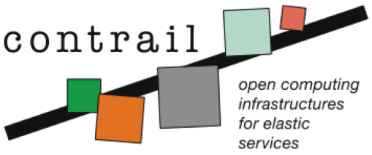

Project no. 257438

CONTRAIL

Integrated Project
OPEN COMPUTING INFRASTRUCTURES FOR ELASTIC SERVICES

Security Test and Evaluation
D7.4

Due date of deliverable: January 30^{th}, 2014
Actual submission date: March 2^{nd}, 2014

Start date of project: October 1^{st} 2010

Type: Deliverable
WP number: WP7
Task number: T7.4,T7.5

Responsible institution: Inria
Editor & and editor's address: Roberto Cascella, Inria

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history:

Version	Date	Authors	Institution	Section affected, comments
0.0	30.07.13	Jens Jensen	STFC	Fairly thorough outline, first few sections
0.1	05.09.13	Roberto Cascella	Inria	Outline
0.2	02.10.13	Aleš Černivec	XLAB	Background section update
0.3	02.10.13	Aliaksandr Lazouski	CNR	Isolation against threats section update
0.4	15.10.13	Aliaksandr Lazouski	CNR	Security evaluation by feature section update
0.5	28.10.13	Cheney Ketley	STFC	Updated Evaluation and assessment - Assessment based on general testing
0.6	02.01.14	Jens Jensen	STFC	More stuff
0.7	15.01.14	Jens Jensen	STFC	Additional threats analysis
0.8	28.01.14	Jens Jensen	STFC	Notes added for the remaining empty sections
0.9	29.01.14	Uros Jovanovic	XLAB	Messaging and Accounting sections
0.9.1	30.01.14	Ales Cernivec	XLAB	ConPaaS Security
0.9.2	31.01.14	Jens Jensen	STFC	Cloud security business opportunities enabled by Contrail
0.9.3	05.02.14	Roberto Cascella	INRIA	IaaS and VIN section. Revision before internal review
0.9.4	07.02.14	Aliaksandr Lazouski	CNR	Protection against race condition section
0.9.5	25.02.14	Jens Jensen and Roberto Cascella	STFC, INRIA	Reviewers comments addressed
0.9.6	27.02.14	Jens Jensen	STFC	More OAuth analysis
0.9.7	28.02.14	Roberto Cascella	INRIA	Executive summary and introduction
0.9.8	02.03.14	Jens Jensen	STFC	Addressing comments and updating some analysis.
1.0	02.03.14	Roberto Cascella	INRIA	Final version

Reviewers:
Gaetano Anastasi (CNR) and Uros Jovanovic (XLAB)

Tasks related to this deliverable:

Task No.	Task description	Partners involved°
T7.3	T7.3 – Controlling Access in Virtual Infrastructures: This task aims at evaluating and implementing fine-grained access control for the core virtual infrastructure. It includes secure access to virtual machines and applying TPM technologies for securing hypervisor and guest OS, providing a distant chain of trust.	**XLAB**, Inria, CNR, STFC, HP-IIC, TISC
T7.4	Security through Compartmentalization and Isolation: This task will develop techniques to guarantee that access to one virtual machine/server does not reveal knowledge about other virtual machines/servers and services running on the shared physical infrastructure.	XLAB, CNR, **STFC**, HP-IIC, TISC
T7.5	Security Analysis, Test and Evaluation: This task will carry out the necessary analysis, testing and evaluation of the security mechanisms developed in CONTRAIL.	XLAB, CNR, **STFC**, TISC

°This task list may not be equivalent to the list of partners contributing as authors to the deliverable
*Task leader

Executive Summary

Security support is a critical component of the Contrail project, because potential customers of the Contrail framework (i.e., deployers of cloud federations) will adopt Contrail only if they feel confident in the security, and potential end users will use Contrail-based federations to run their cloud applications only if they feel the environment is secure. The Contrail framework must provides adequate and comprehensive security support, tailored for the meeds of Cloud computing (specifically, it must be elastic.) In fact, since the Cloud environment is mainly targeted at business users, an enhanced level of security support is a key factor in allowing the adoption of Contrail-based cloud federations, because business users' data could have a high economic value and data theft or damage could result in considerable financial and reputational loss.

