offer Infrastructure as a Service (IaaS) model, which aims to provision basic virtualized computing resources as on-demand and dynamic services. Nevertheless, a single cloud does not have limitless resources to offer to its users, hence the notion of an Inter-Cloud environment where a cloud can use the infrastructure resources of other clouds. However, there is no common framework in existence that allows the service owners to seamlessly provision even some basic services across multiple cloud service providers, albeit not due to any inherent incompatibility or proprietary nature of the foundation technologies on which these cloud platforms is built. In this paper we present a novel solution which aims to cover a gap in a subsection of this problem domain. Our solution offers a security architecture that enables service owners to provision a dynamic and service-oriented secure virtual private network on top of multiple cloud IaaS providers. It does this by leveraging the scalability, robustness and flexibility of peer-to-peer overlay techniques to eliminate the manual configuration, key management and peer churn problems encountered in setting up the secure communication channels dynamically, between different components of a typical service that is deployed on multiple clouds. We present the implementation details of our solution as well as experimental results carried out on two commercial clouds.

Keywords: Cloud computing, secure communication, virtual private

*Corresponding author
Email address: ali.sajjad@bt.com (Ali Sajjad)

1. Introduction

Most of the currently available Cloud Computing solutions are mainly focused on providing functionalities and services at the infrastructure level, e.g., improved performance for virtualization of compute, storage and network resources, as well as necessary fundamental functionality such as virtual machine (VM) migrations and server consolidation etc. In the cases when higher-level and more abstract concerns need to be addressed, existing Infrastructure as a Service (IaaS) solutions tend to focus on functional aspects only. Furthermore, if a cloud's computational and storage infrastructure resources are overloaded due to increased workloads, its service towards it clients will degrade. The idea of an Inter-Cloud [1] has been gaining much traction to address such a situation, where a cloud can borrow the required infrastructure resources of other clouds. However, in order to progress from a basic cloud service infrastructure to a more adaptable cloud service ecosystem, there is a great need for tools and services that support and provide higher-level concerns and non-functional aspects in a comprehensive manner.

The OPTIMIS project [2] is an ongoing effort in this regard which strives to provide a holistic approach to cloud service provisioning by offering a single abstraction for multiple coexisting cloud architectures. Of the various high-level concerns being addressed by the OPTIMIS project, a major concern of high importance is the provisioning of a secure communication framework to the services utilizing the resources of different cloud IaaS providers. The usage pattern of these services is usually quite flexible, i.e., on one hand they might be directly accessed by end-users or on the other hand they might be orchestrated by other Service Providers (SP) for their customers.

There are three fundamental steps in the life cycle of a service in the cloud computing ecosystem; the construction of the service, the deployment of the service to one or more IaaS clouds and lastly the operational management of the service. In the resulting scenarios, the presence of the multiple IaaS providers in the cloud ecosystem is the key issue that needs to be addressed by any inter-cloud security solution. A major goal of service owners is to select IaaS providers in an efficient way in order to host the different components of their services on appropriate clouds. In this respect, third-party cloud brokers [3] can play a major role in simplifying the use, performance

and delivery of the cloud services. These brokers can also offer an intermediation layer spanning across multiple cloud providers to deliver a host of optimization and value-added services which take advantage of the myriad individual cloud services e.g., aggregation of different services or arbitration for a best-match service from multiple similar services. For the numerous interaction possibilities among these parties, whatever the usage scenarios maybe, the security of data and the communication between the consumers of the service and its multiple providers is of paramount importance.

In the light of the above discussion, it is clear that an inter-cloud security solution is highly desirable that would provide a framework enabling seamless and secure communication between the actors of a cloud ecosystem over multiple cloud platforms. Such a solution, however, has to overcome a number of challenges because of architectural limitations. This is because most of the current cloud service platforms, and the multi-tenants environments they offer, make it difficult to give the consumers of their services flexible and scalable control over the core security aspects of their services like encryption, communication isolation and key management. Secure communication is also challenged by lack of dynamic network configurability in most cloud providers, caused by the inherent limitations of the fixed network architectures offered by these providers.

In this work we address the secure, flexible and scalable communication concerns that in our view must be overcome in order to provide holistic provisioning of services to consumers from multiple cloud service providers. We present the architecture and design of an inter-cloud secure communication framework that offers the features of dynamic and scalable virtual network formation, efficient and scalable key management and minimal manual configuration all on top of secure and private communication between the components of the service across multiple cloud platforms. Our architecture provides a single virtual network to the service using resources from multiple cloud providers and offers the capability to efficiently and transparently run services on top of this network while catering for the dynamic growth and shrinkage of the components of the service.

The rest of the paper is organized as follows: In Section 2 we outline the key motivations for our approach. We elaborate on the detailed Inter-Cloud Virtual Private Network (ICVPN) architecture in Section 3. In Section 4 we present our experimental setup and the analysis of the performance results of our solution. In Section 5 we present the background and related works that address peer-to-peer overlays, virtual network connectivity and key manage-

ment issue related to this domain. We conclude in Section 6 with the future directions of our work.

2. Motivation

The design and architecture of our inter-cloud secure communication framework is inspired by a collection of techniques like Virtual Private Networks [4] (VPN) and Peer-to-Peer (P2P) Overlays [5]. Network virtualization techniques like VPNs and P2P Overlays have been shown to provide their users legacy communication functionalities of their native network environments, despite the topology, configuration and management architecture of the actual underlying physical network. This fits perfectly with our goal of providing a secure virtual private network as a service to the consumers operating on top of multiple cloud providers. All complications and complexities of managing a physical network can be handled by the overlay network, enabling the services deployed on multiple clouds to benefit from a customized communication network typically only available in physical local-area environments.

2.1. Peer-to-Peer Overlay

Traditionally, most of the private network solutions for similar problem spaces require the direct and continuous control of a centralized administration entity over every aspect of the overlay network, consisting of all the participants that constitute and facilitate the operation of the service being deployed and run on the multiple cloud providers. Such a central controller provides services to authenticate, secure and police the interactions amongst peers. These centralized solutions make it almost necessary to provide complex support and management functionalities to meet the users' demand of smooth and continuous operation. Furthermore, to robustly handle the loads generated by a large number of users, significant infrastructure resources and services like mirroring or redundant instances and load-balancers must be set aside, incurring additional costs for the cloud service user. Peer-to-peer overlays, on the other hand, are designed to offer improved scalability, flexibility and availability in a distributed fashion without extensive reliance on centralized servers or resources. For these reasons, such overlay networks have been used very successfully to provide specialized application layer services like voice over IP (VoIP) e.g., Skype [6] and file sharing e.g., Bittorrent [7]. Structured P2P overlay networks based on distributed hash tables (DHT)

support the scalable storage and retrieval of *key*, *value* pairs on the overlay network which is very helpful when we need to store and retrieve meta-data related to the virtual private network management. Existing P2P algorithms like Chord [8], Pastry [9] and Tapestry [10] have been widely used to provide scalable and fast information storage and retrieval services for a vast variety of applications. We have leveraged the Kademlia algorithm [11] to cater for our storage and retrieval requirements to build up a virtual private network. This DHT-based algorithm locates values using the peer ID and guarantees that on average, any data object can be located in *O (log N)* peer hops, *N* being the number of peers in the overlay.

