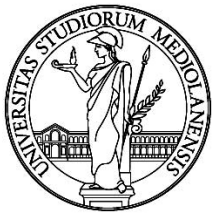

UNIVERSITÀ DEGLI STUDI DI MILANO

DIPARTIMENTO DI INFORMATICA

Corso di Laurea Magistrale in Sicurezza Informatica

Valutazione automatica e continua della compliance in ambienti cloud: Il caso di studio FedRAMP

RELATORE

Prof. Claudio Agostino Ardagna

CORRELATORE

Dott. Marco Anisetti

SECONDO CORRELATORE

Prof. Ernesto Damiani

TESI DI LAUREA DI

Patrizio Tufarolo

Matr. 875041

Anno Accademico 2016/2017

Ai miei genitori e a mio fratello

Acknowledgments

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

.

.

Capitolo 2

Security assurance: lo stato dell'arte e la sfida

2.1 Introduzione

In questo capitolo si approfondirà lo stato dell'arte in materia di **security assurance** e **controllo della compliance**, ovvero la verifica della conformità di un'infrastruttura informatica tradizionale, ibrida o cloud, rispetto a una politica, che può essere sviluppata internamente oppure derivata da un più complesso apparato normativo o da uno standard. In particolare verranno trattate le problematiche di sicurezza introdotte dall'adozione di un approccio *cloud*, all'interno dei processi *IT* di un'organizzazione strutturata sulla base di un'infrastruttura informatica tradizionale.

Spesso si fa coincidere il concetto di cloud computing con quello di outsourcing, di fatto presupponendo che l'adozione di tecnologie cloud corrisponda all'attitudine di concedere a terzi gli oneri di gestione di una parte dell'infrastruttura informatica. La definizione di *cloud computing* a cui si fa riferimento in questo elaborato di tesi è invece quella del NIST[1] nel documento *SP-800-145* nel quale il cloud è presentato come un insieme di tecnologie aventi come obiettivo l'erogazione di servizi e risorse in modalità *on-demand* da un pool condiviso. La condizione di outsourcing, quindi, acquisisce una connotazione non necessaria all'adozione di un servizio cloud.

2.2 Sicurezza nel cloud computing

Lo scopo finale dell'utilizzo di tecnologie *cloud* consiste nella possibilità per un'organizzazione di usufruire di un modello scalabile, elastico, standard, misurabile e orchestrabile al fine di poter garantire continuità di servizio e prestazioni elevate, demandando la gestione dei processi sistemistici a piattaforme centralizzate e intelligenti. A tal proposito il NIST[1] identifica tre modelli di servizio:

IaaS , Infrastructure as-a-Service, nel quale è l'asset erogato è l'infrastruttura informatica, in termini di potenza di calcolo mediante sistemi di virtualizza-

[1] National Institute of Standards and Technology

zione, risorse di rete e storage. Essendo il modello più di difficile gestione, è spesso amministrato tramite un orchestratore.

PaaS , Platform as-a-Service, tramite il quale si fornisce all'utente la possibilità di eseguire servizi personalizzati offrendo meccanismi di contenimento nell'esecuzione, scalabilità e multi-tenancy. L'utente ha un controllo parziale sull'esecuzione del servizio: solitamente egli può interagire in modo limitato con il kernel.

SaaS , Software-as-a-Service, che permette all'utente di usufruire delle funzionalità di un singolo applicativo, riducendo al minimo l'effort computazionale sulla macchina dell'utente stesso. Tipicamente in questa categoria ricadono le applicazioni web, alcune applicazioni mobile e alcuni software per PC.

La parola chiave è quindi **"automazione"**. Questa, oltre a garantire una solidità del modello di distribuzione di un servizio grazie a schemi dichiarativi, apporta notevoli vantaggi anche dal punto di vista della sicurezza, facilitando la gestione degli aspetti di confidenzialità, integrità e disponibilità.
Il paradigma *as-a-service* ha infatti consentito la costituzione di una *baseline* robusta garantita dalla centralizzazione delle funzionalità di security. Queste, essendo erogate come risorse *cloud*, sono interamente gestite dal *cloud service provider*, pubblico o privato, che può demandarne la gestione parziale all'utente mediante meccanismi di orchestrazione, interfacce grafiche ed API.
Se a primo impatto può apparire come un enorme vantaggio, di fatto ciò introduce un *single point of failure*, determinando livelli di rischio aggiuntivi rispetto alle infrastrutture tradizionali. Si pensi, ad esempio, alle funzionalità di *firewalling* offerte generalmente con la denominazione di *security groups* o Firewall as-a-Service (FWaaS): un'implementazione non idonea dal punto di vista funzionale nel substrato infrastrutturale del fornitore di servizi, potrebbe determinare la mancanza di sicurezza per i servizi che ne fanno affidamento. La stessa asserzione è valida per molte altre funzionalità comunemente offerte dal provider: cifratura dei volumi di storage, crittografia e controllo degli accessi nei servizi di block-storage e così via.

