Alma Mater Studiorum - Università di Bologna

Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di laurea triennale in Informatica

PROGETTAZIONE ED IMPLEMENTAZIONE DI UNA PIATTAFORMA CLOUD COMPUTING PER L'EROGAZIONE DI MACCHINE VIRTUALI

Presentata da:
Vincenzo Tilotta

Relatore:
Dott. Vittorio Ghini

ANNO ACCADEMICO 2010-2011 SESSIONE II

Ai miei genitori...

Ringrazio Antonella e Giovanna per avermi regalato tanti sorrisi nelle giornate più cupe e avermi dato la forza di andare avanti durante tutto il mio percorso di studio.
Ringrazio il mio relatore per i preziosi consigli dati durante l'implementazione del progetto e la stesura di questo testo.

INTRODUZIONE

La diffusione dei dispositivi mobili e l'erogazione di servizi sempre più legati alla rete, favorisce una crescita dei servizi che implementano soluzioni di tipo Cloud Computig. Le aziende incrementano il loro business tramite l'erogazione di servizi implementati adottando un modello di tipo cloud. Allo stato attuale un dispositivo equipaggiato da più interfacce di rete, non sfrutta a pieno le sue potenzialità. Le attuali architetture distribuite dedicate alla gestione della mobilità non permettono di selezionare e utilizzare più interfacce contemporaneamente. Al contrario il modello Always Best Packet Switching (ABPS)[1] fornisce un meccanismo che permette alle applicazioni di usare tutte le interfacce disponibili e introduce nuove funzionalità con possibilità di creare politiche di controllo delle comunicazioni e bilanciamento di carico. Il nostro scopo è quello di implementare un servizio di Cloud Computing di tipo IaaS (Infrastructure as a Service) per l'erogazione di macchine virtuali al fine di dare supporto al VoIP[1], funzionante tramite modello ABPS.

Il testo risulta essere così suddiviso:

- CAPITOLO 1 → Parleremo in modo generale del Cloud Computing e di alcune tipologie che si possono implementare. Inoltre verranno affrontati alcuni problemi legati ad esso. Osserveremo in linea generale un servizio di tipo IaaS esistente.

- CAPITOLO 2 → Panoramica degli attuali strumenti utilizzati al fine di implementare servizi di cloud computing. Parleremo di strumenti completi per l'implementazione di servizi quali OpenNebula e sistemi di virtualizzazione tramite KVM.

- CAPITOLO 3 → Accenneremo dei protocolli di comunicazione

1 VoIP (Voice over IP) è una tecnologia che rende possibile effettuare una conversazione telefonica sfruttando una connessione Internet.

utilizzati in questo ambito utili a migliorare le prestazioni dei servizi cloud. Verranno affrontate soluzioni già esistenti al fine di garantire la mobilità mantenendo alti i parametri di QoE.

- CAPITOLO 4 → Parleremo dei concetti di sicurezza informatica in ambito di crittografia, integrità dei dati e protocolli di comunicazione sicuri TLS e SSH.

- CAPITOLO 5 → Verrà illustrato lo scenario di lavoro su cui si basa l'implementazione[2] della piattaforma.

- CAPITOLO 6 → Spiegheremo i passi seguiti nella fase di progettazione e parleremo del protocollo di comunicazione ideato al fine di rendere la piattaforma efficiente in termini di comunicazione di rete e carico di lavoro (CPU).

- CAPITOLO 7 → Verranno illustrate alcune parti salienti dell'applicazione e spiegate alcune soluzioni ai problemi riscontrati. In particolare parleremo delle soluzioni adottate al fine di garantire il multihoming e il rilascio delle risorse non più utilizzate dai client.

- CAPITOLO 8 → Illustrazione dei test effettuati sui vari componenti al fine di testare il multihoming e la distribuzione del carico di lavoro sull'infrastruttura. Forniremo alcuni problemi di cui soffrono i software utilizzati e accenneremo alcune possibili soluzioni.

- CAPITOLO 9 → Affronteremo il futuro fornendo alcuni suggerimenti su come ampliare la piattaforma. Cercheremo di immaginare la nostra piattaforma integrata su un sistema reale.

Indice generale

Elenco Delle figure

CAPITOLO 1

1. Il CLOUD COMPUTING

Il Cloud Computing[3] chiamato da molti “cloud” è un termine gergale che indica l'insieme delle tecnologie pronte a implementare servizi di archiviazione e/o elaborazione dati grazie all'utilizzo di infrastrutture hardware/software virtualizzate in rete. In particolare il Cloud Computing estende la possibilità per i dispositivi collegati in rete locale (LAN) o geografica (WAN) la possibilità di utilizzare software non direttamente installato sul dispositivo stesso. Sono tantissime le aziende che stanno adottando questo modello al fine di vendere il proprio software semplicemente offrendolo come servizio.

Gli aspetti fondamentali che racchiudono le caratteristiche del cloud computing sono:

- *Broad Network Access*: la rete è la componente fondamentale, viene garantito l'accesso remoto delle risorse disponibili.
- *On-demand Self Service*: gli utilizzatori dei servizi cloud possono richiedere le risorse (storage, applicativi, hardware) in relazione alle proprie esigenze. Inoltre chi usufruisce del servizio può gestire il proprio cloud (self service) senza l'intervento del fornitore del servizio.
- *Rapid Elastic*: gli utilizzatori del servizio possono configurare a proprio piacimento le risorse in base alle esigenze reali.
- *Mesured Service*: le risorse vengono tenute costantemente sotto controllo in tempo reale direttamente da chi utilizza il servizio.
- *Resource Pooling*: le risorse rese disponibili da chi offre il servizio vengono condivise e allocate dinamicamente da tutti gli utilizzatori.

Anche se il termine cloud indica in tutta la sua ampiezza un insieme di tecnologie è possibile distinguere varie tipologie di cloud computing. Basandoci sulla classificazione del NIST[4] possiamo distinguere tre

modelli di cloud computing:

- SaaS (Software as a Service): l'utente utilizza applicazioni (software) implementate nell'infrastruttura del fornitore, che è responsabile dell'assemblaggio e della manutenzione di tutte le risorse messe a disposizione. Questa tipologia di cloud risulta essere la più diffusa, in quanto le applicazioni possono essere eseguite utilizzando un browser web.
- PaaS (Platform as a Service): in questo tipo di modello il fornitore del servizio mette a disposizione una piattaforma completa, utile allo sviluppo di applicazioni, testing e deployment.
- IaaS (Infrastructure as a Service): software e hardware vengono utilizzati dagli utenti in remoto. Le risorse hardware vengono assegnate dal fornitore del servizio su richiesta dell'utente nel momento in cui quest'ultimo ne ha necessità.

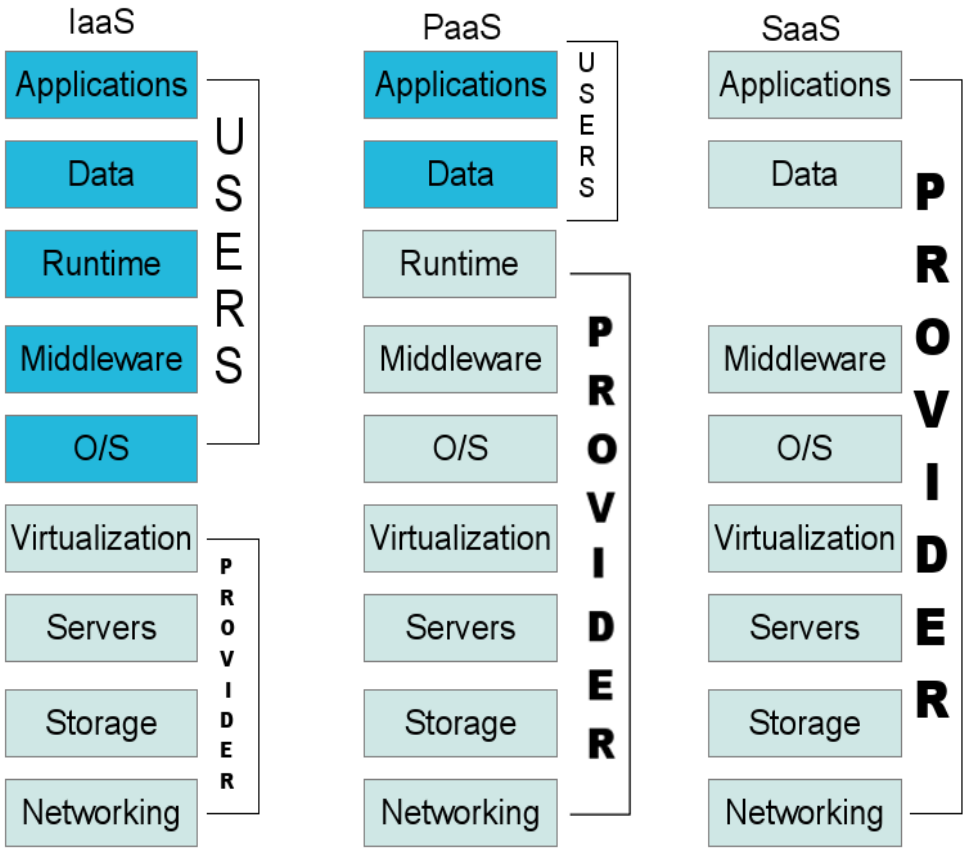

Illustrazione 1: Separazione delle responsabilità nel cloud computing

Questi tre modelli di cloud computing sono legati fra di loro ed hanno un livello di complessità differente in termini di sviluppo. In particolare il modello IaaS è alla base di qualunque infrastruttura che eroga servizi cloud.