Therefore, by provisioning a VPN among the nodes of a P2P overlay network, we can enable feature of using secure communication between the components of a service deployed on multiple clouds. Furthermore, we promote an approach where a distributed and scalable key management framework is utilized to provide the cryptographic primitives used to establish secure tunnels among the nodes of the P2P overlay networks. The synergy of these three technologies produces a scalable, secure and robust inter-cloud communication solution which is able to handle a large number of communicating peers with considerably less management complexity.

In this paper, we present the design and architecture of such an Inter-Cloud Virtual Private Network (ICVPN) solution, which provides secure communication facilities to users that want to deploy their cloud service's components over the infrastructure of multiple cloud IaaS providers. At its core, it provides the ability to automatically establish P2P overlay networks among the VMs constituting a single cloud service that runs on multiple IaaS providers. We handle the platform interoperability and federation issues inherent in the inter-cloud environment by using a VM contextualization approach to provision our virtual machines on multiple cloud platforms as a part of our ICVPN solution's operation. Using the same P2P and VM contextualization techniques, we also offer a distributed key management service which facilitates the distribution of cryptographic constructs like keys and certificates to the VMs constituting the user's cloud service. The configuration and maintenance of the VPN connections using the P2P overlay is autonomous and transparent to the cloud service itself, as a major goal of our work is to free the cloud service user from the complicated configurations typically required to set up the key management and virtual networking infrastructures in similar problem spaces.

2.2. Admission Control

In P2P networks, bootstrapping a new peer is a well-known issue, i.e., there is a need for the new peer to discover the required configurations and peers of the overlay to successfully join the network and access resources. There are some traditional solutions for this issue like server-based peer lists, host caches containing information of the last-known hosts, and random address probing to actively find peers. In our solution, we embed some bootstrapping information in each peer so that when it comes up it can join the overlay but first it has to undergo an overlay admission control process.

Symbol	Explanation
$\mathbf{n}$	A large prime number
$\mathbf{g}$	A primitive root modulo n (often called a generator)
$\mathbf{s}$	A random string used as the user's salt
$\mathbf{P}$	The user's password
$\mathbf{x}$	A private key derived from the password and salt
$\mathbf{v}$	The host's password verifier
$\mathbf{u}$	Random scrambling parameter, publicly revealed
$\mathbf{a}, \mathbf{b}$	Ephemeral private keys, generated randomly and not publicly revealed
$\mathbf{A}, \mathbf{B}$	Corresponding public keys
$\mathbf{H}()$	Cryptographically secure one-way hash function
$(\mathbf{m}, \mathbf{n})$	The two quantities (strings) m and n concatenated
$\mathbf{K}$	Session key

Table 1: Mathematical notation for SRP

The requirement and importance of secure admission control is obvious as scalable key management and secure communication schemes are effective only after the peers join the overlay in a secure admission process. This is also useful to thwart the well-known vulnerability of P2P networks to Sybil attacks [12]. Our admission control scheme is used only once when the peer bootstraps for the first time and all peers of a cloud service share a secret key which is used for authentication at admission control via TLS-SRP (Secure Remote Password) [13].

The use of SRP protocol is very suitable for our system model as it allows a peer to authenticate itself to a bootstrap peer without exchanging the shared secret key, it is resistant to dictionary attacks, and it does not

Peer		**Super Peer**
	$PeerID \rightarrow$	$(lookup\, s, v)$
$x = H(s, P)$	$\leftarrow s$	
$A = g^a \pmod n$	$A \rightarrow$	
	$\leftarrow B, u$	$B = v + g^b \pmod n$
$S = (B - g^x)^{(a+ux)} \pmod n$		$S = (A.v^u)^b \pmod n$
$K = H(S)$		$K = H(S)$
$M[1] = H(A, B, K)$	$M[1] \rightarrow$	$(verify\, M[1])$
$(verify\, M[2])$	$\leftarrow M[2]$	$M[2] = H(A, M[1], K)$

Table 2: The Secure Remote Password protocol

need a trusted third party and thus avoids the overhead of an equivalent PKI-based scheme. The explanation of the mathematical symbols used in the protocol is given in Table 1, whereas the summarised interactions of the protocol itself, that are undertaken between the peers and the super peer, are given in Table 2.

After a peer is authenticated and joins the overlay, it is issued with a session key that is kept in a secure cache and is valid for a set time period. Our scheme can utilize previous session keys to generate new session keys to take advantage of key-continuity and avoiding overloading the authentication system.

2.3. Secure Service based Resource Discovery

To join a peer-to-peer overlay, each peer needs to acquire a unique identification number (peer ID). In most structured P2P systems, this is done by the peer itself by choosing a random number from a large identity space. However, this approach is vulnerable to Sybil attacks [12], in which the attacker can subvert the functioning of a per-to-peer overlay by creating and using a large number of false identities. A common way of dealing with this issue it to use some trusted authority to allocate peer IDs to the participating peers and the peers validate each other by querying the central authority with a validation request. In our model, it can work by designating a stable peer as the Certificate Authorities (CA) for the overlays other peers. The CA can assign peer IDs to the peers and signs a certificate that binds the serviceID of the cloud service making use of our solution and peer ID within the public certificate of the peer for a limited time duration. The peer then

can use this signed certificate to authenticate itself with other peers in the overlay.

However, using this Trusted Third Party (TTP) model to validate peers and allocate them their identities can introduce substantial communicational and computational overhead, especially as the number of peers in the overlay increases. We propose a decentralized solution that overcomes the above mentioned scalability problems by utilizing a functional encryption based scheme [14]. In a generic functional encryption scheme, a decryption key describes a function of the encrypted data to the user. This function $F(\cdot, \cdot)$ is modelled as a Turing Machine and an authority possessing a master secret key (msk) can generate a key skk that can be used to compute the function $F(k, \cdot)$ on some encrypted data. Identity-Based Encryption [15], [16], [17], Searchable Encryption [18] and Attribute-Based Encryption [19] are some examples of a Functional Encryption scheme. To describe it more formally but briefly, A functional encryption scheme (FE) for a functionality F dened over (K, X) is a sequence of four algorithms (setup, keygen, encryption, decryption) satisfying the following correctness condition for all $k \in K$ and $x \in X$ is given in Table 3.