Ulteriori riflessioni possono essere fatte anche per quanto riguarda l'aspetto di integrità del dato: se da una parte il cloud service provider implementa già meccanismi di basso livello per la persistenza dello storage, ridondanza, sistemi di backup automatici, dall'altra non si ha la chiara evidenza di come questi aspetti siano effettivamente gestiti e di come la proprietà sia garantita.

Per quanto concerne la proprietà di disponibilità, la dicotomia va ricercata trattando i concetti di disponibilità del dato e disponibilità del servizio separatamente. Il *cloud computing* offre intrinsecamente solidità in quanto basato sui concetti di scalabilità, elasticità e ridondanza. Grazie ai meccanismi di orchestrazione tramite API è infatti possibile configurare le applicazioni per l'*auto-scaling*, al fine di mantenere una qualità adeguata nell'erogazione del servizio al crescere degli utenti. Ciò, dal punto di vista della sicurezza, ha portato a notevoli benefici per quanto riguarda la mitigazione di attacchi DoS[2], garantendo la continuità di servizio riducendo i costi. Tuttavia esistono dei prerequisiti per garantire la

[2]Denial of Service

disponibilità: innanzitutto il *cloud service provider* deve assicurare la ridondanza dei dati e della rete, contemplando l'ipotesi di distribuire le risorse su più località geografiche, con l'obiettivo sia di prevenire guasti localizzati che di erogare la risorsa dalla località più vicina rispetto all'utente.

Nel momento in cui funzionalità comunemente demandate ad hardware specifico vengono implementano in software, si determinano sia benefici che svantaggi che devono sia essere contemplati in fase di valutazione del rischio che trattati nei contratti di service level agreement. Una compromissione dell'interfaccia di gestione della piattaforma cloud, sia che si tratti di una dashboard sia che si tratti di un'interfaccia API, può portare a un'interruzione di servizio.

Gli standard di sicurezza classici, così come l'assetto normativo e i contratti di *service level agreement*, necessitano di essere adeguati per supportare l'integrazione di tecnologie cloud all'interno degli stack tradizionali, tenendo conto delle problematiche di *shared responsability* presentate.

Il NIST [1] riconosce quattro diversi modelli di deployment:

- **Public Cloud**: modello in cui le risorse sono fornite per un utilizzo pubblico. È tipicamente erogato in outsourcing tramite la rete internet. L'hardware è in mano a un unico provider che eroga servizi in *outsourcing* e ne dispone le metriche e la tariffazione.
- **Private Cloud**: cloud dedicata a un'azienda o organizzazione, sfruttata per erogare servizi appartenenti al provider. L'hardware è generalmente nel datacenter dell'organizzazione.
- **Hybrid Cloud**: approccio ibrido dato dalla composizione di public cloud e private cloud, o di public cloud e infrastrutture tradizionali. Le infrastrutture coinvolte rimangono distinte e sono legate tra loro da un'unica tecnologia (standard o proprietaria) che facilita la migrazione e la portabilità delle risorse.
- **Community Cloud**: modello che fornisce una cloud per uso esclusivo di una comunità di utenti appartenenti ad organizzazioni con obiettivi funzionali comuni. Può essere di proprietà di una o più organizzazioni della community, o di terze parti.

Per ognuno di questi modelli è possibile esplicitare dei requisiti da soddisfare al fine di colmare il rapporto di sfiducia proprio di questo settore[2].

2.3 Valutazione del rischio: vulnerabilità, minacce e attacchi

In letteratura sono stati proposti molti lavori sulla valutazione del rischio su infrastrutture cloud. Nei paragrafi a seguire verranno discussi alcuni di questi approcci, sulla base della metodologia utilizzata da Ardagna et Al.[2]. Le vulnerabilità sono qui categorizzate in tre macro aree, in base alla superficie di attacco:

1. **Livello applicativo**: quando l'attacco è condotto da un qualsiasi attore nei confronti di una piattaforma SaaS

2. **Tenant su tenant**: quando l'attacco è condotto da attori appartenenti a un tenant nei confronti di un altro tenant

3. **Provider su tenant** e **Tenant su provider**: quando l'attacco è condotto dal provider nei confronti di un tenant (tipicamente malevolo) oppure da un tenant nei confronti del provider

2.3.1 Livello applicativo

Si tratta di vulnerabilità tradizionali che da anni tengono sotto scacco il panorama *web services*: si va da attacchi protocollari sulla comunicazione tra servizi fino alla compromissione di applicativi software specifici. Il target dell'attacco sono le piattaforme SaaS, spesso derivate dal porting di un'applicativo tradizionale sul cloud e non nativamente pensate per essere erogate online: per questo motivo sono caratterizzate da una superficie di attacco molto vasta.