1.1 Modelli di distribuzione

I modelli di distribuzione del cloud computing possono essere di quattro tipi:

- *Pubblica*: l'infrastruttura messa a disposizione è di proprietà della società che eroga il servizio, solitamente specializzata in vendita di soluzioni cloud. Le risorse messe a disposizione degli utenti vengono allocate e rilasciate dinamicamente dalla piattaforma.
- *Privata*: la piattaforma viene utilizzata esclusivamente da un'organizzazione specializzata. L'infrastruttura viene utilizzata e amministrata direttamente nell'edificio dell'organizzazione (on premise) oppure fuori dall'edificio (off premise).
- *Community*: l'infrastruttura viene condivisa fra più organizzazioni che lavorano nella stesso ambito. Solitamente queste organizzazioni condividono gli stessi interessi in ambito di missione e sicurezza.
- *Ibrida*: la piattaforma è composta da più modelli di distribuzione (pubblica, community, privata).

1.2 Pregi e difetti del cloud computing

L'implementazione di soluzioni cloud ha tantissimi vantaggi, soprattutto in termini di costi, ma anche tantissimi svantaggi. I punti di forza che hanno reso il cloud computing utilizzato da tantissime aziende sono la scalabilità e la flessibilità. Un utente può allocare e rilasciare le risorse in base alle esigenze del momento. Il software e l'hardware sono subito disponibili, quindi non occorre una pianificazione prima del loro utilizzo. Gli utenti pagano i servizi in base al loro utilizzo effettivo, risparmiando tanti soldi in termini di licenze (non proprietari del software). L'infrastruttura su cui lavorano gli utenti è costantemente aggiornata in quanto è il provider che provvede a risolvere tutte le problematiche.

Gli svantaggi che può portare una soluzione di tipo cloud è soprattutto in termini di disponibilità del servizio (availability of service). Eventuali guasti alla rete o alle componenti facenti parte dell'infrastruttura, possono portare una non disponibilità del servizio. Vista la grossa mole di software e hardware impiegato nell'implementazione di queste piattaforme, bisogna prevenire eventuali attacchi informatici dall'esterno.

1.3 Un servizio reale di tipo IaaS – Amazon EC2

Amazon è stata una delle prime aziende a progettare servizi di tipo cloud, offrendo servizi stabili e user friendly. Abbiamo soffermato l'attenzione sul servizio lanciato nell'agosto 2006, Amazon EC2 (Elastic Compute Cloud)[5]. Il servizio è una farm virtuale[2] in grado di offrire virtual machine ai loro utenti tramite una comoda interfaccia web. Con semplici passaggi è possibile creare istanze di macchine virtuali in differenti continenti del mondo. Attualmente è possibile localizzare le proprie macchine virtuali nelle seguenti località:

- Europa (Irlanda)
- Stati Uniti est (Virginia)
- Stati Uniti ovest (Nord della California)
- Asia, Pacifico (Singapore, Tokyo)

Gli utenti pagano il servizio in base all'utilizzo effettivo in termini di ore, spazio utilizzato, sistema operativo e banda utilizzata. Tutto questo semplifica in maniera significativa l'economia e la praticità delle aziende, che decidono di creare un loro servizio sfruttando un cluster virtuale:

- Capacità autonoma di rimuovere e aggiungere host in base al proprio business.
- Abbattimento dei costi in termini di installazione, upgrade e manutenzione di hardware fisico.
- Abbattimento dei costi nella gestione degli edifici, sparsi sulla

2 Farm virtuale, indica una “fattoria di server” virtuali, un insieme di server capaci di virtualizzare macchine virtuali su un ambiente centralizzato.

crosta terrestre, adibiti a contenere i server.

1.3.1 Funzionamento di Amazon EC2

Il funzionamento del servizio EC2 risulta essere alla portata di tutti. L'utente si registra al servizio e paga in base alle proprie esigente tramite una carta di credito. A questo punto tramite una comoda interfaccia web (illustrazione 2), lancia le proprie macchine virtuali e gestisce le istanze lanciate. Ogni istanza lanciata rappresenta una macchina fisica, con un proprio sistema operativo, pronta all'uso e connessa alla rete con un indirizzo IP.

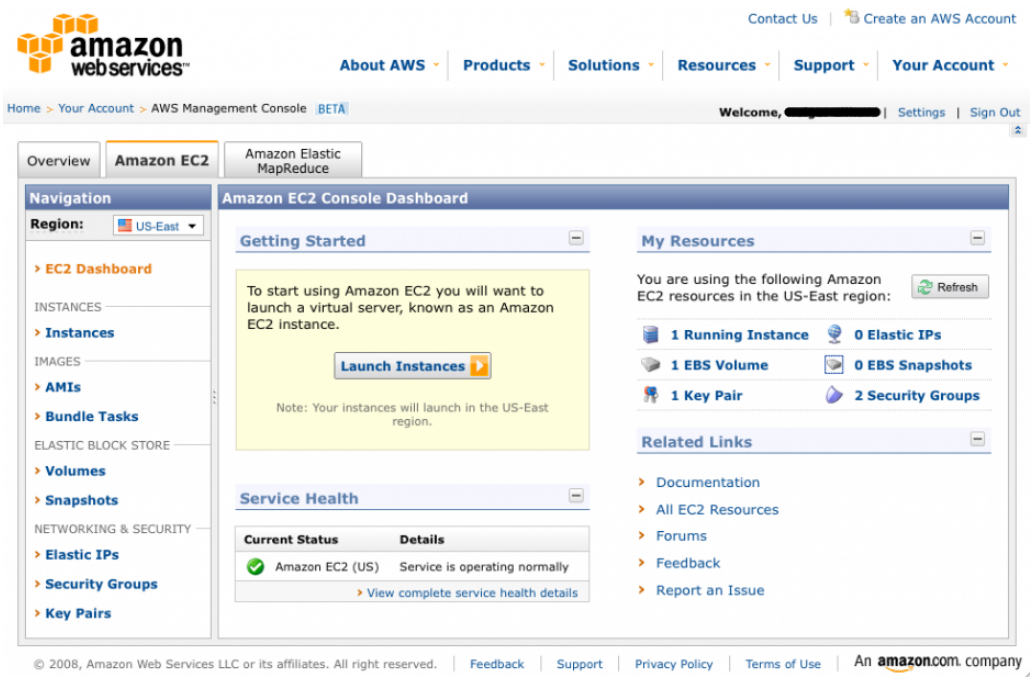

Illustrazione 2: Pannello di controllo per la gestione del servizio EC2

Amazon offre la possibilità di scegliere per le proprie macchine virtuali una serie di sistemi operativi già configurati e pronti all'uso inclusi in file chiamati AMI [3]. Esiste la possibilità di "creare la propria AMI"[6] includendo il proprio sistema operativo con le proprie configurazioni.

3 AMI (Amazon Machine Images) sono dei file immagine che contengono sistemi operativi.

CAPITOLO 2

2. SOFTWARE ORIENTATO AL CLOUD COMPUTING

Il software si evolve verso il cloud computing. Crescono sempre di più il numero delle librerie e dei software open source, che orientano il proprio sviluppo verso una prospettiva cloud. L'utilizzo del software open source nell'ambito cloud computing diventa indispensabile al fine di rendere la propria piattaforma efficiente e stabile. In questo contesto nasce Open Virtualization Alliance[7] un consorzio formato da grandi aziende finalizzato a favorire l'adozione di virtualizzazioni open source tra cui KVM[8] e OpenNebula[9].

2.1 KVM - Kernel-based Virtual Machine

Abbiamo scelto KVM come ambiente di virtualizzazione in quanto favorisce un elevatissima scalabilità, supporta 4096 core (macchine host) e sfrutta fino a 64 TB di memoria RAM. Le macchine "guest" possono utilizzare fino a 64 CPU virtuali e supportare fino a 2 TB di memoria RAM. KVM sfrutta il supporto di virtualizzazione per i processori Intel e AMD (Intel VT o AMD-V). L'inclusione sul Kernel Linux a partire dalla versione 2.6.20 garantisce la personalizzazione delle soglie d'uso di CPU, rete e dischi. Ogni macchina virtuale si comporta come un qualsiasi processo in un sistema linux. Inoltre è previsto un supporto nativo del protocollo RFB per rendere le macchine virtuali raggiungibili dall'esterno. Un tipico esempio del lancio di una macchina virtuale accessibile da remoto tramite un client VNC è dato da:

qemu-system-x86_64 -cdrom my_os.iso -m 384 -vnc :5

Il comando sopra citato attiva una macchina virtuale da un file ISO. Vengono messi a disposizione della macchina guest 384 MB di ram. Il sistema è accessibile dall'esterno tramite la porta 5005.

Software di virtualizzazione di questo tipo solitamente vengono

integrati in progetti per lo sviluppo di architetture cloud come ad esempio OpenNebula.

2.1 OpenNebula un toolkit per il cloud computing

OpenNebula è un insieme di strumenti open source in grado di creare e gestire macchine virtuali in maniera dinamica, all'interno di architetture hardware molto complesse. OpenNebula espone diverse interfacce che consentono all'utente di amministrare e monitorare tutte le fasi di virtualizzazione. Le funzioni implementate nel front-end permettono di aprire, chiudere, spostare le macchine virtuali fra i vari calcolatori che fanno parte dell'infrastruttura.