Sequence	Explanation
$setup(1) \rightarrow (pp, msk)$	Generate a public and master secret key pair
$keygen(mk, k) \rightarrow sk$	Generate secret key for k
$enc(pp, x) \rightarrow c$	Encrypt message x
$dec(sk, c) \rightarrow y$	Use sk to decrypt c

Table 3: Four-tuple Functional Encryption

For ICVPN, we employ a special case of Functional Encryption which falls under the category of systems known as the predicate encryption schemes with public index. For our scheme we make use of the system defined in [19] as Attribute-Based Encryption (ABE), where the decision that which users can decrypt a ciphertext is based on the attributes and policies associated with the plaintext message and the user. In this scheme an authority creates secret keys for the users of the system based on attributes or policies for each user and anyone can encrypt a plaintext message by incorporating the appropriate attributes or policies in the scheme. There are two versions of the ABE, Key Policy ABE and Ciphertext-Policy ABE.

In KP-ABE, attributes are assigned to a ciphertext when creating the ciphertext and policies are assigned to users/keys by an authority which created the keys. A key provides an access formula that operates over the set of attributes that must evaluate to true for decryption to yield the plaintext message. A key can decrypt only those ciphertexts whose attributes satisfy the policy.

In CP-ABE, the users of the system are assigned different attributes and each user is issued a key from an authority for its set of attributes. The ciphertext contains a policy which is a Boolean predicate over the attribute space) and if the users attribute set satisfies the policy, they can use their key to decrypt the ciphertext. Another attractive feature of this scheme is that it is collusion resistant, i.e., multiple users cannot pool their attributes together to decrypt a ciphertext. We describe the implementation of our version of this scheme in Section III.

2.4. Secure Diffie-Hellman for Session Key Generation

The peers of the ICVPN use a Diffie-Hellman exchange based protocol to agree on a secret key S and parameters for establishing the IPsec tunnels between the VMs for secure communication. It avoids the overhead and complexities of the Public Key Infrastructure (PKI) and of managing the certificates in the peers. This protocol provides confidentiality and protection against man-in-the-middle (MiTM) attacks, whereas authentication is handled by the scheme described in the previous section. This protocol comes into action immediately after the communicating peers have completed the discovery phase and want to proceed to the secure communication phase.

The initiating peer A generates its ephemeral key pair before entering the secure communication phase. The peer begins the exchange by sending a Hello message to the other peer. The Hello message contains the peer ID of the peer. Each peer has a unique 160-bit random peer ID (PID) that is generated once at installation time. The PID is used to look up credentials and configuration data from the overlay DHT for a particular peer. The responding peer B replies with a Hello message of its own, containing its PID. On its receipt of the response, peer A sends the DH generator g, the DH prime p and

$$DH_A = g^a \bmod p$$

to the peer B. A hash of the public DH parameters and the responder Bs Hello message is performed and sent in the same message to prevent MiTM

attacks.

$$hash(g \,||\, p \,||\, DH_A \,||\, B(Hello))$$

All subsequent messages also contain a hash image that is used to link the messages together. This allows rejection of false messages injected during an exchange by a MiTM attacker. On receipt of the above message, peer B checks the hash using the received DH parameters for A and its own Hello message. If it matches, it generates its own random DH secret value and computes its public DH parameter

$$DH_B = g^b \bmod p$$

and sends it to A with the hash. It then calculates the DH result as

$$DH_R = (DH_A)^b \bmod p$$

Now A can deduce the same DH result as

$$DH_R = (DH_B)^a \bmod p$$

For the calculation of the shared secret S, first a total hash (H_τ) of all the received and sent messages in the current exchange is calculated by both peers. The final shared secret is the hash of a concatenation of the DHR, the PID's of A and B, and the H_τ.

$$S = hash(DH_R \,||\, PID_A \,||\, PID_B \,||\, H_\tau)$$

The PIDs act as the context fields and H_τ as a nonce value, as recommended in [20].

3. Design and Architecture

In this section we present our Inter-Cloud VPN architecture (ICVPN). The architecture consists of two main components, namely the peer-to-peer overlay and the secure virtual private connections, as described below.

3.1. Peer-to-Peer Overlay

The diversity of the protocols, APIs and capabilities of multiple cloud-providers participating in an inter-cloud scenario is a problem area that our system tries to solve by using P2P techniques and cloud-specific VM contextualization [21]. We take advantage of the fact that all cloud providers

provide automated means of launching VMs on their platforms, therefore the use of a VM contextualization service lets us operate in an inter-cloud environment as underlying differences of each cloud platform can be handled and bridged by the contextualization service.

The core technique employed by the ICVPN is the use of two tiers of P2P overlays. A universal P2P overlay is used to provide a scalable and secure service infrastructure to initiate and bind multiple VPN overlays to different cloud services in an inter-cloud scenario. The universal overlay itself can be initiated either by the cloud service user, an inter-cloud broker or the cloud service providers themselves. Its main purpose is to help with the bootstrapping activity of VPN peers. It also provides other functionalities such as service advertisement, service discovery mechanisms, and service code provisioning, with minimal requirement for manual configuration and administration. This approach acts as an aggregation service for the eventual peered overlay resources (which in this case are virtual machines) span across the inter-cloud domain to help form an inter-cloud virtual private network. The peers of the universal overlay act as super peers for the nodes of the underlying overlays and let new nodes enrol, authenticate, bootstrap and join a particular VPN overlay based on the cloud service requiring a VPN service. In our architecture, we deploy the universal overlay in such a fashion that we have at least one of its peers in every cloud provider participating in a particular inter-cloud scenario.

As depicted in Fig. 1, the cloud service user or the cloud broker could itself be a peer in the universal overlay and a subset of the universal overlay peers can act as super-peers for the peer nodes of the VPN overlay for a particular cloud service. The universal overlay peers can join and leave the system dynamically and additional VMs from the cloud providers can be provisioned to act as the universal overlay peers as well. As both the universal and the VPN overlay nodes are basically VMs provisioned from different cloud providers, they can be demoted or promoted from these overlays respectively based on parameters like performance and availability.

To join the universal overlay, each peer needs to acquire a unique identification number (PID). This is generated by the peer itself on its first initialization on a VM as a unique 160-bit random number. It also needs some bootstrapping data to validate itself with a super peer for admission into the overlay. The bootstrapping data consists of the IP addresses of the super peers, the ID of the service that this particular VM belongs to and the service secret key. This data is embedded in a secure cache on the VM by a VM con-

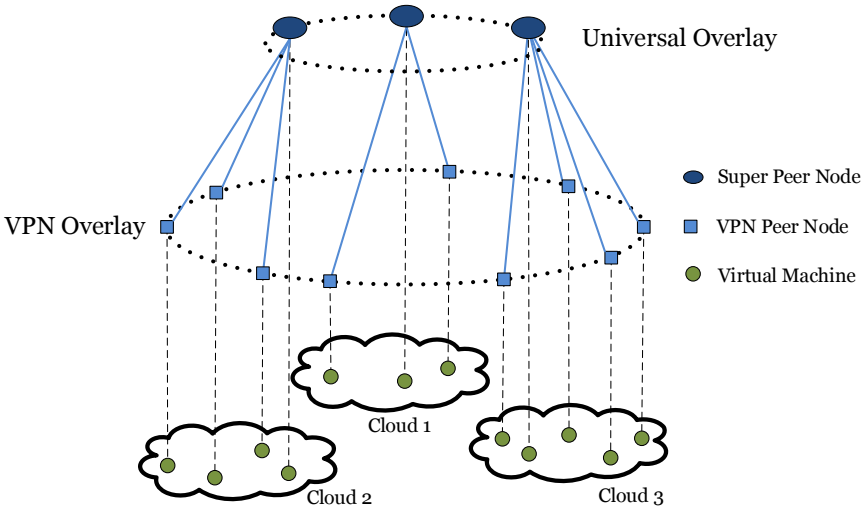

Figure 1: Two-tiered architecture for the Inter-Cloud VPN solution

textualization service [21] when it is provisioned for the service deployment and the same contextualization service is used to install the peer-to-peer client in the VM. So after bootstrapping phase, the peer follows the protocol described in Table 2 for a validated admission into the overlay, using the service secret key as the required password P.