Alcuni lavori significativi citati nel survey di rifermento [2] sono:

- **Gruschka and Iacono, 2009**[], nel quale è stato presentato un *replay attack*, sfruttando una vulnerabilità del meccanismo di verifica della firma digitale sull'interfaccia SOAP di *Amazon EC2*, e sono state eseguiti comandi sulle API con i privilegi di un utente legittimo

- **Bugiel et Al., 2011**[], che hanno analizzato le minacce sulla confidenzialità e la privacy estraendo con successo informazioni sensibili da immagini di macchine virtuali Amazon

2.3.2 Tenant su tenant

Le vulnerabilità *tenant su tenant* sono tipiche dei sistemi virtualizzati, quando tenant differenti condividono la stessa infrastruttura e, più specificatamente, lo stesso hardware fisico: gli attacchi possono avvenire per configurazioni erronee o vulnerabilità sull'infrastruttura di virtualizzazione. Si tratta quindi di attacchi che avvengono al livello più basso dello stack cloud[2].

Tra questi attacchi

-

2.3.3 Provider su tenant, tenant su provider

Le vulnerabilità di questo tipo si verificano ogni qual volta un utente o un'organizzazione sposta le proprie risorse su un'infrastruttura cloud non fidata - nella quale il provider è malevolo oppure semplicemente curioso - oppure nel caso in cui l'utente inizia ad usare un servizio cloud con l'obiettivo di attaccare il provider (ad esempio creando botnet per lanciare attacchi denial of service, attaccando le API di orchestrazione e così via) [2].

Le tipologie di attacchi che sfruttano queste vulnerabilità, sono generalmente rivolte al livello IaaS[2], ma non è esclusa la possibilità di attacchi a livello PaaS e SaaS.

Il survey si sofferma sul lavoro Liu2010 , il quale illustra una attacco DDoS basato sulla saturazione della banda della rete virtuale: la virtualizzazione dello stack di rete a livello software (*software-defined network*) richiede, oltre a risorse di rete, anche un'elevata capacità di calcolo.

Le vulnerabilità provider su tenant sono invece trattate da Rocha e Correira [2011] che propongono una panoramica dei possibili attacchi alla confidenzialità - che possono essere condotti anche dal fornitore di servizi - discutendone le contromisure, e da Blekeirz et al [2013] che si concentrano sulla problematica di proteggere i clienti da attacchi condotti da provider esterni, fornendo un'architettura *Cryptography as-a-Service client-driven*.

La problematica di confidenzialità nella casistica *provider-on-tenant* è anche l'oggetto di De Capitandi Vimercati et. al in cui è descritta una tecnica per preservare la confidenzialità del dato proteggendo con l'allocazione dinamica dello stesso ad ogni accesso su tre nodi, affrontando anche i problemi di collusione tra gli eventuali service provider coinvolti. De Capitani di Vimercati et al. [2013] affronta ancora una volta il provider *onesto ma curioso* con una soluzione per l'integrità dei risultati delle query di join, che discute la casistica di un server di storage di terze parti e di fornitori di potenza di calcolo esterni e malevoli, che producono i risultati del join per basi di dati ospitate esternamente.

2.4 Tecniche di sicurezza per la cloud

Data l'etereogeneità delle problematiche e degli approcci adottati negli articoli citati, è possibile affermare che garantire proprietà di sicurezza in ambienti cloud è molto impegnativo: questi lavori presentano solamente soluzioni parziali al problema, affrontando di volta in volta problemi specifici e presentando tecniche sviluppate *ad-hoc*[2]. Saranno di seguito presentati alcuni approcci e tecniche per garantire la sicurezza su sistemi cloud.

2.4.1 Autenticazione e controllo degli accessi

I sistemi tradizionali per l'autenticazione e il controllo degli accessi si sono verificati inefficienti per la cloud, pertanto è stato necessario definire nuovi approcci. L'adozione di nuovi *pattern* di sviluppo orientati alla scalabilità - come ad esempio il pattern *micro-services*, naturale evoluzione delle architetture SOA - ha reso necessario sviluppare meccanismi di autenticazione decentralizzati e federati. Almulia and Yeun [2010] offrono una panoramica sui protocolli di autenticazione e *identity management*, analizzandone la sicurezza, l'effort implementativo e i costi.

Costituendo parte critica per la maggior parte dei sistemi, i servizi di autenticazione, gestione dell'identità e gestione delle policy di accesso sono erogati *as-a-service*.

Takabi e Joshi [2012] hanno descritto un sistema di gestione delle policy *as-a-service* (PMaaS, *Policy Management as-a-service*) che fornisce un punto di controllo centralizzato indipendente dalla locazione della risorsa. Prima di accedere a una risorsa è necessario contattare il server di autenticazione e autorizzazione centralizzato che rilascerà il *grant* dopo opportuna verifica. *Azure Active Directory*, il

porting *SaaS* di Microsoft Active Directory, provvede sia a funzionalità di autenticazione che di policy management e fornisce alcuni driver di integrazione per la maggior parte dei protocolli noti.