Contrail WP7, "Security in Virtual Infrastructures," covers all the classic security issues, such as authentication, authorization, data confidentiality and integrity, as well as accounting and logging. Also in scope of WP7 is the evaluation of the Contrail security components and of the entire Contrail federation, looking for vulnerabilities, and evaluating the isolation of tenants against others.

Hence, the design of the Contrail security (ConSec) components has been performed while designing the rest of the Contrail framework, thus achieving a high level of integration with the other framework components. The final specification and architecture of ConSec was already covered in D7.3: the planned support for virtualisation, along with the support for data confidentiality and integrity, and for accounting and logging.

This document is the final output of WP7; it covers the final security solutions designed and used in the Contrail project, and evaluates them qualitatively or quantitatively. We have carried out a qualitative analysis and evaluation of how the security requirements have been satisfied; the requirements have been identified at the beginning of the project and then later refined during the course, based on the analysis and verification of Contrail components needs, security standard practise, and feedback based on potential usage of the security solutions by external communities. The quantitative analysis concerns performance measurements and overhead impact on the Contrail system. The evaluation is completed with an assessment of the security based on general testing. Of course not every target was met – had we met every target, it would have been a sign that we had not set them ambitiously enough – but ConSec nevertheless managed to deliver an integrated end-to-end security, covering the whole path from the user's desktop over federation services to the virtual machine and the secure mounting of filesystems within it. ConSec delivered authentication using federated identity management, authorisation, accounting, access control policies, usage control, and provided security for both infrastructure and elastic services. ConSec made use of standards

and existing code and protocols whenever possible, and developed code to "glue" things together. Indeed, ConSec has actually contributed to OAuth by delivering a python implementation (in collaboration with a climate science project), and has made standards-compliant extensions to OAuth to control and monitor the delegations.

This deliverable provides the updated list of security requirements initially identified in D7.1 and whether the security goals have been addressed and the resolution in Contrail. The security goals have been grouped in 4 sets: authentication, authorization, data confidentiality and integrity, and logging.

The techniques to guarantee security through compartmentalisation and isolation are detailed for the components managing the resources at the infrastructure level and for ConPaaS (Contrail PaaS) that can deploy directly on the IaaS. The techniques adopted are meant to ensure that access to one virtual machine/resource does not reveal knowledge about other virtual machines/servers and services running on the shared physical infrastructure (isolation). This deliverable discusses in details the techniques adopted in Contrail to protect users in a multi-tenanted environment, such as the clouds in general and federated clouds in particular: federation level authorization support, policy framework to access resources and data, and authentication with associated level of assurance, a Quality of Protection (QoP) term which via the SLA mechanism provides dynamic support for protecting critical applications while leaving less critical ones more easily accessible.

We have performed extensive penetration tests to determine the vulnerabilities of the Contrail system: some were identified and in some cases already eliminated, some others need further programming due to the research purpose of the software developed in Contrail, but to a large extent they highlight the need to use best practices for configuration of security (use of certificates, correct configurations of firewalls). Finally, we have analysed the OAuth delegation protocol used in Contrail, which has already undergone extensive formal analyse

Contents

1 Introduction

This document is the final report of the evaluation of the security of the infrastructure developed or deployed by the Contrail project. Its main focus is on the tasks T7.4 (isolation, the level of protection users have from each other or from the environment) and T7.5 (security evaluation).

Security is one of the important achievements of Contrail. A specific software release (1.3) was made to make the software available to external communities for further reuse. The security developed in Contrail consists on several parts grouped into two main components: ConSec (Contrail Security) mainly for authentication and the UCON model for authorization. The analysis and evaluation of the security code developed in Contrail refer to Contrail Release 1.4.

1.1 Structure of the document

This document is made of 5 main sections and 1 appendix. Section 2 discusses the potential threats and attacks to the Contrail security system. The requirements identified in D7.3 [24] are presented along with the resolution. Finally this section presents how these requirements have been implemented for each component.

Section 3 presents the protection mechanisms adopted in Contrail to isolate the components and protect them from attacks. For each component we detail the way security has been implemented.

Section 4 discusses the solutions identified and implemented to protect the Contrail system from attacks either carried out by federation users (isolation in a multi-tenant environment), Cloud users, and system administrators. This section also presents how the authorization framework is used to enforce isolation and the how the policy about level of assurance for authentication have been implemented.