Once the peer has joined its overlay, it needs to discover its neighbours and additional configuration data to establish secure tunnels with them so that the deployed service can communicate securely with its different components. In order to achieve this, we use the following scheme based on the Functional Encryption predicates discussed in the last Section. A simplified step-wise description of the scheme is as follows:-

i. A super peer sets up its own Master Secret ms and Public Parameters pp
ii. The super peer generates a private key for itself using the $ServiceID$ as the public key i.e. $Pub_{SP} = ServiceID \wedge SuperPeerID$ for each service the super peer is managing
iii. The VPN peer requests for pp on boot up from the super peer
iv. The VPN peer sends a Provisioning Request to super peer encrypted by the super peer's public key (Pub_{SP})
v. The super peer issues a private key to the VPN peer encrypted by its own private key, against the public key $Pub_{VPN} = VMID \wedge PeerID \wedge$

$ServiceID$

vi. The super peer inserts the VPN peers public key in the overlay DHT to keep a record of issued private keys, against the $key(ServiceID) = value\ (List\ of\ VMID)$ and for each peer; $key\ (VM_{ID_i}) = value\ (Pub_{VPN_i})$

vii. The VPN peer requests lists of other peers from super peer and it returns result of $key(ServiceID)$ encrypted using $Pub_{VPN_x} = PeerID\ \wedge\ ServiceID$

In a typical usage scenario, the cloud service user is responsible for provisioning virtual machines from cloud service providers to deploy and run their applications. These virtual machines are considered as the peers of the VPN overlays and the complete life-cycle of the peers is handled by a P2P client embedded in the appliance image used to instantiate a virtual machine on a cloud platform. However, a further advantage of the universal overlay approach is that the peers of a VPN overlay can get, update and modify the P2P client program dynamically from the super-peers in the universal overlay. The program to be run is signed by the super-peers for validity and it can check for updated versions of itself by querying for the associated *serviceID* in the persistent store of the universal overlays DHT.

3.2. Secure Virtual Private Connections

The key feature of our ICVPN is establishing a secure communication network between the peers of the overlay formed over a collection of cloud providers infrastructure. Therefore, after successfully joining the overlay network to become part of a service, a VPN peer starts the process of creating secure tunnels to the other peers of the service it wants to communicate with, according to the functional operations of that particular service. To achieve this, we make use of IPsec [22] to authenticate and encrypt each IP packet of a communication session between the peers, thus creating end-to-end tunnels which provide protection against eavesdropping, message tempering and message forgeries. For establishing mutual authentication between peers at the beginning of the session and negotiation of cryptographic keys to be used during the session, we employ the Internet Key Exchange protocol [23], which can make use of standard cryptographic primitives like public key cryptography [24] and AES [25]. In our approach, we make use of the secure Diffie-Hellman based scheme described in Section II to derive a secure session key which is used in the AES-CBC mode to ensure the confidentiality of the traffic exchanges between the peers using the tunnel [26]. Our

approach removes the Diffie-Hellmans well-known susceptibility to a MiTM attack. This is done by providing a way to authenticate the Diffie-Hellman exchange. In most traditional systems, this is done by depending on digital signatures backed by a centrally managed PKI. However, it has been shown from a practical point of view that deploying and managing a central PKI can be a complex and problematic experience as evident from the DigiNotar and Comodo incidents [27]. PKIs require too many managerial as well as computational and communicational resources, which are not easy to commit by a small scale cloud service customer. Especially in our target use case, where such customers wants to use the resources of multiple cloud providers, they typically does not want to deal with issues like cross-carrier authentication, certificate revocation lists, and other complexities. It is therefore a much simpler approach to avoid PKIs altogether, especially when developing secure commercial products. Hence, we augment the Diffie-Hellman exchange with the SRP scheme combined with secure hash usage at the start of the key exchange and PKI is not required for this approach to authenticate the Diffie-Hellman exchange. The session keys generated for the IPsec communication are valid for a short period of time and when the keys expire the protocol is run again to come up with new session keys to maintain the IPsec tunnels.

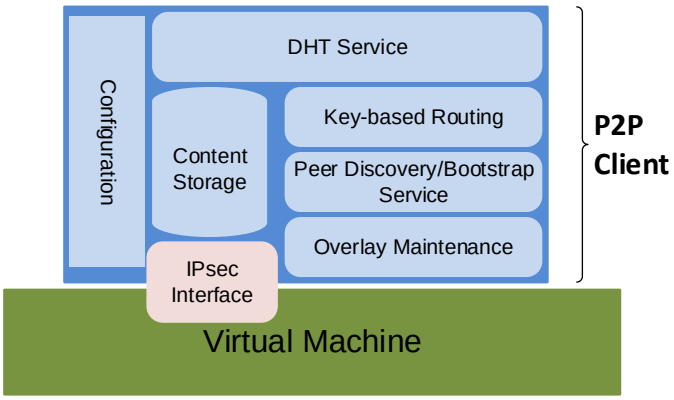

Figure 2: Architecture of a P2P client in the VPN overlay

Another practical advantage of this approach is the reuse of existing frameworks and tools which have been thoroughly tried and tested in a myriad of different domains, are widely used and have been adopted in both academic and commercial domain. The main components of the peer-to-peer

client used to construct a virtual private network in our model are shown in Fig. 2. These include the standard components required to form a structured peer-to-peer overlay like the Distributed Hash Table (DHT) service, which basically acts as the command-and-control channel for the ICVPN solution, key-based routing, peer discovery, bootstrapping service and overlay maintenance service. All of these services are provided by a modified Kademlia implementation mentioned in Section IV. In addition to these peer-to-peer specific components, we have a secure content storage for the client where sensitive data like keys, passwords, and security tokens etc. are stored. The configuration component is integrated with the overlays DHT so that the clients behaviour can be modified dynamically by push new configurations to it from the super peers. The configuration component manages both the peer-to-peer related configurations as well as the policies used to configure the IPsec tunnels between the peers for the use of the higher-level services using the client to provide the secure communication framework.