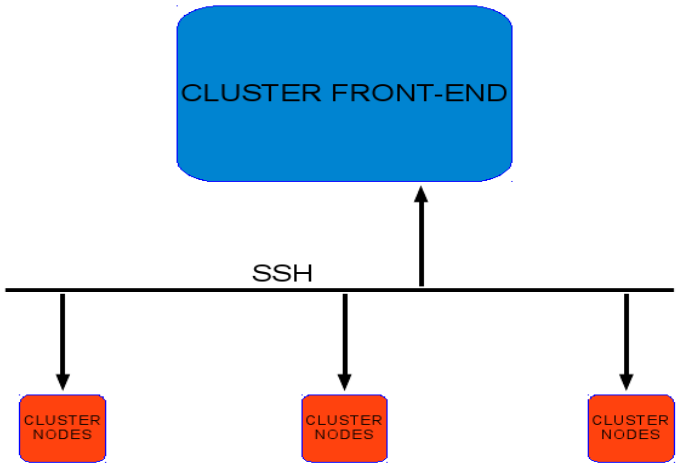

Illustrazione 3: Schema funzionamento OpenNebula

L'illustrazione 3 mostra uno schema sul funzionamento di OpenNebula. Tutte le operazioni compiute sui nodi tramite il front-end vengono eseguite mediante il protocollo SSH[10], offrendo una gestione centralizzata di tutte le infrastrutture fisiche e virtuali create. Le tipologie di macchine virtuali e la loro configurazione di rete, viene stabilita da un apposito file di configurazione (VM Template). Tipicamente un template definisce:

- le risorse fisiche da assegnare alla macchina virtuale.

- la quantità di memoria.
- il numero di processori.
- il nome del sistema.
- la tipologia di rete.

Un tipico file di configurazione è rappresentato nell'illustrazione 4. Nell'immagine viene configurata una macchina virtuale, settando 512 MB di memoria RAM e limitando l'utilizzo di CPU al 50%. Inoltre vengono fornite alcune informazioni che riguardano il sistema operativo e le informazioni riguardanti la rete e il server VNC.

```
NAME   = vm01
CPU    = 0.5
MEMORY = 512

OS      = [ BOOT   = hd ]

DISK   = [
  source   = "/var/lib/one/images/vm01.qcow2",
  target   = "hda",
  readonly = "no" ]

NIC    = [ NETWORK="LAN" ]

GRAPHICS = [type="vnc",listen="127.0.0.1",port="-1"]
```

Illustrazione 4: Esempio di un template OpenNebula

CAPITOLO 3

3. PROTOCOLLI ORIENTATI AL CLOUD COMPUTING

I protocolli di comunicazione si evolvono con i dispositivi multihomed, che utilizzano sempre di più la rete per accedere a servizi di comunicazione tramite VoIP o/e servizi di cloud computing. È bene ricordare che l'adozione del VoIP con dispositivi mobili multihomed spesso risulta essere limitante in termini di QoS[4]. Il cambiamento dell'indirizzo IP durante una chiamata VoIP audio e video, comporta una rottura temporanea dello streaming[5]. Questo fenomeno comporta una disconnessione temporanea alla rete riducendo notevolmente il QoE[6]. La necessità di superare gli attuali limiti di cui soffrono le recenti architetture distribuite, dedicate alla gestione della mobilità è diventata una priorità. L'adozione del modello ABPS supera i limiti sopra citati.

3.1 Panoramica al VoIP

Il VoIP (Voice Over Internet Protocol) è l'insieme delle tecnologie che permette la trasmissione di voce in tempo reale tramite la rete. I pacchetti dati contenenti le informazioni vocali in formato digitale vengono istradati nella rete, nel momento in cui uno dei partecipanti alla conversazione emette un suono tramite un apposito microfono. Il corretto funzionamento del VoIP richiede l'uso in parallelo di due differenti protocolli di comunicazione. Il protocollo RTP (Real-time Transport Protocol) si occupa del trasporto dei dati e la descrizione delle sessioni multimediali tramite SDP (Session Description Protocol). La gestione delle chiamate viene spesso affidata a due protocolli differenti supportati da due grandi enti di standardizzazione.

4 QoS (Quality of Service) il termine indica la qualità del servizio in termini di comunicazioni di rete

5 Streaming indica un flusso di dati trasmessi da una sorgente a una destinazione

6 QoE (Quality of experience) indica la qualità dell'esperienza dell'utente.

L'ITU (International Telecommunication Union) favorisce la diffusione del protocollo H.323 invece l'IETF favorisce il supporto al protocollo SIP (Session Initiation Protocol) ormai diventato diffusissimo. Fortunatamente quasi tutti i dispositivi immessi sul mercato supportano entrambi i protocolli

3.2 Il protocollo SIP

Il protocollo SIP si occupa di gestire le sessioni (modificare, terminare, instaurare) tra due o più entità. Il protocollo è costruito a strati come visibile sotto.

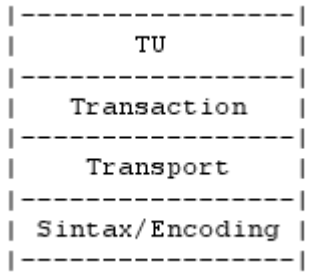

Illustrazione 5: Strati protocollo SIP

Il livello più in basso si occupa della sintassi e della codifica delle informazioni seguendo il sistema di codifica Bakus-Naur[7]. Lo strato di trasporto definisce come vengono inviate le richieste e ricevute le risposte. Il terzo strato è quello di transazione, esso si occupa di supervisionare le transazioni tra il client e server. In particolare il livello di transazione tiene traccia delle richieste, risposte e timeout. Esso ha il compito di ritrasmettere i messaggi per i quali è stato raggiungo il timeout. Subito sopra lo strato "Transaction" ritroviamo lo strato TU (Transaction User), esso si occupa di creare un'istanza di transazione (per ogni richiesta inviata) a cui associa un protocollo di trasporto, una porta e un indirizzo IP di destinazione. Lo strato TU è inoltre addetto a cancellare l'istanza creata generando una risposta d'errore per quella transazione. Il punto di forza del protocollo è la scalabilità derivata dal sistema di instaurazione delle sessioni.

7 BNF (Bakus-Naur Form) è un formalismo grazie al cui è possibile descrivere la sintassi in linguaggi formali

Il server gestisce le connessioni degli utenti solo durante la fase iniziale, successivamente il flusso (dati) della conversazione avviene tra gli interlocutori. Questo garantisce al server di gestire numerose connessioni. Gli agenti che interagiscono sono:

- *SIP Registrar Server*: Il componente si occupa di registrare al servizio SIP tutti gli UA (User Agent) che effettuano la registrazione tramite un messaggio REGISTER. Una volta accettata la richiesta di registrazione il Registar Server provvederà a registrare questa associazione all'interno del suo user location database. Tale database verrà aggiornato ad ogni richiesta di registrazione.
- *SIP Redirect Server*: questo componente fornisce all'UA l'indirizzo del destinatario. Una volta ricevuto l'indirizzo, il chiamante può effettuare un collegamento autonomo.
- *SIP Proxy Server*: ha funzionalità di intermediario, esso risponde direttamente alle richieste oppure le innoltra a un client, server o proxy.

3.3 Il protocollo RFB

RFB[11] (remote framebuffer) nasce con lo scopo di poter controllare un desktop remoto. Il protocollo per la gestione delle operazioni nel desktop è stato reso estremamente semplice, in modo da ridurre al minimo le esigenze del client VNC connesso al Server RFB. Gli agenti che solitamente vengono coinvolti nell'implementazione del protocollo sono un client e un server.

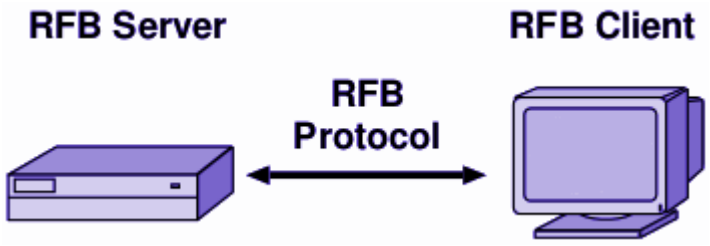

Illustrazione 6: Funzionamento del protocollo RFB

Il client è l'applicazione più semplice coinvolta nel protocollo. Essa si limita a ricevere dal server i dati già compressi e pronti per essere stampati a video. Al contrario il Server risulta essere più complesso rispetto al client, in quanto si occupa della codifica dei pixel in modo da risultare compatibile con il client. Le implementazioni più efficienti ottimizzano l'invio dei dati inviando soltanto le porzioni dello schermo che sono state aggiornate (solitamente quelle vicino al cursore). I server VNC sono dotati di un elementare HTTP server in modo da consentire il controllo remoto della macchina virtuale direttamente tramite browser. Dal punto di vista della sicurezza, implementa un meccanismo per proteggere il desktop remoto tramite password. È bene sottolineare che la cifratura avviene soltanto in fase di autenticazione, il resto delle comunicazioni rimane in chiaro.

3.4 Mobilità senza confini con ABPS

Per mobilità senza confini (Seamless Mobility) intendiamo l'insieme delle architetture, che rendono possibile ad ogni nodo della rete di essere raggiunto da altri nodi. Questa soluzione viene adottata al fine di monitorare i parametri QoS e selezionare la rete migliore. In un contesto di comunicazione IP-based nello specifico VoIP, l'indirizzo IP rappresenta un identificatore univoco per l'host. Essendo l'host un dispositivo mobile, la sua capacità di cambiare indirizzo IP e interfaccia di rete (MAC address) frequentemente, vanifica la possibilità di essere identificato tramite il suo indirizzo IP. Il seguente fenomeno causa continue discontinuità nelle comunicazioni. Questo rappresenta un problema per tutti i nodi, che in precedenza avevano instaurato una comunicazione con il dispositivo, in quanto non conoscono il nuovo indirizzo IP. La soluzione comune al problema adottata nei dispositivi mobili si basa su due principi fondamentali:

1. Ogni terminale mobile definisce un identificatore univoco, che lo distingue indipendentemente dall'interfaccia o dall'indirizzo di rete.
2. Fornire un servizio per la localizzazione: ogni identificatore deve essere associato al suo reale indirizzo di provenienza.