Section 5 presents the security evaluation from a qualitative and quantitive perspective. This section summarises also the general testing performed on Contrail from a security point of view and discusses what are the countermeasures adopted to address the vulnerabilities of the well know OAuth protocol analzyed with formal methods in literature.

Section 6 presents the way the security is operated in Contrail at the federation level, in particular with respect to the need for federated identity frameworks.

Section 7 concludes this deliverable and provides a general assessment of the work achieved in Contrail for security.

Appendix A details the penetration tests against the portal application and the called components, particularly the federation-api.

2 Background: threats, use cases, attacks

A key part of the security evaluation is the question: *what are we evaluating?* Called the *Target of Evaluation*, or TOE, this is the software, product, or infrastructure, whose security is being evaluated. We refer the reader to Deliverable D10.4 and D10.6 [16, 10] for a detailed description of the final Contrail architecture and to Deliverable D7.3 [24] for the security components. The TOE is Contrail release 1.4 (final release is 2.0) whenever the evaluation refers to a software product. If the evaluation needs to refer to a deployment of Contrail, it will (unless specified otherwise) refer to a cross-site deployment of the Contrail components with a site providing federation portals/login for other sites, another site providing VEP (Virtual Execution Platform), etc. In the case of VEP, the cloud resource it protects is either a private cloud running wholly within a site, or a public cloud provided by a commercial provider using VEP to manage its datacenter.

> Unless specified otherwise, the TOE is Contrail Release 1.4 as a software product, and Contrail Release 1.4 deployed as a cross-site e-infrastructure. The final release of Contrail is Release 2.0, but all security features were already integrated in Release 1.4 making the evaluation still valid.

Additional useful questions are then: what are we trying to protect? And who are we trying to protect it from? For the former, we shall look at this in more detail in the coming sections. As regards the latter, it is the subject of this section as well as section 4.

2.1 Security goals

In D7.1[23], *Security Requirements, Specification and Architecture for Virtual Infrastructures* (March 2011), a number of security goals were identified. The purpose of this section is to list them, so we can assess whether they were addressed in Contrail or not.

As already listed and defined in D7.3 [24], we have grouped the requirements into 4 sets:

- *Authentication*,
- *Authorization*,
- *Data Confidentiality and Integrity*,
- *Logging*.

Taking all these four sets of requirements users should be provided with **transparency** and **accountability** provided by the Contrail Federation and guaranteed by other Contrail components like VEP in a transparent way for the user. Moreover, access to user's data is only in domain of the user. Here we provide the updated list with the information (reference) about goals we have addressed and where they were addressed. We have created a table for each group with description of the goal and a short description of the resolution.

2.1.1 Authentication

Goals defined in the Authentication set are given in this subsection. These goals define requirements for authentication of users to the main levels (parts) of the Contrail Federation: *Federation level*, *Provider level*, *ConPaaS*, *GAFS*, *VIN*.

Ref	*Description*	*Resolution*
WP7-ATH-01	Accesses of users to the Federation must be authenticated	Done on the Federation API level
WP7-ATH-02	Accesses of users to the Cloud IaaS services must be authenticated	Authentication mechanism is provided by VEP API
WP7-ATH-03	Accesses of users to the ConPaaS services must be authenticated	ConPaaS Director Service uses local database of users and interfaces with Contrail IDP
WP7-ATH-04	Accesses of users to the Global Autonomous File System should be authenticated	GAFS authenticates users using X.509 certificates
WP7-ATH-05	Accesses of users to the Virtual Infrastructure Network should be authenticated	Use of delegated user certificates
WP7-ATH-06	Federation users want to perform the authentication process only once, when they access the Federation	Single Sign On (SSO) mechanisms provided by Contrail IDP (using SimpleSAMLphp SAML2.0 solution)
WP7-ATH-07	Federation users want to use their existing credentials to authenticate on some Cloud services	Contrail IdP acts as IDP bridge to external identity providers
WP7-ATH-08	It should be possible to use existing authentication infrastructure	Contrail IdP supports federations: Shibboleth, external SAML2 IdPs, OpenID
WP7-ATH-09	Cloud service providers wants to use their native authentication framework in federation Cloud scenario	Contrail IdP (SimpleSAMLphp) provides a way to extend authentication modules: writing a new one is possible since it is well documented

2.1.2 Authorization

Goals defined in the Authorization set are given in this subsection. These goals define requirements for the authorization of the users to the main parts of the Contrail system: *Federation level*, *Provider level*, *ConPaaS*, *GAFS*, and *VIN*.