The P2P client software sets up and configures the IPsec security associations according the service network security policy, which is advertised by the cloud service user through the DHT of the Universal Overlay. The peers of the underlying VPN overlay periodically check for any update in the security policy and apply and enforce any changes on the kernel of the VM through the P2P client's IPsec interface.

4. Implementation and Evaluation

We implemented a working prototype of ICVPN using the Java programming language, that can be deployed on Linux based virtual machines. Our implementation is built using open source libraries and APIs. Specifically, we use the BouncyCastle library [28] for most of the cryptographic operations, the cpabe library [29] for the CP-ABE based access control, and the TomP2P library [30] for its implementation of the Kademlia [11] peer-to-peer protocol and the overlay DHT. In addition, we use BT Compute Cloud [31] and Flexiant Cloud [32] as our cloud service providers.

We present the results of a series of experiments we conducted to evaluate the effect of our prototype ICVPN solution upon the network performance of a service deployed on two different cloud IaaS providers. We use a 3-tier web service comprising of database, business logic and presentation components deployed on nine virtual machines hosted on the clouds of British Telecom Ltd. and Flexiant Ltd., our partners in the EU OPTIMIS project. The

purpose of these experiments is to evaluate the architecture being proposed, in terms of service latency and service throughput, in a practical scenario with a service deployed over a real wide-area network, with the BT cloud geographically located in Ipswich, England and Flexiant cloud located in Livingston, Scotland. We define service latency as the inter-cloud round-trip time taken by a HTTP request, issued by a service component on one cloud, to get a response from the target service component on a different cloud. Similarly, service throughput is the inter-cloud network throughput between service components deployed on different clouds.

4.1. Service Latency

We compare the latency between the components of the service deployed on different cloud providers, as the latency between the components in the same cloud is almost negligible as they are usually hosted on the same hypervisor. We measured the latency by using the round-trip delay of an HTTP HEAD request/response pair, as the components of the web service communicate with each other using HTTP protocol and ICMP, the de facto latency measurement protocol, is blocked in the networks of our cloud providers. We computed the average latency by running 10 experiments very hour for a period of 24 hours, firstly without using the ICVPN solution and then with it.

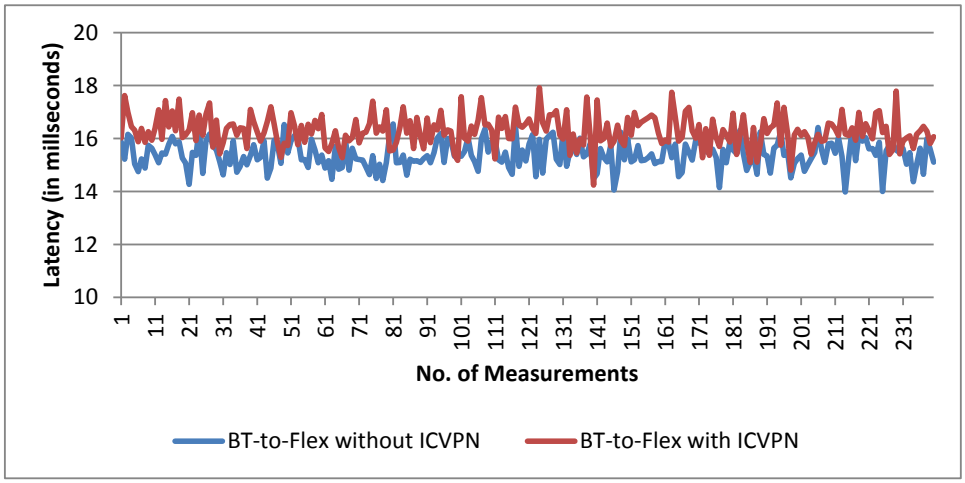

Figure 3: Service latency of 240 round-trip time experiments from BT to Flexiant clouds

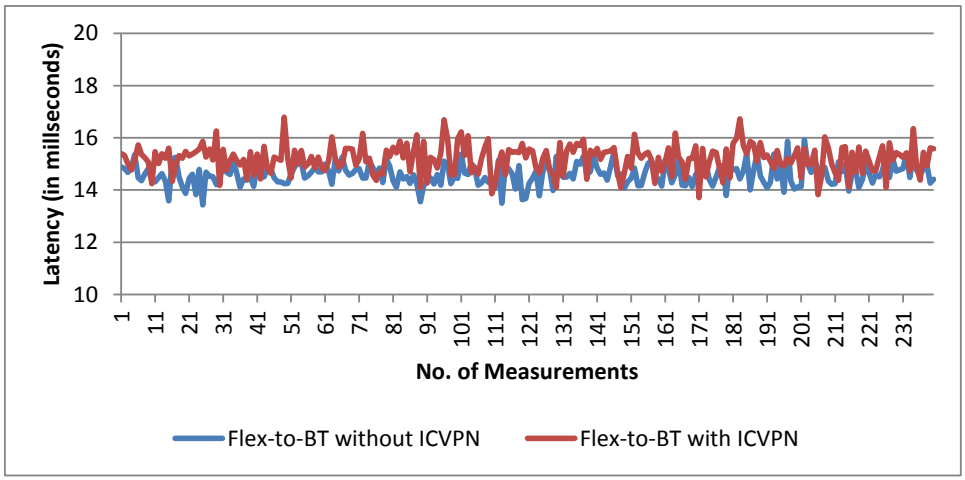

Figure 4: Service latency of 240 round-trip time experiments from Flexiant to BT clouds

Looking at the results shown in Fig. 3 and Fig. 4,, we can see that using our solution only has a small impact on the HTTP latency, increasing it just by about 5%. For ease of analysis, we collect the network traffic dump when running our experiments, using the *tcpdump* packet sniffer. We found out from the traffic dumps that the increased delay we encountered is mostly due to the additional packets transmitted and received by the peers for the purposes of key exchange and cryptographic primitives negotiation when establishing an IPsec tunnel. After this initial handshake phase is over, the latency performance is almost same in the comparative experiments.

4.2. Service Throughput

We measure the throughput between components of the service deployed on different cloud providers by using Iperf [33], a commonly used network testing tool. We measured the throughput in both directions by transferring 30 MB data, a size chosen empirically to saturate the WAN links between the components and get the throughput results representing realistic conditions. We computed the average throughput by running 10 experiments every hour for a period of 24 hours, firstly without using the ICVPN solution and then applying the security policy to tunnel the traffic through IPsec. The results are shown in Fig. 5.