Le attuali architetture utilizzate per la Seamless Mobility sono basate su IPv6[12]. Questo costringe tutti i dispositivi che partecipano alle comunicazione a supportare IPv6. Inoltre tali architetture costringono

tutti i nodi ad inserire uno strato intermedio tra il livello di rete e quello di trasporto, cosa non sempre favorevole specialmente su nodi fissi. Le architetture attualmente utilizzate sono:

- MIP (Mobile IP versione 6)
- FMIP (Fast Handover Mobile Ipv6)
- HMIP (Hierarchical Mobile Ipv6)
- Proxy Mobile Ipv6 (PMIP)

Una soluzione al problema citato sopra è l'adozione del modello ABPS in grado di rispettare a pieno i requisiti QoS. Dal modello possiamo distinguere due componenti principali.

1. *SIP mobility*: entità a livello applicazione che si occupa di riconfigurare un indirizzo IP.
2. *Vertical Mobility*: entità a livello data-link e network il cui compito è quello di monitorare lo stato delle interfacce di rete installate sul dispositivo. La scelta dell'interfaccia avviene sulla base di opportune metriche e politiche.

Il componente SIP mobility risiede su un terminale mobile, invece il Vertical Mobility è posto su un server collegato su una rete pubblica. Il server rappresenta il nodo centrale per tutte le comunicazioni VoIP. Ogni dispositivo mobile che intende effettuare una comunicazione dovrà prima fare una richiesta ad esso, il quale si occuperà di gestire le sessioni. I dispositivi che accedono al server per effettuare le comunicazioni rimangono all'oscuro dei meccanismi adottati in ambito protocollare (SIP e RTP/SRTP).

La struttura dell'applicazione (client) presente sul dispositivo mobile è dotata di un architettura a 4 livelli:

1. Il primo livello è il più basso dell'architettura, esso si occupa della realizzazione dei meccanismi di handover[8] e load balancing[9].
2. Il secondo strato è addetto a garantire autenticazione mutuale e

8 Handover è un meccanismo adottato sui dispositivi mobili che permette di cambiare il canale di comunicazione

9 Load Balancing è una tecnica che permette di bilanciare il carico di rete fra le varie componenti che interagiscono alla comunicazione.

firma dei pacchetti.

3. Questo livello rappresenta il protocollo SIP
4. L'ultimo livello è l'interfaccia (GUI) utente che permette di gestire le funzionalità implementate.

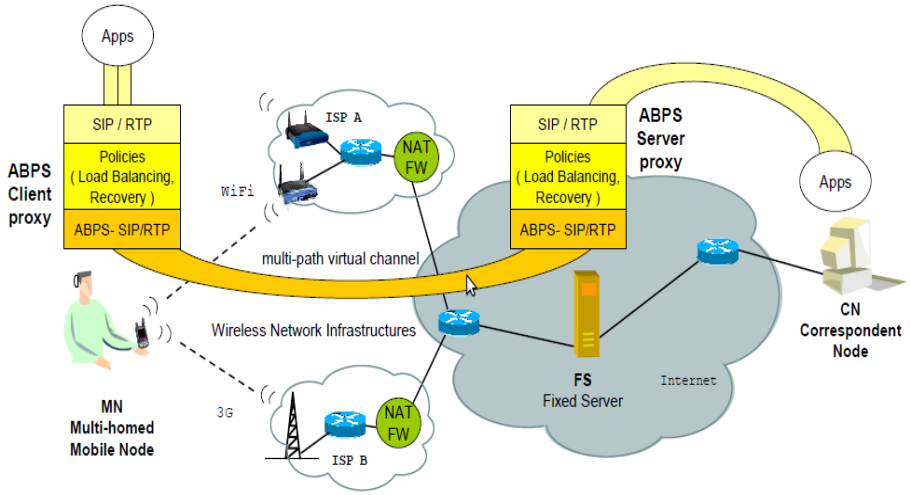

Illustrazione 7: Scenario ABPS

L'illustrazione 7 rappresenta uno scenario in cui agisce il modello ABPS:

- L'applicazione (Client ABPS) installata sul dispositivo mobile comunica con il server al fine di ottenere la comunicazione.
- Il server (Proxy ABPS) si occupa di gestire la comunicazione scambiando messaggi con il client.

La soluzione adottata dal modello ABPS è in grado di sfruttare un sistema VoIP distinguendo il traffico proveniente da un dispositivo senza basarsi su IP. In questo contesto il dispositivo può inviare dati da più indirizzi IP e da diverse interfacce anche contemporaneamente, mantenendo la continuità nella comunicazione.

CAPITOLO 4

4. CONCETTI DI SICUREZZA

Le architetture cloud sono tipicamente risorse condivise utilizzate da varie applicazioni. Questo comporta lo scambio di informazioni spesso sensibili e/o private. Nasce così l'esigenza di utilizzare comunicazioni cifrate in maniera tale da coprire diversi aspetti in ambito di sicurezza informatica:

- *privacy* - prevenire il rilascio di informazioni non autorizzato.
- *autenticazione* - verifica dell'identità degli utenti coinvolti nelle comunicazioni.
- *integrità* - garantire che un messaggio arrivato a destinazione non risulti alterato.

Il concetto di cifratura risulta essere un'operazione abbastanza semplice. Il mittente cifra il messaggio originale (plaintext) tramite una funzione cifrante. Il messaggio cifrato (ciphertext) viene inviato al destinatario, esso potrà leggere il testo in chiaro applicando una funzione decifrante in maniera tale da ricostruire il testo originario. Questo procedimento di cifratura / decifratura dipende in generale da una chiave segreta, (nel caso di cifratura a chiave segreta) condivisa da entrambi gli interlocutori. L'operazione di lettura da parte di un estraneo in ascolto alle comunicazioni, diviene sufficientemente ardua in quanto non conosce la chiave segreta. Esistono diversi algoritmi di cifratura, fra i più diffusi ed efficienti troviamo DES (Data Encryption Standard), AES (Advanced Encryption Standard), RSA (algoritmo di Rivest, Shamir e Adleman) e MD5 (Message Digest 5). Tramite questi algoritmi è stato possibile implementare protocolli di comunicazione che oggi risultano essere cardini della sicurezza in ambito delle comunicazioni di rete. In particolare i principali protagonisti delle attuali comunicazioni sicure risultano essere TLS (Transport Layer Security) e SSH (Secure Shell).

4.1 Crittografia simmetrica

La crittografia simmetrica chiamata anche a chiave privata è una tecnica di cifratura. Questo tipo di algoritmi utilizzano una singola chiave, condivisa fra due interlocutori, utile sia a cifrare che a decifrare. Il sistema simmetrico fornisce a due interlocutori un canale bidirezionale con cui scambiare informazioni in maniera confidenziale. Fin quando la chiave rimane segreta il sistema garantisce la proprietà di autenticazione, ossia la garanzia che il messaggio giunto a destinazione non è stato alterato da una persona diversa dal mittente. Questi tipi di algoritmi hanno un problema intrinseco che riguarda la distribuzione delle chiavi. I due interlocutori devono scambiare la chiave per poter comunicare, cosa che potrebbe comportare dei rischi nel caso in cui un agente esterno intercetti la chiave condivisa dai due.

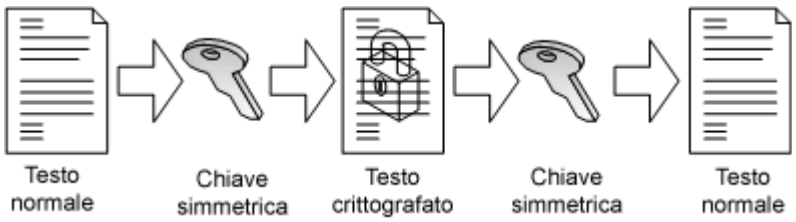

Illustrazione 8: Crittografia a chiave privata

Gli algoritmi attuali (più diffusi) che utilizzano questa tecnica di crittografia sono DES e il nuovo standard AES. Gli algoritmi simmetrici vengono solitamente affiancati da algoritmi di tipo asimmetrici, per compensare le lacune riguardanti la distribuzione delle chiavi.

4.2 Crittografia asimmetrica

La crittografia asimmetrica chiamata anche a chiave pubblica è una tecnica di crittografia con il quale ogni interlocutore viene munito di una coppia di chiavi:

- chiave pubblica, ha il compito di cifrare i messaggi destinati al mittente che possiede la chiave privata. Questa chiave deve essere distribuita su tutti i canali di comunicazione pagina web, firma mail, Key Distribution Center.

- chiave privata, solitamente in possesso dall'entità che può decifrare i messaggi precedentemente crittografati tramite la chiave pubblica.

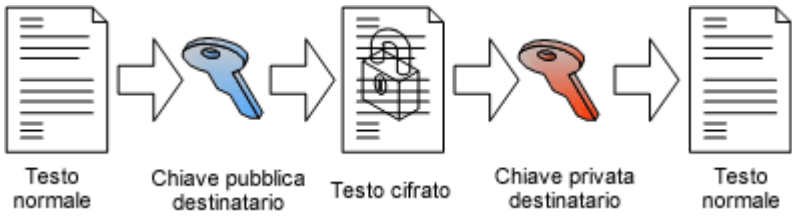

Illustrazione 9: Crittografia a chiave pubblica

I più comuni algoritmi di crittografia asimmetrica come ad esempio RSA si appoggiano al classico problema della fattorizzazione di un numero in fattori primi. Allo stato attuale compiere una scomposizione di un numero (sufficientemente grande) nei suoi fattori primi risulta essere complesso a livello computazionale. Alcuni degli algoritmi più diffusi che utilizzano una tecnica di crittografia asimmetrica sono RSA, DSS (Digital Signature Standard), Diffie-Hellman (Diffie-Hellman key exchange). Anche in questa tipologia di algoritmi è necessario mantenere segreta la chiave privata. L'eventuale violazione della chiave privata compromette l'integrità e la confidenzialità di tutti i messaggi scambiati.