Ref	*Description*	*Resolution*
WP7-ATZ-01	Accesses of users to the Federation must be authorized	Done on the Federation level with calls to the PDP service
WP7-ATZ-02	Accesses of users to the Cloud IaaS services must be authorized	Authorization mechanism is provided by the Provisioning Manager and VEP via the PM
WP7-ATZ-03	Accesses of users to the ConPaaS services must be authorized	ConPaaS provides certificates for users and checks are made for the validity of these
WP7-ATZ-04	Accesses of users to the Global Autonomous File System should be authorized	GAFS authenticates users using X509 certificates and authorization is made based on certificates
WP7-ATZ-05	Accesses of users to the Virtual Infrastructure Network should be authorized	Use of delegated user certificates OAuth2 is used for obtaining delegated user certificates
WP7-ATZ-06	Cloud service providers want to control the usage of their resources	Monitoring and Accounting is available to be used on the Federation level Monitoring also at the Provider level

2.1.3 Data Confidentiality and Integrity

Goals defined in the Data Confidentiality and Integrity set are given in this subsection. These goals define requirements for the data exchange policies between main parts of the Contrail software stack and end-users: *Federation level* – end-user, *Provider level* – end-user, *ConPaaS*, *GAFS*, and *VIN*.

Ref	*Description*	*Resolution*
WP7-DCI-01	Confidentiality and Integrity of data exchange between users and the Federation must be assured	All communication mechanisms use Secure communications (SSL), digital signatures are used (SAML2)
WP7-DCI-02	Confidentiality and Integrity of data stored by the Federation must be assured	Communication between DB and servlet is local; DB is not exposed, passwords are encrypted (Blowfish)
WP7-DCI-03	Confidentiality and Integrity of data exchange between users and Cloud IaaS services must be assured	Communication between federation and provider level is secure, mutual service authentication is done (x509)
WP7-DCI-04	Confidentiality and Integrity of data stored on Cloud IaaS services must be assured	VEP has local database, not exposed, and use of user certificates and mutual authentication.
WP7-DCI-05	Confidentiality and Integrity of data exchange between users and ConPaaS services must be assured	Use of user certificates and mutual authentication between ConPaaS services
WP7-DCI-06	Confidentiality and Integrity of data stored by ConPaaS services must be assured	ConPaaS Director does not expose ConPaaS database; ConPaaS Managers use x509 authentication
WP7-DCI-07	Confidentiality and Integrity of data stored on the Global Autonomous File System must be assured	The user can enable checksumming of file content to ensure data integrity
WP7-DCI-08	Confidentiality and Integrity of data exchange on the GAFS must be assured	Use of secure communication using client/server x509 certs, mutual authentication
WP7-DCI-09	Confidentiality and Integrity of data exchange through the VIN must be assured	Use of client/server x509 certificates, use of delegated user certificates

2.1.4 Logging

Goals defined in the Logging set are given in this subsection. These goals define requirements for the logging mechanism that should be used by main parts of the Contrail: *Federation level*, *Provider level*, *ConPaaS*, *GAFS*, and *VIN*. The logging should enable end-users traceability of their deployments and their the usage of their data by the system.

Ref	*Description*	*Resolution*
WP7-LOG-01	Accesses to Cloud resources must be logged	Logging mechanism is provided on the Federation and Provider APIs level
WP7-LOG-02	Users that access logs must be authenticated	Only authenticated user has access to her logs (calls made by the user)
WP7-DCI-03	Accesses of users to the logs must be authorized	Authorization mechanism is provided on the API level (XACML policies)
WP7-LOG-04	Confidentiality and Integrity must be assured	Logs are communicated over secure channel

2.2 Security threats and required investigations

Extending work by the CSA ([7]), Table 1 summarises the threats to components of Contrail which were outlined in D7.1, section 3.2. The $\surd$ symbol denotes that the threat *does* need to be checked and investigated for that component (and in a few cases, we have renamed the threat as listed in D7.1.) The $\diamondsuit$ symbol denotes special problems which need to be specifically addressed: for GAFS, several data threats were identified and will have to be investigated. This will be the subject of section 3.7 and more generally in section 4.6. Similarly, the analysis of the threats associated with the "longer trust chain" for VEP is covered in section 3.2

2.3 Standards, certifications, guidance

Here we list some of the relevant ISO standards relevant to Contrail system.