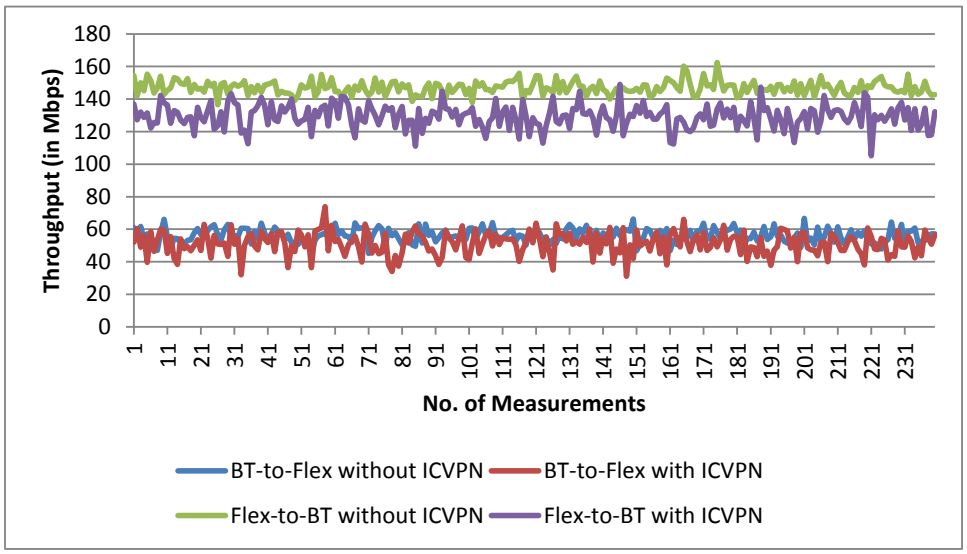

Figure 5: Service throughput of 240 data transmission experiments in both directions between BT and Flexiant clouds

From the throughput results, the first thing that stands out is the difference in the throughput values depending on the direction of transferring the data. Although we don't have the detailed knowledge of the underlying physical wide-area network connectivity between the two cloud service providers, such readings are not unheard of in this domain and are usually due to differences in upstream and downstream traffic conditions, different routes chosen by the IP packets or network configuration issues. Irrespective of that, by looking at the comparative results it is clear that we just incur a small overhead in the throughput, of about 10%. By analysing the traffic dumps generated from the throughout test, we can attribute this overhead to the IKE and IPsec handshakes in addition to the extra time taken by the VM kernel in encrypting and encapsulating 30 MB of data for each throughput test.

4.3. Service Scalability

One of the main motivation of using P2P overlays in our solution is their ability to scale as the number of VMs in the inter-cloud VPN service increases with the possible increase in the workload. Some cloud services can easily expand to hundreds even thousands of VMs across multiple clouds and it is important that our solution is able to cope with this sort of flexibility.

Therefore, in order to measure the scalability of our solution, we observe the scale-up behaviour of the super peers of our universal overlay as more and more P2P clients request to enrol and join their respective VPN overlays. The metric that we use to measure the scalability is the number of bootstrapping requests that a super peer can service per second as more and more VPN peers try to join an overlay. For this measurement, due to the limitation of resources and privileges in our test-bed cloud providers, instead of launching thousands of VMs to emulate a large number of peers trying to join an overlay, we launch only a few VMs containing the P2P client in each cloud provider but create more and more instances of the peer in the same VM to simulate a heavy workload. On the other hand, we increase the number of super peers handling the bootstrapping linearly and observe how many requests they were able to process per second by looking into their log files.

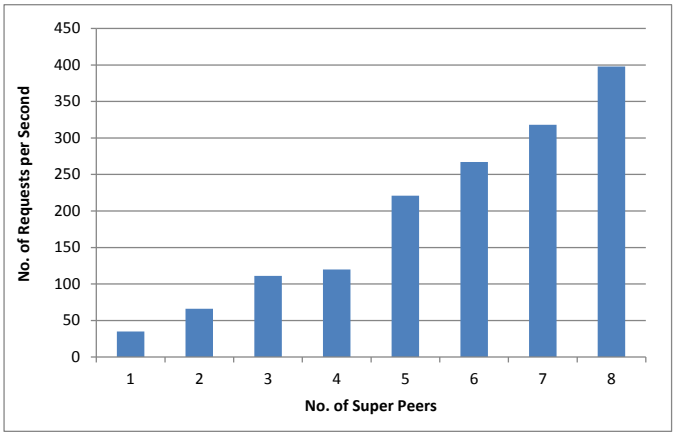

Figure 6: Effect of increasing the number of Super Peers on the maximum requests processed per second

As we can see in Fig. 6, our solution was able to handle more bootstrapping requests per second as we increased the number of super peers. Again, due to the limitation of resources and privileges, we limit the number of VMs acting as the dedicated super peers to four on each of our two test-bed cloud providers. We can see from the progression in Fig. 6, our solution was able to scale linearly to hundreds of requests per second as the number of super peers was increased.

4.4. Secure Resource Discovery Overheads

One of the main overheads in peer-to-peer overlays related to the cost of the resource discovery after the peers have bootstrapped. Securing this process further adds to this overhead but in an effort to characterise the effect of our secure resource discovery mechanism, we compare it with an alternate design of a PKI-based system where the super peers have the functionality of a Certificate Authority (CA), each peer is issued a signed certificate upon authenticated completion of the bootstrapping process and queries the Universal Overlay DHT for resource discovery and gets the resulting data back which is encrypted by the owning peer using its private key. We remove the cost of the DHT lookups from our measurements as their theoretical complexity is known to be $O\ log(n)$ for Kademlia DHT but due to the nature of actual runtime measurements they can add unhelpful noise to the data. We define the runtime cost for both designs as the time duration between the start and end of the secure resource discovery process.

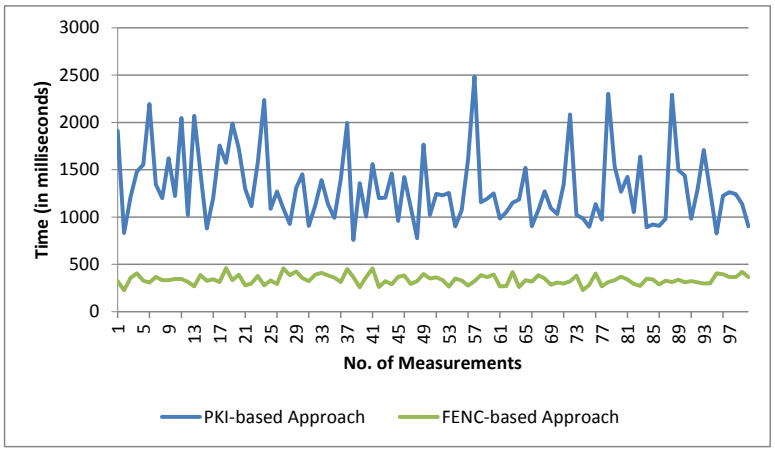

Figure 7: Secure resource discovery for 100 runtime analysis measurements between PKI and Functional Encryption approaches on ICVPN

From the results shown in Fig. 7, the mean runtime of the PKI-based design is 1313.52 milli-seconds whereas that for our Functional Encryption based scheme is 338.81 milli-seconds. This shows that our scheme incurs about 74.2% less overhead than a PKI based scheme.