4.3 Cifratura a senso unico

Nel momento in cui si ha la necessità di nascondere un'informazione riservata, come ad esempio una password memorizzata dentro un database, abbiamo bisogno di cifrare l'informazione in maniera tale da non poter compiere l'operazione inversa. Per fare ciò utilizziamo delle funzioni di hash chiamate anche cifratura a senso unico (one-way hash function). I più noti ed efficienti algoritmi di hash sono MD5[13], SHA1 / SHA256 (Secure Hash Algorithm), HMAC[14] (keyed-hash message authentication code). Nella tabella sotto abbiamo applicato alcuni algoritmi di hash, con caratteristiche differenti, alla stringa “DON'T PANIC”.

ALGORITMO	OUTPUT
MD5	507c82010eaea3819cd43b7d4718d988

SHA1	66eaf30d8c3acc73a21e052d1e817285f481d741
SHA256	5b7a2fc17a5be71049b7ae1b4f510e2cdcf85b5127ad3cd870ac804c3454e2b8

L'informazione viene trasformata in una stringa incomprensibile, in maniera tale da non mostrare il vero contenuto nel caso di attacchi. È da notare che l'operazione inversa risulta essere particolarmente difficoltosa.

4.4 Prestazione degli algoritmi di cifratura

Dal punto di vista prestazionale[15] DES e MD5 risultano essere diversi ordini di grandezza più veloci di RSA. Con processori di ultima generazione l'algoritmo DES elabora i dati ad una velocità di circa 100 Mbps e MD5 a circa 600 Mbps, invece RSA arriva soltanto a circa 100 Kbps. Per questa ragione le cifrature a chiave pubblica e quelle a chiave privata vengono miscelate al fine di migliorare le prestazioni nelle comunicazioni. In moltissime architetture viene infatti utilizzato in un primo momento RSA, al fine di scambiare le chiavi private e successivamente DES.

4.5 Il protocollo TLS

TLS[16] è un protocollo standard sviluppato al fine di migliorare il vecchio protocollo SSL. Il protocollo TLS si colloca ad un livello superiore rispetto ai protocolli di trasporto di tipo affidabile ed a un livello inferiore rispetto al livello applicazione. Questo consente il suo utilizzo su svariati software.

Il protocollo TLS è composto da due livelli:

- Livello *record protocol*: è il livello inferiore che si occupa della sicurezza a livello di connessione e di trasmissione.

Il livello record protocol utilizza algoritmi simmetrici quali DES e RC4. Ogni messaggio viene corredato da un parametro di verifica di integrità dei dati (message integrity check) definito da un attributo MAC keyed. Le computazioni a livello MAC vengono effettuate

sfruttando le più comuni funzioni di hash sicure come MD5 e SHA1.

- Livello *client*: questo livello è collocato sopra il livello record. In questo strato vengono definiti tre diversi protocolli TLS Handshake, TLS Change Cipher Spec e TLS Alert.

Il protocollo TLS Handshake si occupa di autenticare l'interlocutore utilizzando tecniche di cifratura che possono essere a chiave pubblica o privata. Successivamente il protocollo TLS handshake si occupa di trasmettere al livello record protocol l'esito dell'autenticazione. Sarà il protocollo record a occuparsi della cifratura dei dati. I protocolli TLS Change Cipher Spec e TLS Alert riguardano per lo più il livello client.

Il protocollo TLS Alert si occupa della gestione dei messaggi di errore o di avviso (warning), inviati nel caso in cui ci siano errori in ambito di cifratura o a livello di certificati di sicurezza. TLS Change Cipher Spec si occupa di inviare un singolo byte in maniera tale da segnalare al client il tipo di strategia di cifratura da utilizzare.

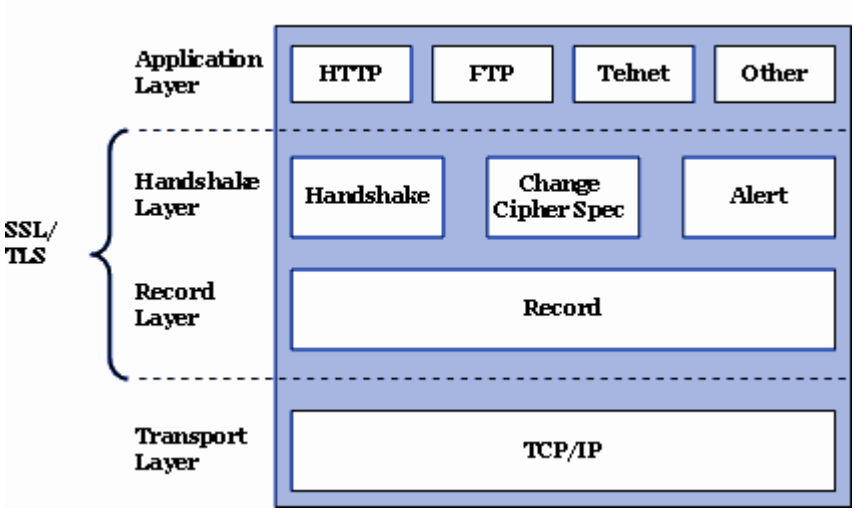

Illustrazione 10: Strati e struttura del protocollo TLS

4.6 Il protocollo SSH

SSH è un protocollo di rete utile a fornire un servizio di accesso remoto. Nasce con lo scopo di sostituire telnet, ormai marcato come obsoleto[17] dal IETF (Internet Engineering Task Force), in quanto

non implementa meccanismi di autenticazione e protezione delle comunicazioni. SSH ha una struttura di tipo client/server, l'utente utilizza il client per accedere in modo sicuro su una macchina remota che ospita il server. La struttura del protocollo SSH è divisa in tre strati come ben visibile nell'illustrazione 11.

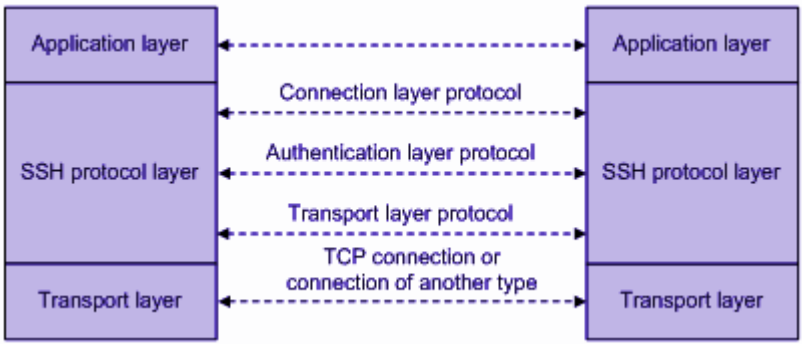

Illustrazione 11: Architettura del protocollo SSH

L'ultima versione di SSH, la versione 2 è strutturata in tre protocolli:

- SSH-TRANS: apre un canale di comunicazione sicura fra client e server.
- SSH-AUTH: implementa i meccanismi di autenticazione.
- SSH-CONN: implementazione del “tunnel ssh” e del port forwarding

Nel momento in cui un utente vuole accedere ad una macchina remota tramite SSH verrà prima applicato il protocollo SSH-TRANS, il cui obbiettivo è fornire un canale di comunicazione sicuro, utilizzando RSA. In questo modo client e server possono facilmente “accordarsi” su una chiave di sessione, in maniera tale da utilizzare per le future comunicazioni un algoritmo di cifratura sincrono, solitamente utilizzando Triple DES (DES triplo). A questo punto il client può autenticarsi al sistema scegliendo uno dei tre meccanismi che SSH mette a disposizione:

1. Il client si autentica al server comunicando la password di accesso.
2. Questo meccanismo prevede l'utilizzo di cifratura a chiave pubblica: l'utente invia preventivamente la propria chiave pubblica al server.
3. L'ultimo meccanismo chiamato anche autenticazione basata su

host (host-based authentication) prevede che un certo insieme di host, considerati fidati dal server, vengano automaticamente autenticati al sistema.

In maniera del tutto opzionale è stato implementato SSH-CONN al fine di fornire supporto alle applicazioni considerate insicure basate su TCP. In particolare è possibile lanciare le applicazioni considerate insicure facendo passare il traffico di un'applicazione tramite un “tunnel SSH”. Questo meccanismo viene chiamato port-forwarding implementato da SSH-CONN.