2.3.1 ISO/IEC 27002:2013 Information technology

The 27002 ISO standard [13] is a popular, internationally-recognized standard of good practice for information security. It provides a code of practice - it is a generic, advisory document. It recommends information security controls addressing information security control objectives arising from risks to the confidentiality, integrity and availability of information.

Contrail software as a cloud federation provider incorporates some of the primary functions provided by ISO 27002:

- collects, stores and monitors log data throughout its collection, transportation, use and disposal

	VIN	GAFS	IaaS	VEP	ConPaaS	SLAM
Abuse/Nefarious use	√		√		√	√
Special Data risks		♢				√
Insecure interfaces/APIs	√					
Malicious Insiders	√	√				
Account/service/traffic hijack			√			
Shared tech vulnerabilities	√					
Unknown risk profile	√				√	
Activity patterns	√					√
Multiple jurisdiction			√			√
Longer trust chain			√	♢	√	
Mutual auditing						
Machine image security	√		√		√	

Table 1: Table of threats

- protects the integrity of the log data and records activities making the records available to users
- enables role-based data views, masks views on the data collections based on user roles
- it still lacks the security intelligence through analytics and statistics - it does provide insights into system to some extent

Comparing Contrail Control specifications to the specifications provided by CloudAudit-ISO27002 [11] comparison matrix, we can find a lot of control specifications satisfied by Contrail. Comparison of the all control specifications is out-of-scope of this document.

2.3.2 ISO/IEC 27017

The 27017 ISO standard [15] (in draft state) provides guidance on the information security elements/aspects of cloud computing, it recommends cloud-specific information security controls. The standard will be a code of practice recommending relevant information security controls for cloud computing, based on and extending those recommended by ISO/IEC 27002. Publication is very unlikely before 2014, quite possibly not until 2015. The project has widespread

support from national standards bodies plus the Cloud Security Alliance among others. The guidance specifies some 35 control objectives (organized in *security control categories*) to protect the **confidentiality**, **integrity** and **availability** of information. Moreover, it also defines the continuity of information security that should be planned and reviewed as an integral part of the system. The revised 2013 version of ISO/IEC 27002 was published in September 2013.

2.4 Protecting distributed environments

Unsurprisingly, we use cryptographic methods to secure the infrastructure – indeed, without end-to-end (point-to-point) cryptographic security, most protocols would be vulnerable to impersonation, replay, or man-in-the-middle attacks. In fact, we even rely on security features of socket level protocols (TLS, RFCs 4346 and 5246) to prevent some of the replay attacks, as well as offering the point-to-point authentication, message integrity, and confidentiality.

However, cryptographic protocols are also subject to cryptanalysis and attacks, and it is well known that the choice of weak ciphers can lead to a compromise of the security goals (loss of integrity, or loss of confidentiality.) In general we define three kind of low-level (cryptologic) attack models:

- attack on Plaintext or Ciphertexts (section 2.4.1)
- attack on Protocols (section 2.4.2)
- attack on the Infrastructure (section 2.4.3)

In these three sections, we examine each of these cases from the point of view of the federation (core), the IaaS provider (VEP), GAFS, and ConPaaS.

2.4.1 Attack on Plaintext or Ciphertexts

In general there exists plethora of different plaintext or ciphertext attacks:

- Ciphertext-only attack. This is what most attackers will see, and with strong ciphers, message security should be assured (whether an attacker can store it and attack it later with greater computing power is a factor which depends on the choice of algorithms and recommended best practice.)
- Known-plaintext attack. This is where an attacker could guess some of the message and use it to derive the key. Such an attack is quite possible: as we use REST web services, a substantial section of the initial part of each message will be an HTTP header. We note that e-commerce has the same issue, and, in general, has more at stake, so we recommend that good ciphers be chosen.