Tuttavia, poiché molte realtà complesse dispongono già di meccanismi di autenticazione mediante *ticket granting* isolate dalla rete Internet, sono stati ideati anche modalità di autenticazione e controllo degli accessi completamente *stateless* (ad esempio OAuth). È il caso dei *JSON Web Token*, formalizzati nella RFC 7519: l'*authentication server*, dopo aver validato la richiesta di autenticazione, restituisce un token JSON firmato che contiene l'identità dell'utente e tutti i *grant* per le autorizzazioni ad esso relative. Non esiste il concetto di sessione, la validità del token è data esclusivamente da una marca temporale e da una durata. Il token può essere utilizzato quindi per autenticare le richieste verso i vari servizi, cui spetta l'onere di verificarne la validità del contenuto e della firma, decifrabile tramite segreto condiviso con il server di autenticazione che lo ha emesso. I vantaggi di un approccio simile sono molteplici, tuttavia è impossibile revocare il token una volta emesso. Eventuali blocchi sono effettuabili tramite sistemi di *blacklisting* che riporterebbero in auge la problematica della decentralizzazione che si voleva risolvere. La prassi è quindi quella di emettere token one-time o con durata breve, al fine di minimizzare la durata di una possibile finestra temporale di attacco.

2.4.2 Crittografia e firma digitale

La crittografia è essenzialmente utilizzata per proteggere la confidenzialità dei dati, delle comunicazioni e le attività sensibili da tutti quegli avversari che mirano a disturbare l'operatività della cloud. La maggior parte della letteratura utilizza tecniche di crittografia per preservare la confidenzialità: l'obiettivo di queste metodologie è di facilitare la migrazione dei dati gestiti da sistemi tradizionali verso la cloud. Tuttavia non sono assenti tecniche focalizzate su altre proprietà di sicurezza, come l'utilizzo della firma digitale per curare gli aspetti di integrità e privacy.

Trusted Computing

Il *trusting computing* è una tecnica utilizzata per effettuare computazioni sicure, basata sull'utilizzo della crittografia asimmetrica e di un dispositivo hardware dedicato (TPM, Trusted Platform Module) tramite il quale è possibile *i)* identificare univocamente i dispositivi con un numero di serie e una chiave di cifratura implementata in hardware *ii)* cifrare informazioni con la chiave di cifratura *iii)* firmare informazioni con la chiave di cifratura. Queste funzionalità pongono le basi per una serie di utilizzi avanzati volti a preservare l'integrità e la confidenzialità di dati - sia in transito su una rete, che memorizzati su disco o sui firmware del dispositivo - codice e hardware, riducendo o annichilendo gli effetti di eventuali attacchi.

Boampeng e Washeh nel 2012 hanno proposto un modello per utilizzare il TPM al fine di garantire la correttezza dei processi di autenticazione, l'integrità e la confidenzialità sulla cloud. Portare il TPM sul cloud significa realizzarne una

versione virtuale, così come illustrato da Krautheim nel 2009, basandosi sul concetto di virtual-TPM (vTPM) già descritto da Berger et al. nel 2006. Il vTPM è un componente software che implementa le stesse funzionalità del TPM hardware, garantendo la multi-tenancy mediante istanze multiple e multiplexing. I vantaggi dell'utilizzo di una tecnologia di *trusted computing* nel contesto cloud sono molteplici, come la possibilità per l'utente di fare enforcement di politiche di privacy togliendo la possibilità al cloud service provider di modificarle, fornendo una soluzione parziale problematiche di *shared responsability* discusse. Come illustrato da Velten and Stumpf [2013] e più recentemente da Szefer e Lee, il TPM può essere utilizzato per garantire confidenzialità e integrità a tutti i livelli dello stack, prevenendo tampering da parte del fornitore di servizi e attacchi da parte di altri tenant o da malware.

2.5 Approcci per assurance, testing, monitoraggio e compliance

I progressi nella ricerca sulla sicurezza della cloud hanno portato la necessità di avere tecniche di *security assurance* per aumentare la confidenza degli utenti nei confronti del provider[3].

Per *assurance* si intende la modalità per ottenere, con un certo livello di precisione, la consapevolezza che l'infrastruttura e/o le applicazioni manterranno nel tempo una o più proprietà di sicurezza, e la loro operatività non sarà compromessa indipendentemente da malfunzionamenti o attacchi[4]. In accordo con Ardagna et Al.[2], è possibile affermare che quello di *assurance* è un concetto più esteso della mera nozione di *sicurezza informatica*, comunemente definita come *la protezione delle informazioni e dei sistemi informativi da accessi, utilizzi disclosure, interruzioni del funzionamento, modifiche e distruzioni non autorizzate*. Nella cloud è molto facile avere livelli di sicurezza elevati con livelli di assurance scarsi poiché le funzionalità di sicurezza realmente implementate sono difficilmente percepite.