CAPITOLO 5

5. SCENARIO

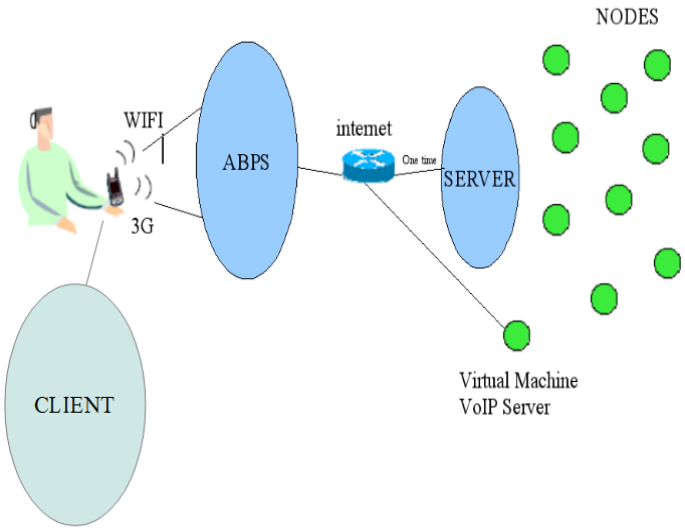

Illustrazione 12: Scenario

Abbiamo ipotizzato uno scenario che vede un PDA[10] equipaggiato di più interfacce di rete accedere ad una VM[11] lanciata in remoto su un calcolatore. Il dispositivo utilizza tutte le interfacce di rete grazie all'adozione del modello ABPS. Questo comporta continui cambiamenti di MAC address e indirizzi IP. Il nostro studio si impone come obbiettivo quello di rendere efficiente la connettività fra il PDA e il calcolatore che ospita la VM. Questo avverrà grazie all'implementazione di un protocollo di comunicazione che selezionerà il calcolatore più "vicino" al PDA in base alla latenza calcolata tramite pacchetti ICMP[12] e in base al carico di lavoro. Nel nostro caso è stato essenziale sfruttare le attuali tecnologie che permettono di implementare il Cloud Computing al fine di distribuire

10 PDA (Personal Digital Assistant) detto anche palmare, è un computer di dimensioni contenute.
11 VM (Macchina Virtuale) è un software in grado di emulare il comportamento di una macchina fisica
12 ICMP (Internet Control Message Protocol) è un protocollo che si occupa di trasmettere informazioni riguardanti malfunzionamenti, controllo o messaggi fra i componenti di una rete.

il carico di lavoro generato dalle macchine virtuali sull'hardware disponibile.

CAPITOLO 6

6. PROGETTAZIONE DELLA PIATTAFORMA

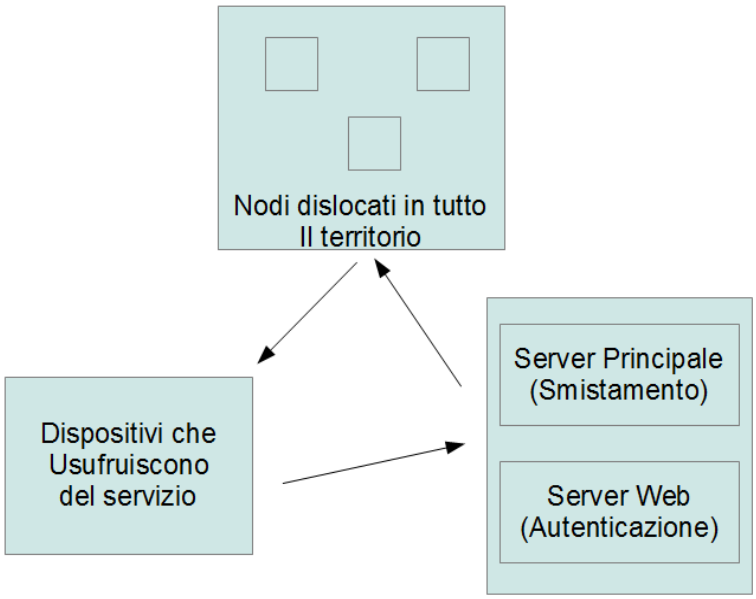

Illustrazione 13: Architettura della piattaforma

Ci siamo messi nei panni di un'azienda che vuole fornire macchine virtuali ai suoi clienti, inventando una piattaforma di cloud computing utile ad erogare questo servizio. L'architettura (vedi illustrazione 13) da noi implementata prevede l'implementazione di 3 componenti essenziali:

- Applicazione web si occupa di registrare ed autenticare i dispositivi che vogliono accedere al servizio.
- Server principale instaura le comunicazioni con i client e smista il carico di lavoro sui nodi.
- Nodi (Calcolatori), dislocati su tutto il territorio, utilizzati per ospitare le macchine virtuali.

- Client installato sul dispositivo, si occupa di comunicare con il server centrale al fine di ottenere il servizio richiesto.

Tutte le componenti comunicano fra di loro tramite un protocollo di comunicazione di cui parleremo nel dettaglio nel paragrafo successivo. La struttura dell'architettura prevede dei vincoli che impongono ai client una comunicazione a due fasi:

1. Registrazione al server web tramite protocollo HTTP, questa fase è eseguita soltanto al primo accesso.
2. Autenticazione al server principale e richiesta di servizio, queste operazioni dovranno essere eseguite ogni tal volta che si vuole accedere al servizio

I vantaggi di usare questo approccio sono molteplici. Tutti i client registrati al servizio vengono inseriti in un log file contenente l'indirizzo IP dell'utilizzatore (prevenzione dei reati informatici). I nodi non hanno dipendenze dal sistema, questo garantisce il corretto funzionamento anche nel caso in cui uno o più nodi della rete risultasse offline.

6.1 Progettazione del protocollo

Per completare la nostra architettura abbiamo bisogno di definire un protocollo di comunicazione che ci garantisca i seguenti aspetti:

- I client connessi alla piattaforma possono utilizzare il servizio anche se sotto NAT[13]. Spesso alcuni utenti si ritrovano a navigare su internet con ISP[14] che forniscono connettività sotto NAT, quindi non raggiungibili direttamente.
- Prevenire attacchi di tipo MITM[15], tutte le comunicazioni fra client e server devono essere cifrate.
- Il server deve “guidare” i nodi a non sprecare le risorse hardware. Evitare sprechi di energia e costi di manutenzione dell'hardware.
- I nodi non devono partecipare attivamente al protocollo. Questo

13 NAT (Native Address Translation) è una tecnica che consiste nel modificare gli indirizzi IP dei pacchetti in transito su un sistema che agisce da router.
14 ISP (Internet Service Provider) è il fornitori di servizi internet.
15 MITM (man in the middle) è un tipo di attacco informatico, l'attaccante è in grado di leggere le informazioni scambiate fra due interlocutori.

garantisce il corretto funzionamento nel caso in cui ci fossero guasti.

- La macchina virtuale deve essere accessibile da qualunque interfaccia di rete e IP.
- Il dispositivo mobile che accede alla macchina virtuale non deve ripetere la fase di autenticazione ogni tal volta che si collega alla macchina virtuale.

La prima fase del protocollo di cui ci occuperemo è quella di registrazione e autenticazione. In questa fase vi è uno scambio di informazioni fra il dispositivo che vuole sfruttare il servizio e il server centrale. Nell'illustrazione 14 è possibile visualizzare uno schema dei messaggi scambiati.

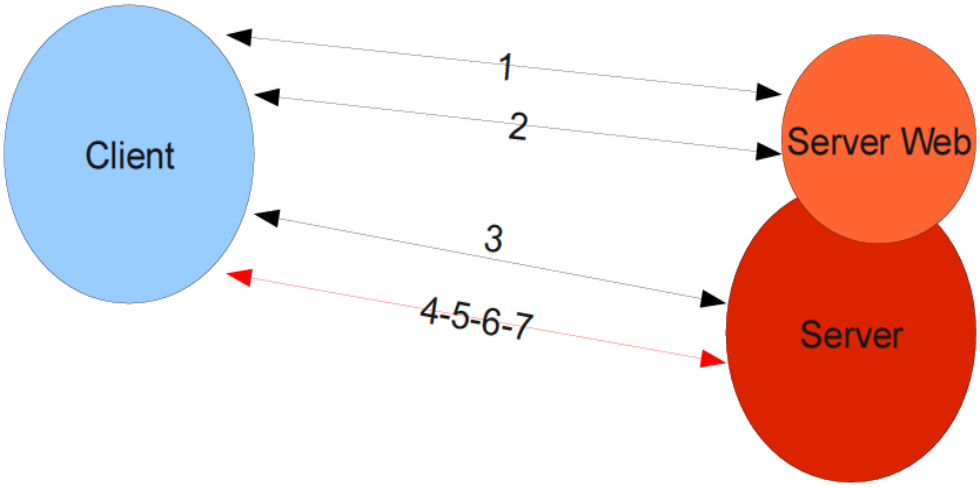

Illustrazione 14: Fase di registrazione e autenticazione

Al fine di semplificare le cose abbiamo numerato i passaggi delle comunicazioni in ordine temporale:

1. Il client genera una coppia di chiavi a 2048 bit tramite algoritmo RSA[16] e invia al server web tramite POST HTTP la chiave pubblica corredata di un "username" generato casualmente (caratteri alfa-numerici). La chiave RSA viene

16 RSA (Rivest, Shamir, Adleman) è un algoritmo di crittografia asimmetrica, utilizzabile per cifrare o firmare informazioni.

generata con il tool OpenSSL[18].

2. Il server web si occupa di memorizzare le informazioni inviate dal dispositivo in modo da riutilizzarle durante la fase di autenticazione. Le chiavi vengono memorizzate in base all'username direttamente sul filesystem.
3. In questo passo il client instaurata una connessione TLS con il server centrale, in modo da rendere tutte le comunicazioni successive confidenziali.
4. Il client invia al server il suo username.
5. Il server invia al client un numero casuale, criptato con la chiave pubblica del client stesso.
6. Il client decifra il crittogramma con la propria chiave privata e lo invia al server centrale.
7. Il server confronta il numero generato in precedenza con il messaggio appena ricevuto dal client. Se il messaggio è identico, il client è autorizzato a fare una richiesta di servizio altrimenti la connessione viene interrotta.