- Chosen-plaintext attack. An attacker would not have the ephemeral symmetric key which is used to secure the connection, so an attacker could only insert chosen plaintext into the service if they send it to the legitimate user and trick them into sending it into the socket. This does seem a bit far fetched and not of great value. More likely is the type of attack where the attacker uses cross site scripting or similar (see the analysis of OAuth in section 5.4 for a discussion of this type of attack.)
- Adaptive chosen-plaintext attack. If a chosen plaintext attack is not feasible, an adaptive one is less so.
- Chosen-ciphertext attack. This attack is possible: an attacker would simply intercept some of the envrypted messages from the legitimate user and replace them (we assume a powerful attacker who controls the channel.) However, this attack would lead to the server receiving gibberish which cannot be deciphered into meaningful text (or, to be precise, the probability that it makes sense is negligible.) The server would thus get confused and would not be able to respond properly to the client. Eventually the connection would time out, or garbage data would be received leading to an error being returned from the SSL/TLS libraries to the server, resp., client.
- Adaptive chosen-ciphertext attack: here an attacker would need to see the effect of inserting chosen ciphertext, but as the effect of the first insertion is highly likely to be unhelpful – other than being disruptive to the user – there seems to be little additional value in a chosen ciphertext attack.

Federation level In Contrail we address these issues with the use of well known mechanisms preventing these kind of attacks on the communication level. For encryption of database passwords on the federation level we use Blowfish [29] (more precisely BCrypt[1] which implements OpenBSD-style Blowfish). We use randomly generated salt with `log_rounds` set to 12. Since the creation of the algorithm it has been analyzed considerably, and it is slowly gaining acceptance as a strong encryption algorithm. Blowfish is unpatented and license-free, and is available free for all uses.

For message exchange security on the communication level we are using SSL libraries provided by Python and Java SDKs.

Provider level On the provider level SHA1 is used to hash passwords for accessing the accounts on the Virtual Execution Platform (VEP). SHA-1 is the most widely used of the existing SHA hash functions, and is employed in several widely used applications and protocols. In 2005, cryptanalysts found attacks on SHA-1

[1] http://www.mindrot.org/projects/jBCrypt/

suggesting that the algorithm might not be secure enough for ongoing use[2]. NIST required many applications in federal agencies to move to SHA-2. This has been changed in the latest version of VEP 2.1 integrated in Contrail Release 1.4.

Global Autonomous File System – GAFS GAFS is a distributed filesystem, providing high availability data services based on federated credentials. GAFS can be configured to use X509 certificates[3] and SSL support can be also used to encrypt the data for the transmission. Encryption prevents eavesdropping by third parties among instances of GAFS services. This applies both to connections between clients and servers as well as between different servers (e.g. for the replication of data).

ConPaaS In ConPaaS passwords are hashed with `md5` producing a 128-bit (16-byte) hash value. Additionally, digest is returned as a string of double length, containing only hexadecimal digits which may be used to exchange the value safely in non-binary environments:

```
hashlib.md5(password).hexdigest()
```

Additionally, ConPaaS uses PKI to authenticate applications and users.

2.4.2 Attack on Protocols - network security

Main attacks that are possible on the protocols used within the Contrail are listed bellow.

- Eavesdropping
- Known-key attack
- Replay attack
- Impersonation (Masquerading) attack
- Dictionary attack
- Forward search attack
- Interleaving attack

Contrail services use certificates and digital signatures to protect against forgery to prevent the signer from repudiating a signed document. By using PKI Contrail avoids eavesdropping, known-key attacks, and all other attacks listed above. Lets go through the Contrail stack and see how Contrail protects from the attacks.