5. Related Work

The central thrust of our architecture is the provisioning of a secure virtual private network over multi-cloud infrastructure. VPNs have been a mainstay for providing secure remote access over wide-area networks to resources in private organizational networks for a long time. Well-known tools and softwares like OpenVPN [34] are used to create secure point-to-point or site-to-site connections for authenticated remote access. However, the main problem in client/server based approaches is that they require centralized servers to manage the life cycle of all the secure connections for the participating clients, hence suffering from a single point-of-failure. Another issue is the quite complex and error prone configuration problems especially if you want to construct and manage a large-scale network not having a relatively simple topology, as it would require customized configuration on every client and even more elaborate management and routing configuration on the server-side. Another major drawback is the complexity of key distribution among all the participating clients in a VPN, as the software itself does not provide any key distribution service and all keys have to be manually transferred to individual hosts. In case of PKI model, an additional requirement of a trusted Certificate Authority exists that has to issue individual certificates to all the servers and clients constituting a VPN, which incurs an additional communication overhead when forming a virtual private network.

There have been some other VPN solutions for large-scale networks aimed at grid and cluster computing environments, such as VIOLIN [35] and VNET [36], that do not follow a strict client/server model based approach. VNET is a layer 2 virtual networking tool that relies on a VNET server running on a Virtual Machine Monitor (VMM) hosting a virtual machine in a remote network which establishes an encrypted tunnel connection to a VNET server running on a machine (called Proxy) inside the users home network. All of the remote virtual machines communication goes through this tunnel and the goal of the Proxy is to emulate the remote virtual machine as a local host on the users home network, in effect presenting it as a member of the same LAN. The motivation of this approach is to tackle the users lack of administrative control at remote grid sites to manipulate network resources like routing and resource reservations etc. but it suffers from the previously discussing problem of complex and manual configuration though going for the simplicity of a private LAN. Also the scalability will be a big issue for the Proxy as the number of remote virtual machines grows as each will require a secure

tunnel connection and corresponding virtual network interface mapped to the Proxys network interface by the VNET server software.

VIOLIN is a small-scale virtual network with virtual routers, switches and end hosts implemented in software and hosted by User-Mode Linux (UML) enabled machines as virtual appliances. It allows for the dynamic establishment of a private layer 3 virtual network among virtual machines, however, it does not offer dynamic or automatic network deployment or route management to setup the virtual network. Virtual links are established between the virtual appliances using encrypted UDP tunnels that have to be manually setup and are not self-configuring, making it cumbersome to establish inter-host connections in flexible and dynamic fashion.

P2P VPN solutions like Hamachi [37] and N2N [38] have come up as peer-to-peer alternatives to centralized and client/server model based VPNs. Hamachi is a shareware application that is capable of establishing direct links between computers that are behind NAT firewalls. A back-end cluster of servers are used to enable NAT traversal and establish direct peer-to-peer connections among its clients. Each client establishes and maintains a control connection to the server cluster. It is mainly used for internet gaming and remote administration but suffers from scalability issues as each peer has to maintain the connection with the server as well as any other peers it wants to communicate with, ending up with the overhead of a mesh-topology. It therefore offers limited number of peers (16 per virtual network) and limited number of concurrent clients (50 per virtual network). The keys used for connection encryption and authentication are also controlled by the vendors servers and individual users do not initially control who has access to their network. N2N is a layer 2 VPN solution which does not require a centralized back-end cluster of servers like Hamachi but it uses a peer-to-peer overlay network similar to Skype, where a number of dedicated super-nodes are used as relay agents for edge nodes that cannot communicate directly with each other due to firewall or NAT restrictions. The edge nodes connect to a super-node at start-up and pre-shared TwoFish [39] keys are used for link encryption. As it operates on layer 2, the users of the overlay have to configure their IP addresses etc. It also assumes node membership as relatively static with edge nodes rarely leaving or joining the network over their life cycle.

More recently, some commercial cloud computing services have been made available by different vendors that provide a virtual private network inside their public cloud offering and offering the customers some limited degree of

control over this network, which is called a Virtual Private Cloud (VPC). Prime examples in this domain are Amazon Virtual Private Cloud [40], Google Secure Data Connector [41] and CohsiveFT VPN-Cubed [42]. These are aimed at enterprise customers to allow them to access their resource deployed on the vendors cloud over an IPsec [22] based virtual private network. Although these products allow the possibility of leveraging the cloud providers APIs to flexibly grow and shrink their networks, the management and configuration is as complex as a traditional network as components of the VPC such as internet gateways, VPN servers, NAT instances and subnets have to be managed by the customers themselves. Furthermore, the customers are required to setup a hardware IPsec device on their premises that connects to an IPsec gateway in the VPC running as a virtual appliance which integrates the enterprises network with the VPC subnet in the cloud. Most importantly, with the exception of [42], these solutions are locked to single cloud vendor and [42] provides use of a selective set of cloud providers by placing its virtual appliances as VPN gateways in these cloud infrastructures and allowing the customers to join these gateways in a mesh topology manually.

6. Conclusion

In this paper, we present a scalable and robust secure communication framework for services deployed in an inter-cloud environment. We employ the flexibility and scalability afforded by structure peer-to-peer overlays to join virtual machines running on different cloud IaaS providers with each other using IPsec tunnels, hence providing confidentiality, authentication and integrity for all the data exchanged between different components of the service. Our solution needs minimal manual configuration as peers are automated to discover the information needed to perform their operations from the Universal Overlay. We also provide a distributed and scalable key management solution for the consumption of the virtual machines to setup the secure communication channels. Our solution supports the dynamic addition and removal of nodes from the VPN overlay as we use the peer-to-peer DHT not just as a command and control channel for managing the VPN peers but also for the churn management of peers in the VPN overlay. We have evaluated a prototype implementation based on experiments conducted in realistic conditions, over multiple cloud infrastructure environments and found minimal latency and throughput overhead of creating and maintaining

the ICVPN connections among the participating VMs of a service.

7. Acknowledgement

We acknowledge financial support for this work provided by the European Commission's Seventh Framework Programme under grant agreement number 257115, OPTIMIS.

References

[1] R. Buyya, R. Ranjan, R. N. Calheiros, Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services, in: Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2010, 2010.

[2] A. J. Ferrer, F. Hernndez, J. Tordsson, E. Elmroth, C. Zsigri, R. Sirvent, J. Guitart, R. M. Badia, K. Djemame, W. Ziegler, OPTIMIS: a holistic approach to cloud service provisioning, in: First International Conference on Utility and Cloud Computing, 2010.

[3] Gartner, Cloud consumers need brokerages to unlock the potential of cloud services (Jul. 2009).
URL `http://www.gartner.com/it/page.jsp?id=1064712`

[4] A. S. Tanenbaum, D. J. Wetherall, Virtual private networks, in: Computer Networks, 5th Edition, Prentice Hall, 2010, p. 821.

[5] D. Andersen, H. Balakrishnan, F. Kaashoek, R. Morris, Resilient overlay networks, SIGCOMM Comput. Commun. Rev.

[6] S. Baset, H. Schulzrinne, An analysis of the skype peer-to-peer internet telephony protocol, CoRR.