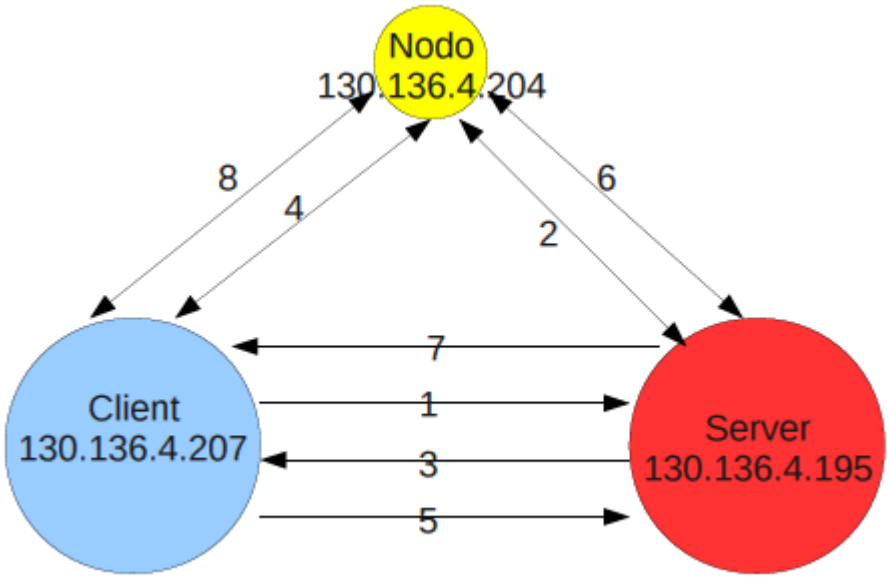

Illustrazione 15: Richiesta di servizio

A questo punto il dispositivo risulta essere autenticato al sistema, può quindi iniziare la fase che prevede la richiesta di servizio e l'attivazione della macchina virtuale nel nodo più "vicino" al

dispositivo. In questa fase del protocollo sono coinvolti il server, il dispositivo e tutti i nodi che fanno parte della nostra architettura. L'illustrazione 15 mostra uno schema semplificato delle comunicazioni e degli agenti coinvolti. Come nel caso precedente abbiamo numerato il flusso delle informazioni in ordine temporale:

1. Il client invia al server una richiesta di macchine virtuali.
2. Il server "sonda" il carico di lavoro dei vari nodi in termini di carico di CPU.
3. Il server invia al client una lista di nodi che può accettare richieste di macchine virtuali. In particolare viene inviata una lista lunga N / 2 dove N è il numero totale di nodi sondati dal Server.
4. Il client invia pacchetti ICMP ai nodi della lista e ne valuta la latenza.
5. Il client invia al server l'indirizzo del nodo con latenza minore.
6. Il server attiva la macchina virtuale sul nodo scelto dal client.
7. Il server comunica al client l'indirizzo e la porta fisica su cui è attiva la macchina virtuale.
8. Il client chiude la connessione con il server e instaura una connessione tramite VNC[17] sul nodo, al fine di utilizzare il servizio.

6.2 Scelte di progettazione

Abbiamo progettato il protocollo come definito nel paragrafo precedente immaginando un utilizzo nel mondo reale. Questa situazione ha fatto emergere diversi aspetti che riguardano la sicurezza informatica, delle reti e lo spreco delle risorse hardware. In particolare il punto 5 della fase di autenticazione garantisce l'autenticità del client. Soltanto un client che conosce la chiave privata (generata nella fase di registrazione) può rispondere alla "sfida" ricevuta al punto 6. Inoltre il punto 8 garantisce l'uso del sistema da parte di un dispositivo che sfrutta a pieno il protocollo ABPS. Approfondiremo questi aspetti nei capitoli successivi.

17 VNC (Virtual Network Computing) sono software di controllo remoto per il controllo a distanza

CAPITOLO 7

7. IMPLEMENTAZIONE

Durante la fase implementativa è stato necessario riadattare e configurare alcune soluzioni in base alle caratteristiche della nostra piattaforma. La prima fase dell'implementazione ha visto la riconfigurazione e la riscrittura di alcune parti di OpenNebula in base alle nostre esigenze.

L'interfaccia del gestore delle macchine presenta queste opzioni:

- onevm list: questo comando genera un output con l'elenco delle macchine virtuali disponibili, macchine attive e nodi della rete. Il comando viene utilizzato dall'amministratore di sistema che controlla lo stato del servizio. Esempio di output generato nell'illustrazione 16.
- onevm deploy X Y PASSWORD: il seguente comando esegue una macchina virtuale X su un nodo Y, protetta da una PASSWORD (X e Y sono i rispettivi ID). Il comando viene lanciato dal server nel momento in cui il client completa la fase di autenticazione.
- onevm remove X Y: rimuove dal nodo Y la macchina virtuale accessibile in remoto dalla porta Y. Il comando viene utilizzato dal server al fine di liberare le risorse non più utilizzate.

L'unico comando che differisce dall'interfaccia implementata su OpenNebula risulta essere “deploy”. È stato necessario aggiungere un parametro “PASSWORD” in quanto si ha la necessità di proteggere le macchine virtuali a run-time.

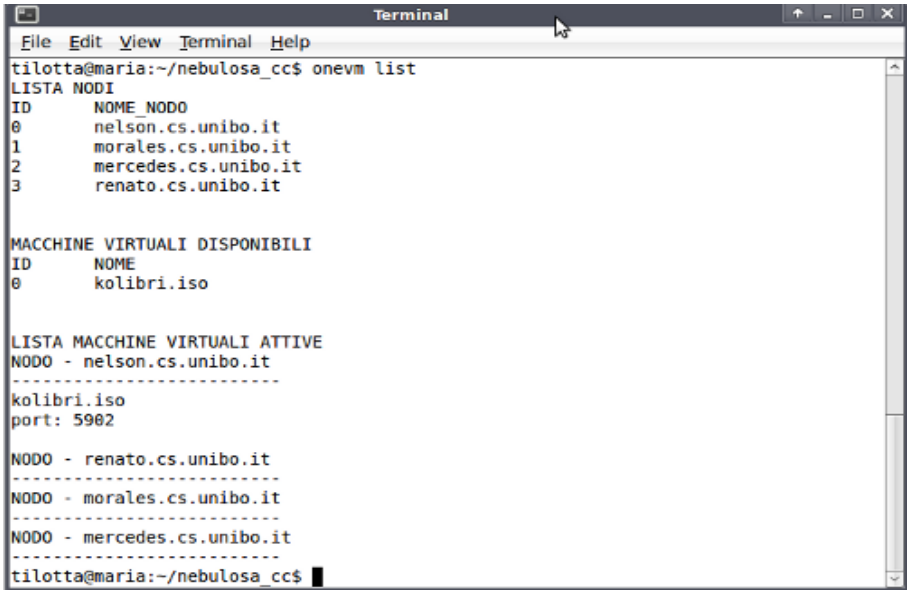

Illustrazione 16: Output generato dal gestore delle macchine virtuali

Per ogni macchina virtuale eseguita su un nodo vengono riservate due porte di rete sulla macchina fisica. Una porta è riservata ad una eventuale applicazione che vuole riceve dati (VOIP, SSH, ecc..), l'altra è riservata al fine di rendere la virtual machine accessibile dall'esterno.

7.1 Il server centrale

Il server centrale è il componente che si occupa di gestire le registrazioni, autenticazioni e monitoraggio del carico di lavoro a livello di CPU (utilizzo) su tutto l'hardware disponibile (i nodi). Il carico di lavoro sui vari nodi viene estrapolato tramite il comando:

ssh NODO ps aux | awk {'sum+=$3;print sum'} | tail -n 1

Le informazioni sul carico di lavoro vengono successivamente ordinate e inviate al client. Il server interagisce con i client solo durante la fase di registrazione, autenticazione e con i nodi dell'infrastruttura tramite l'interfaccia esposta del gestore delle macchine virtuali.

Al fine di liberare le risorse hardware, sprecate dalle macchine virtuali il server "sonda" le comunicazioni fra il client e il nodo. In particolare controlla che la comunicazione non abbia stato ESTABLISHED. Se

tale controllo risultasse negativo il server chiuderebbe la macchina virtuale liberando le risorse. Il monitoraggio viene effettuato direttamente sul nodo che ospita la macchina virtuale, basandosi soltanto sulla porta di comunicazione e non sull'IP:

ssh NODO netstat -nat | grep ESTABLISHED | awk {'print $4'} | tr ':' ' ' | awk {'print $2'} | grep PORTA

Questa soluzione garantisce la continuità del servizio anche nel momento in cui il dispositivo decidesse di utilizzare più di un indirizzo IP o interfaccia di rete.

7.1.1 Protezioni delle macchine virtuali

Allo stato attuale tutti i client / server che implementano il protocollo RFB supportano le protezioni della macchina virtuale tramite un meccanismo di login tramite password. Per questo motivo è stato necessario implementare sul server un meccanismo di protezione in maniera tale da sfruttare l'unico meccanismo di protezione esistente. Nel momento in cui il server lancia una macchina virtuale su uno dei nodi, attiva l'unico meccanismo di protezione della macchina virtuale. Il server lancerà il seguente comando:

ssh NODO "nohup echo change vnc password 'STRINGA' |
COMANDO PER LANCIARE LA MACCHINA VIRTUALE

Eventuali attaccanti che decidono di intercettare i desktop virtuali sono “costretti” a conoscere la password di protezione. Questo meccanismo garantisce la sicurezza sugli accessi alle macchine virtuali. Un eventuale attaccante avrebbe difficoltà ad eseguire un bruteforce, sulle porte in ascolto nei vari nodi, in quanto protette da password.