[2] https://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html
[3] http://www.xtreemfs.org/xtfs-guide-1.4/index.html#sec:cfg_ssl

Federation level All services on the Federation level use client-server SSL authentication using X509 Certificates. All communication goes over encrypted channel using session key to encrypt data flowing between parties. This provides data/message confidentiality. All federation services providing REST API (**Federation API**, **OAuth Authorization Server**, **CA Server**, and **Accounting Manager**) use similar mechanism to enable SSL communication – services use Tomcat servlet container with enabled SSL Connector. Current implementation does not prevent the use of weak ciphers – it is simply an issue of configuring the ciphers properly. The default configuration should be set to only allow strong ciphers: Administrators are encouraged to set connector for Tomcat by defining connector properties:

```
sslProtocol="SSLv3" ciphers="SSL_RSA_WITH_RC4_128_MD5,
SSL_RSA_WITH_RC4_128_SHA, TLS_RSA_WITH_AES_128_CBC_SHA,
TLS_DHE_RSA_WITH_AES_128_CBC_SHA,
TLS_DHE_DSS_WITH_AES_128_CBC_SHA, SSL_RSA_WITH_3DES_EDE_CBC_SHA,
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA,
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA"
```

There is a general issue with default passwords being set in the system, particularly with such things as the default administrator account. It would be prudent to adapt the installation process to query the user for passwords, but we have not done that, as it would preclude automatic installs. Instead, we have provided a *checking script* which examines the configuration files for unconfigured passwords and other installation gotchas, and highlights the error it finds (but leaves it to the installer to correct them, as, in general, installers are expected to know what they are doing...)

This script was written to be extensible, taking a list of predefined tests and applying them to the configuration on a given Contrail federation core system, but with tests being easy to add in the future[4].

There is also the question of testing the install: we cannot, obviously, test ConSec without security, yet it would be tempting fate to install dummy certificates with the system. Someone would run a federation with the dummy certificates, just like occasionally one sees Apache web servers with the "snakeoil" CA – as the private key is distributed with the source, the certificates are completely insecure. Instead, we developed a certificate generation script which is able to generate dummy host certificates for an install, thus enabling tests to run. It is, however, not recommended to use it in practice, as the certificates would not be issued by a trusted CA – it is precisely the host certificates that need to be signed

[4]The `file` command in Linux is an analogy: it is told how to recognise different files, and extending it to recognise more files is quite simple.

by a CA external to Contrail, to bootstrap the security in the usually distributed infrastructure.

Provider level Similar to Federation level, provider level also uses the same mechanism by enabling SSL connectors in servlet containers serving these Contrail services.

Global Autonomous File System – GAFS As already discussed in the section 2.4.1, GAFS uses PKI to protect all communication channels and therefore prevents the listed attacks on the protocol.

ConPaaS ConPaaS uses SSL settings by default in the Nginx settings.

2.4.3 Attack on the Infrastructure - host security

Perimeter security often involves network intrusion detection systems (NIDS; e.g. Snort, Suricata). NIDS systems inspect and monitors local traffic for irregularities, e.g.: port scans, DOS attacks, Known vulnerability exploit attempts. This kind of inceptions and preventions are possible with proxying through such inspection/prevention system. Contrail does not use such systems so this kind of perimeter security is not used within Contrail.

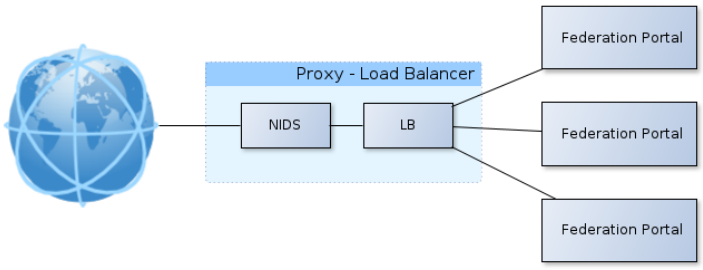

Figure 1: Proposal of NIDS system in-front Contrail Federation portals.

In figure 1 we propose such system and is easily deployable on existing IaaS. Of course there are limitation to such systems: it is a fact that such load balancer would be single point of failure if it is only one. In this case Contrail could provide several access points over resource discovery system. Contrail does not provide such system.

Other infrastructure attacks that are possible:

- Eavesdropping

- Known-key attack
- Replay attack
- Impersonation (Masquerading) attack
- Dictionary attack
- Forward search attack
- Interleaving attack

This section describes what Contrail does for the infrastructure (host) protection. We describe features by major Contrail components similar to subsections before.