L'obiettivo di questo lavoro di tesi è quello di fornire un framework per la security assurance i) insistendo sulla valutazione continuativa dello stato di sicurezza sulla cloud ii) offrendo un framework cloud-based per la security assurance insistendo su

- testing di proprietà non funzionali
- monitoraggio continuativo della sicurezza del sistema
- conformità del sistema a politiche di sicurezza, siano esse definite internamente ad un'organizzazione, siano esse provenienti da uno standard di settore
- ottemperare alle esigenze di transparency degli utenti della cloud, offrendo una dashboard panoramica sullo stato della cloud del provider

Testing di proprietà non funzionali Il *testing* è definito come la fase del ciclo di vita del software composta da tutte le attività, statiche o dinamiche, atte a deter-

minare che questo soddisfi i requisiti specificati e che sia conforme all'obiettivo proposto, nonché per rilevare eventuali difetti.

Nel contesto *cloud* possiamo riconoscere due tipologie di soluzioni di testing: quelle specifiche per il collaudo di infrastrutture cloud e quelle generiche per il testing del software, applicabili anche a servizi cloud.

Il lavoro di tesi si focalizzerà maggiormente sulla prima categoria insistendo sulla validazione delle proprietà a tutti i livelli dello stack (in accordo con Riungu et. Al [2010]); nonostante ciò il framework proposto può essere adattato ad entrambe le tipologie.

Monitoraggio continuativo della sicurezza del sistema La natura stessa dei sistemi cloud complica notevolmente l'analisi delle informazioni relative allo stato dei servizi: a causa dell'elevata complessità dei software impiegati nell'orchestrazione e nell'erogazione delle risorse è spesso difficile rilevare cambiamenti nello stato del sistema, il cui back-end è continuamente tempestato di eventi. È quindi necessario introdurre una componente di monitoraggio e collezionamento di eventi.

Per valutare aspetti non funzionali come la sicurezza, è poi necessario che questi eventi vengano contestualizzati: sono necessarie pertanto analitiche *stateful*, effettuabili anche tramite strumenti più complessi o provenienti dal mondo big-data.

Proprio per facilitare scenari di *software integration* il framework proposto nei prossimi capitoli è stato strutturato esasperando la modularità, ed è stato basato principalmente su tecnologie *open-source*.

Come per il testing, anche per il monitoraggio è possibile individuare sia soluzioni generiche sia soluzioni specifiche per il mondo *cloud*. Software come *Nagios* (piattaforma di monitoraggio distribuita general purpose) e *Ganglia* (soluzione per il monitoraggio delle performance dei cluster in ambito grid computing) rientrano nella prima categoria, ma vantano livelli di espandibilità tali da poter essere adeguati ai sistemi di collezionamento delle metriche dei maggiori software cloud. Prodotti come *Sensu*, *Sysdig*, *Weave* contengono strumenti specifici per la cloud.

Per quanto riguarda gli aspetti di sicurezza, la disponibilità di potenza computazionale on-demand, ha garantito la possibilità di effettuare il deploy scalabile di sistemi IDS e IPS.

Modi et al. surveyed different attacks affecting availability, confidentiality, and integrity, and reviewed approaches providing IDS and IPS in the cloud. The authors focus on insider attacks, flooding attacks, user to root attacks, port scanning, attacks on hy- pervisor or VMs, and backdoor channel attacks. Then they present the evolution of IDS and IPS, and explain how IDS and IPS have been used to increase cloud security. The authors also present a useful summary of existing IDS approaches (see Table IV in Modi et al. [2013b]) discussing their advantages and drawbacks. Patel et al. [2013] investigate new issues, challenges, and requirements when intrusion detection and prevention functionalities are deployed in the cloud and introduce a survey of existing technologies, while Ficco et al. [2013] provide a survey of cloud-oriented distributed intrusion detection systems. The latter survey presents a distributed, hierarchical, and multi-layer

architecture for intrusion detection, which supports complex event correlation analysis.