7.2 Il client

Il client risulta essere l'entità più semplice. La sua funzione principale è quella di ottenere accesso ad una virtual machine e visualizzarne il contenuto, direttamente sul dispositivo, tramite un'interfaccia grafica. Dopo la fase di autenticazione il client provvede ad instaurare una

comunicazione VNC direttamente sul nodo che ospita la macchina virtuale:

vncviewer NODO:PORTA -passwd FILEPASSWORD -AutoSelect=0

È bene sottolineare che il client in questa fase accede alla macchina virtuale tramite una password. La password utilizzata dal client è equivalente alla “challenge” precedentemente inviata dal server.

CAPITOLO 8

8. TEST EFFETTUATI E MIGLIORAMENTI

È stato simulato il comportamento del sistema nel caso in cui un client faccia richieste di virtual machine in maniera continuativa. A tal proposito è stato scritto uno script, in grado di lanciare svariate istanze dell'applicazione installata sul dispositivo. Le istanze lanciate virtualizzano KolibriOS[19], un sistema operativo di dimensioni ridotte, in grado di avviarsi in pochi secondi. Lo scopo dell'esperimento è quello di visualizzare la distribuzione delle macchine virtuali sui nodi dell'infrastruttura simulando una situazione reale. Al momento dell'esecuzione tutti i nodi presentavano un dispendio di CPU pari al 0%.

Lancio di 15 istanze simultanee:

NOME NODO	ISTANZE SUL NODO
calaf	6,00%
altoum	6,00%
bess	42,00%
clara	46,00%
renato	0,00%

Lancio di 30 istanze simultanee:

NOME NODO	ISTANZE SUL NODO
calaf	26,00%
altoum	25,00%
bess	13,00%
clara	26,00%
renato	10,00%

Lancio di 60 istanze simultanee:

NOME NODO	ISTANZE SUL NODO
calaf	20,00%
altoum	21,00%
bess	20,00%
clara	26,00%
renato	13,00%

Dopo diverse esecuzioni del test abbiamo notato una distribuzione equa delle macchine virtuali sui nodi facenti parte dell'infrastruttura. In particolare maggiore è il numero di istanze lanciate, più il carico di lavoro risulta essere distribuito sui nodi. Questa situazione dovrebbe garantire una corretta efficienza in termini di prestazioni su tutta la piattaforma, oltre che una maggiore fluidità nel ricevere le informazioni da parte del client.

8.1 Test per il supporto multihoming

Ogni client può accedere alle macchine virtuali tramite più interfacce di rete o indirizzi IP. Al fine di testare questo comportamento abbiamo scritto uno script che simula l'utilizzo di più interfacce di rete durante la connessione alla macchina virtuale.

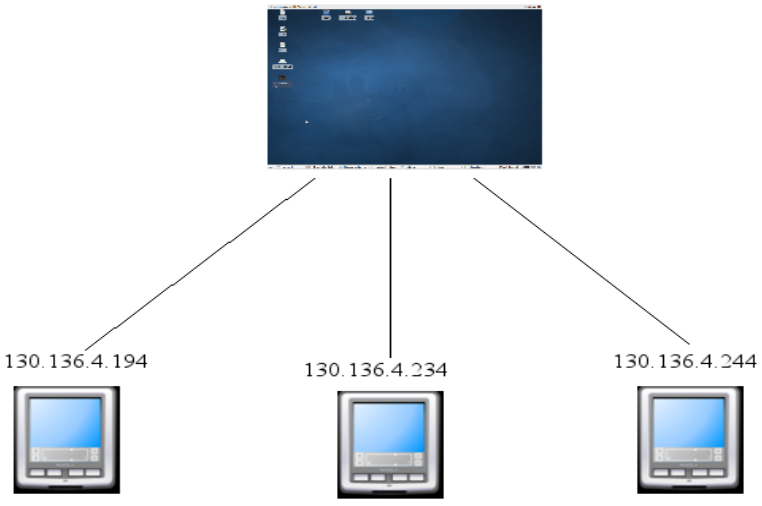

Illustrazione 17: Simulazione di un ambiente multihoming

È possibile visualizzare un esempio di ambiente simulato

nell'illustrazione 17. Tramite questa simulazione abbiamo appurato che il client può instaurare una connessione alla macchina virtuale utilizzando più interfacce contemporaneamente e differenti indirizzi IP.

8.2 Miglioramento dei meccanismi di misurazione

Se pur i test dimostrano il buon bilanciamento di carico è necessario migliorare il sistema di rilevamento del carico di CPU sui nodi. Allo stato attuale il meccanismo di rilevamento del carico di CPU viene calcolato dal server tenendo conto dello stato della CPU al momento dell'accesso. Questo comporta una scarsa qualità della stima del carico del lavoro sui nodi, in quanto un processo potrebbe essere momentaneamente in “stato di stop” proprio quando il server sonda il carico di CPU sul nodo. Al fine di affinare la misurazione del carico di lavoro bisognerebbe considerare la misurazione del carico di CPU basandoci su una stima media. Questo migliorerebbe la misurazione e quindi la distribuzione delle macchine virtuali sui nodi a disposizione.

8.3 Miglioramento della sicurezza al server VNC

L'attuale server VNC utilizzato, TightVNC Server, limita la password di protezione a 8 caratteri. Questo potrebbe essere un rischio nel caso in cui un attaccante decida di effettuare un attacco di tipo brute-force sulla password di protezione. Una soluzione a questo problema potrebbe essere implementare un meccanismo di protezione, in maniera tale da consentire una lunghezza arbitraria della password.

CAPITOLO 9

9 CONCLUSIONI SVILUPPI FUTURI

La mancanza di strumentazioni adeguate ci ha portato a testare l'infrastruttura su macchine (condivise) orientate all'ambiente desktop. Tali macchine possono subire un calo delle prestazioni in quanto utilizzate per garantire diversi servizi (mail, ssh, desktop, accessi remoti). I test effettuati confermano il buon bilanciamento del carico di lavoro sui nodi dell'infrastruttura. Tuttavia siamo convinti che l'utilizzo di strumenti idonei quali server dedicati orientati alla virtualizzazione e dispositivi mobili su cui installare il client, possa migliorare di molto l'esattezza dei risultati e le prestazioni della piattaforma. Ci aspettiamo successive implementazioni della piattaforma che vedono il client direttamente integrato nel sistema operativo in modo tale che possa attivarsi direttamente al boot. In particolare ci aspettiamo l'integrazione del client su dispositivi mobili con sistema operativo Android[20] in ambiente VoIP funzionante tramite il modello ABPS. Inoltre al fine di alleggerire le operazioni di configurazione e rendere la piattaforma gestibile da interfaccia web, sarebbe ideale configurare l'architettura implementata su servizi reali di cloud computing come ad esempio il servizio offerto da Amazon EC2, di cui abbiamo parlato nei primi capitoli. Questo permetterebbe la distribuzione a livello mondiale di tutti i nodi facenti parte dell'architettura. Inoltre la gestione dei nodi verrebbe nettamente migliorata. Potrebbe essere allettante integrare la piattaforma implementata al servizio EC2 in maniera tale da sfruttare le potenzialità del servizio. In particolare ogni istanza facente parte della piattaforma EC2 potrebbe rappresentare ogni singolo componente della piattaforma implementata. Con questo meccanismo di macchine virtuali nidificate si potrebbe avere maggiore controllo della piattaforma a un livello superiore e disporre di un parco macchine di dimensioni elastiche.

Bibliografia

1: V. Ghini, G. Lodi, F. Panzieri, Always Best Packet Switching: the MobileVoIP Case Study, Journal of communications, vol. 4, no. 92009 Ottobre 2009
2: Vincenzo Tilotta, Source code of nebulosa-cc, http://code.google.com/p/nebulosa-cc
3: George Reese, Cloud Application Architectures, 2010
4: NIST - Information Tecnology Laboratory, NIST Cloud Computing Program, http://www.nist.gov/itl/cloud, 2010
5: Amazon, Amazon Elastic Compute Cloud (Amazon EC2), http://aws.amazon.com/ec2/
6: Amazon, Amazon EC2 - Getting Started Guide, http://docs.amazonwebservices.com/AmazonEC2/gsg/2006-06-26/, 2006
7: BMC software, Eucalyptus System, HP, IBM, Intel, Red Hat, SUSE, Open Virtualization Alliance, http://www.openvirtualizationalliance.org, 2011
8: Red Hat, KVM, http://www.linux-kvm.org
9: OpenNebula Community, OpenNebula, http://opennebula.org
10: Network Working Group of the IETF, The Secure Shell (SSH) Authentication Protocol, http://www.ietf.org/rfc/rfc4252.txt, January 2006
11: Tristan Richardson, RealVNC Ltd, The RFB Protocol, http://www.realvnc.com/docs/rfbproto.pdf, 2010
12: Network Working Group, Internet Protocol, Version 6 (IPv6) Specification, http://tools.ietf.org/html/rfc2460, 1998
13: IETF - Network Working Group, The MD5 Message-Digest Algorithm, http://tools.ietf.org/html/rfc1321, April 1992
14: IETF - Network Working Group, HMAC: Keyed-Hashing for Message Authentication, http://tools.ietf.org/html/rfc2104, February 1997
15: Larry L. Peterson, Bruce S. Davie, Computer Network: A System Approach, 2004
16: IETF - Network Working Group, The Transport Layer Security (TLS) Protocol - Version 1.2, http://tools.ietf.org/html/rfc5246, August 2008
17: IETF - Network Working Group, TELNET PROTOCOL SPECIFICATION, http://www.ietf.org/rfc/rfc854.txt, 1983
18: The OpenSSL Project, OpenSSL, http://www.openssl.org
19: Community, KolibriOS, http://kolibrios.org
20: Google Inc., Open Handset Alliance, Android, http://www.android.com