[7] B. Cohen, The BitTorrent protocol specification (2001).
URL `http://www.bittorrent.org/beps/bep_0003.html`

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, in: ACM SIGCOMM, 2001.

[9] A. Rowstron, P. Druschel, Pastry: Scalable, decentralized object location, and routing for Large-Scale Peer-to-Peer systems, in: Middleware 2001, 2001.

[10] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, J. D. Kubiatowicz, Tapestry: a resilient global-scale overlay for service deployment, Selected Areas in Communications, IEEE Journal on.

[11] P. "Maymounkov, D. Mazières, "kademlia: A peer-to-peer information system based on the xor metric", in: "Revised Papers from the First International Workshop on Peer-to-Peer Systems", "Springer-Verlag", "2002".

[12] J. Douceur, The sybil attack, in: Peer-to-Peer Systems, Springer Berlin / Heidelberg, 2002.

[13] D. Taylor, T. Wu, N. Mavrogiannopoulos, T. Perrin, Using the Secure Remote Password (SRP) Protocol for TLS Authentication, RFC 5054, 2007.

[14] A. S. Dan Boneh, B. Waters, Functional encryption: a new vision for public-key cryptography, Commun. ACM 55 (11) (2012) 56–64.

[15] A. Shamir, Identity-based cryptosystems and signature schemes, in: Proceedings of CRYPTO 84 on Advances in cryptology, Springer-Verlag New York, Inc., New York, NY, USA, 1985, pp. 47–53.
URL `http://dl.acm.org/citation.cfm?id=19478.19483`

[16] D. Boneh, M. K. Franklin, Identity-based encryption from the weil pairing, in: CRYPTO, 2001, pp. 213–229.

[17] C. Cocks, An identity based encryption scheme based on quadratic residues, in: IMA Int. Conf., 2001, pp. 360–363.

[18] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: EUROCRYPT, 2004, pp. 506–522.

[19] S. Hohenberger, B. Waters, Attribute-based encryption with fast decryption, in: Public Key Cryptography, 2013, pp. 162–179.

[20] L. Chen, Recommendation for Key Derivation Using Pseudorandom Functions, NIST Special Publication 800-108, 2009.

[21] D. Armstrong, K. Djemame, S. K. Nair, J. Tordsson, W. Ziegler, Towards a contextualization solution for cloud platform services, in: CloudCom, 2011, pp. 328–331.

[22] N. Doraswamy, IPSec : the new security standard for the Internet, intranets, and virtual private networks, 2nd Edition, Prentice Hall PTR, 2003.

[23] C. Kaufman, Internet key exchange protocol version 2 (ikev2), in: RFC 5996, 2010.

[24] W. Diffie, M. Hellman, New directions in cryptography, IEEE Transactions on Information Theory.

[25] F. I. P. S. P. 197, Announcing the advanced encryption standard (aes) (2001).

[26] R. Housley, Using advanced encryption standard (aes) ccm mode with ipsec encapsulating security payload (esp) (2005).

[27] N. Leavitt, Internet security under attack: The undermining of digital certificates, Computer 44 (12) (2011) 17–20. doi:10.1109/MC.2011.367.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6096548

[28] L. of the Bouncy Castle, Bouncy castle java cryptography apis (2013).
URL http://www.bouncycastle.org/java.html

[29] J. Bethencourt, A. Sahai, B. Waters, Advanced crypto software collection: The cpabe toolkit (2011).
URL http://acsc.cs.utexas.edu/cpabe/

[30] T. Bocek, Tomp2p: A p2p-based high performance key-value pair storage library (2012).
URL http://tomp2p.net/

[31] B. Telecom, Bt compute cloud (2013).
URL https://cloud.btcompute.bt.com

[32] Flexiant, Flexiant, your cloud simplified (2013).
URL http://www.flexiant.com/

[33] L. C. Ajay Tirumala, T. Dunigan, Measuring end-to-end bandwidth with iperf using web100, in: Web100, Proc. of Passive and Active Measurement Workshop, 2003.

[34] J. Yonan, OpenVPN - an open source SSL VPN solution.
URL http://openvpn.net/

[35] X. Jiang, D. Xu, VIOLIN: virtual internetworking on overlay INfrastructure, in: In Proc. Of The 2nd Intl. Symposium On Parallel And Distributed Processing And Applications, 2003.

[36] A. I. Sundararaj, P. A. Dinda, Towards virtual networks for virtual machine grid computing, in: In Proceedings of the 3rd USENIX Virtual Machine Research And Technology Symposium, 2004.

[37] Hamachi - a zero-configuration virtual private network.
URL https://secure.logmein.com/products/hamachi2

[38] L. Deri, R. Andrews, N2N: a layer two Peer-to-Peer VPN, in: Resilient Networks and Services, 2008.

[39] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, The Twofish encryption algorithm: a 128-bit block cipher, John Wiley & Sons, Inc., New York, NY, USA, 1999.

[40] Amazon, Virtual private cloud.
URL http://aws.amazon.com/vpc

[41] Google, Secure data connector.
URL http://code.google.com/securedataconnecto

[42] CohesiveFT, VPN-Cubed.
URL http://www.cohesiveft.com/vpncubed

Ali Sajjad received the BE degree from National University of Sciences and Technology, Pakistan, in 2003 and ME degree from Kyung Hee University, South Korea, in 2006. He is currently a PhD student in the School of Engineering and Mathematical Sciences at City University London, UK. His research interests are in the areas of applied cryptography and network security, with a current focus on secure communication and data services in cloud computing.

Muttukrishnan Rajarajan received his BEng and PhD degrees from City University London in 1994 and 1999 respectively. From 1999 he worked at City University London as a Research Fellow. In August 2000 he moved to Logica as a Telecommunication Consultant. After a few years in the industry Raj is now a Reader in Information Security Systems. He is also the Programme Director for the Engineering with Management and Entrepreneurship programme. Raj leads the Information Security Group (ISG) at City University London which carries out research in the domains of Cloud Security, Identity Management and Access Control and Lightweight Cryptography.

Andrea Zisman is a Professor in the Department of Computing, part of the School of Informatics at City University London, and a member of the Software Engineering research group in this department. Andrea holds a BSc degree in Computer Science from Catholic University of Pernambuco, Brazil, an MSc degree in Applied Mathematics to Computer Science from University of Sao Paulo, Brazil, and a PhD degree in Computer Science from Imperial College of Science, Technology & Medicine, London, UK. Andrea's research interests lie in the areas of software and service engineering and automated support of distributed data.

Theo Dimitrakos has over 15 years' experience in ICT including 10 years' experience in Trust and Information Security. He is a Chief Security Researcher in BT UK, where he leads internal research and innovation programmes and international research collaboration activities in Trust and Security for Cloud Computing, Federated Identity and Access Management, Intrusion Prevention and End-point Security. He is the chair of the IFIP working group on Trust management and a member of various industry and government advisory groups and forums. He holds a PhD from Imperial College London.