Some approaches to intrusion detection and prevention in the cloud are summarized below. With respect to traditional IDS, Christodorescu et al. [2009] consider an important aspect in cloud security, namely, the security of VMs over which cloud ser- vices and functionalities are deployed. They propose an approach to increase VM in- trospection [Ardagna et al. 2014] and provide an architecture securing the customers' virtualized workloads. The approach makes no assumption on the integrity of the VMs. The paper also describes a rootkit-detection and rootkit-recovery service running out- side the VM as an application of the presented introspection approach. Lee et al. [2011] propose a multilevel intrusion detection system that checks the users' authentication information and applies different levels of security strength to them based on their degree of anomaly. The anomaly level of users is determined based on their config- uration (such as the IP coverage and vulnerable ports) and then updated regularly based on their behavior in using the cloud. Benali et al. [2010] present a distributed and privacy-preserving network intrusion detection system. Their approach is based on collaborative intrusion detection and on secure multiparty computation for privacy- enhanced evaluation of the global state of the network. Considering IPS, Stolfo et al. [2012] present fog computing, a solution to mitigate data theft attacks from insiders in the cloud. Their proposal is based on decoy technology that launches a disinformation attack when an insider attack is detected through monitoring. Yu et al. [2013b] define a resource allocation solution based on intrusion prevention servers, which permits to counteract DDoS attacks. The proposed solu- tion focuses on protecting servers that are vulnerable to DDoS attacks; to this aim, it employs different intrusion prevention servers to distinguish malicious from nor- mal traffic directed to the entity under attack. Variable attack surfaces have also been used as an attack mitigation strategy. Xing et al. [2013] present SnortFlow, an open-flow intrusion prevention system that automatically reconfigures the cloud net- working system to counteract attacks. Recently, Luo et al. [2014] proposed a federated cloud security architecture that proactively defends the cloud against cyber threats and attacks, by deploying controls at application, network, and system levels.

2.5.1 Conformità del sistema a politiche di sicurezza

The use of certification techniques to provide enough evidence that a software system holds some nonfunctional properties and behaves correctly has become widespread in the last 20 years and is also becoming important in cloud environments. Many certification solutions and schemes have been proposed in the past. A survey of certifi- cation schemes used to evaluate and certify security properties of software in general, and of security controls in particular, can be found in Damiani et al. [2009a]. How- ever, as pointed out in Anisetti et al. [2013b], "existing certification techniques are not well-suited to the service scenario," and in turn to the cloud scenario. In fact, such techniques "usually consider static and monolithic software, provide certificates in the form of human-readable statements, and consider system-wide certificates to be used at deployment and installation

time." By contrast, in a cloud environment, a certifica- tion scheme needs to accomplish the dynamic, multilevel, and hybrid nature of clouds. In addition, it must integrate with cloud-specific runtime processes, involving service

deployment, discovery, selection, and composition, and management activities, includ- ing migration, elasticity, and resource allocation. The first step toward cloud certification consists in the definition of certification so- lutions for services. Damiani et al. [2009b] study the issue of assessing and certifying SOA operation, by means of security certificates including signed test cases. Also, the US-based Software Engineering Institute (SEI) [SEI 2011] defines a certification and accreditation process for services following requirements by the US Army CIO/G-6. Kourtesis et al. [2010] use Stream X-machines to increase SOA reliability. Their so- lution manages conformance testing via the SOA registry, which evaluates functional equivalence between a service and its specifications. If equivalence is verified, a certifi- cate is awarded to the service. Furthermore, some papers (e.g., Ryu et al. [2008] and Papazoglou et al. [2011]) analyze the management of evolving services subject to dy- namic changes. This scenario, which introduces the need of continuous service redesign, has direct impact on cloud/service security certification. Changes may in fact invalidate certificates, thus requiring recertification. Anisetti et al. [2012, 2013a, 2013b] propose a security certification scheme that implements a model-based testing approach, and extends it to cope with certification of evolving services and service compositions. The proposed solution relies on a Symbolic Transition System (STS)-based service modeling to the aim of automatically generating test cases for service certification. Focusing on cloud computing, only a few preliminary solutions to the cloud certifica- tion problem have been proposed. Khan and Malluhi [2010] discuss the problem of es- tablishing trust between the cloud and its customers, and describe possible approaches to support trust in the cloud, including service certification. From a different point of view, Grobauer et al. [2011] provide an overview of current vulnerabilities affecting the cloud at different levels, and identify certification as a preferred approach for vul- nerability management. Spanoudakis et al. [2012] discuss the need of providing novel models for cloud service certification and present a hybrid, incremental, and multilayer approach to cloud certification. Sunyaev and Schneider [2013] present an overview of the possible benefits a certification solution for cloud services could give to all cloud ac- tors, addressing the lack of transparency, trust, and acceptance. Bertholon et al. [2011] present CERTICLOUD, a solution that builds on a trusted platform module to protect and verify the integrity of IaaS providers. CERTICLOUD is based on two protocols: (1) TPM-based Certification of a Remote Resource (TCRR) verifies the integrity of phys- ical resources, and (2) VerifyMyVM verifies the integrity of the environment of the user when deployed in the cloud. Muñoz and Maña [2013] introduce a solution to security certification in the cloud that combines software and hardware-based certification. The proposed approach is based on trusted computing technology and aims to bridge the gap between cloud certification and trusted computing. Krotsiani et al. [2013] propose an approach to the incremental certification of cloud services. The proposed approach targets all layers of the cloud stack and is based on continuous monitoring. Cimato et al. [2013] introduce a conceptual framework supporting the specification of basic, hybrid, and

incremental models for the certification of cloud-based services. In partic- ular, they define a metamodel supporting the management of the whole certification process: from security property definition, to evidence generation and certificate lifecy- cle management

2.5.2 Cloud auditing e Compliance

Another important aspect of cloud assurance is the capability of observing the cloud behavior and evaluating its compliance with customer policies and law regulations. In other words this goal can be expressed with the slogan "making the cloud auditable." Audit solutions can increase the transparency of the cloud, thus increasing the level of trustworthiness between the cloud itself and its tenants. Specifically, Haeberlen [2010] and Pearson [2011] respectively claim the need of an accountable cloud, which helps to increase users' trust and supports both providers and customers in the identification of responsibilities in case of disputes and problems. Later, Rasheed [2013] provided an overview of the state of the art in cloud auditing, focusing on user requirements, techniques for security auditing, and capabilities of cloud service providers to address audit requirements. Wang et al. [2010, 2013b] use a homomorphic authenticator with random masking to provide an auditing system for the cloud with privacy in mind. Mei et al. [2013] present TTP-ACE, a trusted third-party–based auditing system for the cloud. TTP-ACE is aimed at increasing accountability of cloud service providers and protecting the cloud users. A number of public auditing solutions that do not rely on a TTP have become available. In a seminal paper, Wang et al. [2011] propose a system supporting integrity verification and addressing the dynamic evolution of data files. Then, Wang et al. [2012] introduce an integrity auditing mechanism that relies on distributed erasure-coded data. A priori encoding of data permits users to audit a cloud storage at low computation and communication costs. After that, Wang et al. [2013] designed a complete public auditing mechanism for the cloud. Their approach guarantees shared data integrity as well as efficient revocation of users using proxy resignatures. Public verifiers are then capable of auditing data integrity with no need to retrieve the entire data from the cloud. Recently, Wang et al. [2014] proposed an approach to privacy-preserving public auditing, which supports integrity verification of shared data in the cloud. The proposed solution is based on a ring signature that protects the identity of the signers from public auditors, and allows integrity verification without requiring the disclosure of the entire file. Birnbaum et al. [2013] introduce a new behavioral modeling scheme to audit VM behaviors and detect suspicious processes. The proposed cloud security auditing solution has been evaluated on a private cloud computing platform. Rajkumar et al. [2013] describe an efficient auditing approach based on raptor codes that provides data integrity in the cloud. The same approach also supports functionalities for recovering data in case of failures. Shetty [2013] considers the analysis of network traffic as a fundamental aspect of cloud auditing to the aim of verifying security of data exchanged between a cloud provider and users. The proposed approach is based on IP geolocation of network devices, monitoring data security in the network, and analysis of large cloud auditing logs. Yang and Jia [2013] define an auditing framework for cloud

storage, which ensures that data have been saved following agreements with data owners. They also provide a secure and privacy-preserving auditing protocol, with no trusted parties, which supports dynamic operations and batch auditing. Ni et al. [2014] show that the auditing protocol in Yang and Jia [2013] is insecure against active adversaries in the cloud, and that adversaries can modify cloud data without being detected. They also propose a solution to solve the problem, preserving all properties of the original protocol. Doelitzscher et al. [2012, 2013] propose Security Audit as a Service (SAaaS), a cloud audit and incident detection system. Their goal is to present a solution that addresses the limitations of traditional audit and intrusion detection systems when moved to the cloud and reacts to changes in the cloud infrastructure. SAaaS is aimed at increasing transparency of cloud by giving customers access to data about security incidents. In a later development, Doelitzscher et al. [2013] presented a cloud audit policy language for the SAaaS architecture, which aims to enrich SAaaS toward the definition of a complete audit system. The presented approach mostly targets IaaS level, is focused on security monitoring, and is aimed at presenting auditing data through a standard interface. Zhu et al. [2013] propose a dynamic audit service relying on an index-hash table that supports provable updates to outsourced data. Dynamic auditing guarantees timely anomaly detection. Zawoad et al. [2013] present Secure-Logging-as-a-Service (SecLaaS), a logging system that provides VM logs to forensic investigators preserving privacy and confidentiality of cloud users. Also, SecLaaS preserves log integrity from dishonest investigators or cloud providers. Recently, Cloud Security Alliance (CSA) started an effort called CloudAudit [CSA 2014], which focuses on the provisioning of a common interface and namespace supporting enterprises in the management of their internal audit processes.

2.5.3 Esigenze di transparency

The concept of transparency, that is, higher access to low-level (back-end) data pro- duced by the cloud infrastructure and to evidence collected on the security of cloud data and applications, has been recognized as the basis for an effective approach to cloud assurance [Ardagna et al. 2014; Spanoudakis et al. 2012]. Lack of transparency in fact makes the cloud and its security issues not clear to end users. Chauhan et al. [2013] claim that security threats "require cloud customer to look for more transparency and controls" and that "SLAs and contracts do not provide technical and measur- able method to find the security control status of cloud hosted application/data." They present an approach supporting the measurement of the security status of a system. In particular, they propose a Security Measurement System (SMS) that interacts with cloud-hosted applications to retrieve metric information. Jenkins [2013] claims that there are three fundamental aspects to consider for securing businesses moving to the cloud. First, there is the need of a solution to risk assessment and management, evaluating the impact a movement to the cloud would have on the business. The sec- ond aspect is transparency, meaning that the cloud customers must be well aware of cloud provider practices. Third, policy and compliance become a must. Cloud providers, following the transparency requirement, should not only show their complian-

ce to standards/regulations and the supported policies, but also explain how they achieve and maintain their compliance levels under the "comply-or-explain" principle [MacNeil and Li 2006]. In Knode [2009], the concept of transparency is introduced as a way to doc- ument, evaluate, and observe "technical controls (e.g., auditing, access control, system configuration, encryption), management controls (e.g., vulnerability assessments, risk assessments, system and service acquisition), and operational controls (e.g., configura- tion management, awareness and training, change management)". Also, transparency aims to provide a trusted cloud service, which evaluates cloud providers and their trustworthiness. In this context, CloudTrust Protocol (CTP) is the mechanism under the user control allowing it to ask and retrieve information about the cloud provider infrastructure. In addition, according to Ardagna et al. [2014], transparency is fun- damental to support both introspection, that is, the capability of a cloud provider of examining and observing its internal processes, and outrospection, that is, the ability of customers and service providers to examine and observe cloud's internal processes, involving their activities, data, and applications, for security purposes. A proper solu- tion to assurance in the cloud should embrace both introspection by cloud providers and outrospection by cloud customers (tenants in general) and, therefore, balance the burden of security processes and controls between providers and customers. In this section, we survey approaches to the verification and validation of cloud infrastructures. Cloud assurance approaches have been categorized according to the classification of assurance techniques in Section 2.2: testing, monitoring, certification, audit/compliance, and SLA. These categories of solutions focus on increasing trust in the cloud infrastructure, can target all levels of the cloud stack, and aim to empower cloud users. Table III shows our classification of cloud assurance solutions based on the implemented assurance techniques.

2.6 Certificazione

T

Capitolo 3

FedRAMP - Federal Risk and Authorization Management Program

3.1 Introduzione

.

.

3.2 Struttura

.

.

.

.

3.3 FedRAMP readiness

.

.

.

.

3.4 Controlli di sicurezza per la conformità

.

.

.

.

3.5 FedRAMP in Amazon AWS

.

.

Capitolo 4

Moon Cloud: un framework per il monitoraggio e la verifica della sicurezza

4.1 Introduzione

.

4.2 Architettura e componenti

.

.

.

.

4.3 Moon Cloud come strumento di verifica della compliance

.

4.3.1 Controlli di sicurezza

.

.

.

.

4.3.2 Regole di valutazione

.

.

.

.

Capitolo 5

Implementazione dei controlli di sicurezza FedRAMP in Moon Cloud

5.1 Controlli automatici

5.1.1 Una sezione per ogni security control implementato

5.2 Controlli ad interazione umana

5.2.1 Questionari per l'assessment dei processi di business

Capitolo 6

Validazione del framework

6.1 Deployment di Moon Cloud

6.2 Sicurezza del deployment

6.2.1 Caratteristiche del deployment

6.2.2 Confidenzialità

6.2.3 Integrità

6.2.4 Disponibilità e affidabilità

6.2.5 Data remainance

6.3 Scalabilità e Prestazioni

6.3.1 una sezione per ogni security control eseguito

Capitolo 7

Conclusioni e sviluppi futuri

Bibliografia

[1] National Institute of Standards e Technology (Peter Mell Timothy Grance). *The NIST Definition of Cloud Computing*. URL: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[2] Claudio A. Ardagna et al. «From Security to Assurance in the Cloud: A Survey». In: *ACM Comput. Surv.* 48.1 (lug. 2015), 2:1–2:50. ISSN: 0360-0300. DOI: 10.1145/2767005. URL: http://doi.acm.org/10.1145/2767005.

[3] C. A. Ardagna et al. «On the Management of Cloud Non-Functional Properties: The Cloud Transparency Toolkit». In: *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*. Mar. 2014, pp. 1–4. DOI: 10.1109/NTMS.2014.6814039.

[4] K.M. Goertzel et al. *Software Security Assurance: A State-of-the Art Report (SOAR)*. Information Assurance Technology Analysis Center, 2007. URL: https://books.google.it/books?id=xxHPMgEACAAJ.