Federation level Working with certificates is a good and proven way to implement security in a distributed infrastructure, as long as the issuance can be sufficiently controlled. With the use of certificates Contrail successfully shields from attacks such as **eavesdropping**, **replay attacks**, **impersonation (Masquerading) attack** and other kind attacks. Of course, if client's private key is stolen, Contrail cannot protect the calls made directly on Federation API component using client (delegated) certificates. With the issuance of delegated certificates, we only assure that the certificate will expire in short period of time. After that, user of the certificate will need to ask for new one and if the original user has detected the problem (monitoring, accounting and alerting services can be used to monitor events), and the user could change the password. Password is encrypted and hidden on the federation database and can be changed only with knowing of the federation password. If the federation password is stolen, federation coordinator needs to do intervention. The coordinator should create new password for existing federation user.

Provider level Communication between Federation and Provider Level is also protected with delegated certificates (similar to the Federation level description).

Global Autonomous File System – GAFS GAFS security features are:

- GAFS is made up of distributed nodes, each of which needs to be trustedâ and hence authenticated
- users are authenticated to GAFS,using X509 Certificates and to authorize user requests based on attributes within the certificates e.g. when mounting a filesystem in a virtual machine
- GAFS supports X.509 certificates (mutual nodes authentication) and SSL encrypted connections

It is worth noting that it is not possible to mix SSL-enabled and non-SSL services in an GAFS installation. Each GAFS service needs a certificate and a private key in order to run in secure mode. GAFS services also need a trust store that contains all trusted Certification Authority certificates.

ConPaaS ConPaaS handles infrastructure (host) attacks with the use of load balancers but no NIDS is placed within the infrastructure (see figure 1). ConPaaS has internal limit of used credentials per user account. This limit is used in order the user (or malicious user) drains the account used for provisioning the IaaS (federation) layer. This protects from EDOS attacks (Economic Denial of Sustainability attacks[5]).

2.5 Protecting virtual environments

VEP supports mysql as well as sqlite as database backends, as already described in Contrail Deliverable D7.2 [18]. There is a data security risk if sqlite backend is used, data is accessed with the (UNIX) privilege of the application, rather than as a separate account protected by the database itself. In other words, the data is protected only by the filesystem access. Keeping all data encrypted inside sqlite can be a mitigation, with keys obtained from a secure key distribution mechanism.

3 Isolation by component

3.1 Infrastructure, internal security, protection

In Contrail, the IaaS services run on the provider's infrastructure, and under the provider's terms and control. Therefore, guarantying protection at the infrastructure level concerns several issues: i) isolation of the component per se with protection from unauthorised users and Cloud providers; ii) protection and integrity of the data stored on Cloud IaaS services; iii) protection and integrity of the data exchanged between users and Cloud IaaS services.

The internal security is not only achieved at the Contrail component level but it is also determined by the level of protection that can be achieved in IaaS providers and platforms. In this section we analyse in detail each Contrail service and the infrastructure elements to provide a thorough analysis of the internal security.

[5] `http://goo.gl/rcVqvJ`

3.2 VEP

The Virtual Execution Platform (VEP) is the component of the Contrail architecture that enables IaaS service providers to participate in the Contrail Cloud Federation. VEP also allows the Contrail Federation to deploy end users' distributed applications, described in an appropriate Open Virtualization Format (OVF) [4] compliant description, on the cloud provider's computational resources. These applications have specific resource requirements that have an associated Service Level Agreement (SLA) which must be honoured, and any violation of such an agreement must be notified to the Contrail Federation. VEP also provides support for application monitoring and reporting that enables SLA enforcement by the Federation. A detailed description of the VEP component can be found in deliverable D5.2 [12] and D5.3 [17].

Security of VEP is important because:

- It is (or can be) used to bootstrap security in components that it manages (e.g. a VM), or indirectly in components beyond that (such as a VIN agent running inside the VM, or to connect to a GAFS server).
- VEP manages user credentials for the resource. When a user authenticates (via the Federation services) to VEP, VEP maps the user to the resource user identity. This is just a name mapping because the user can access VEP only through the SLA Manager at the provider level (SLAM-P) and the Provider Manager (PM). VEP trusts completely these two components and they are registered in VEP as administrators. The PM and the SLAM-P send command to VEP specifying the user they are acting for. Then, VEP (as a Provider) will need to hold the full credentials that the user will be using to access the Resource.

This section outlines the security measures enforced by the VEP component in terms of authentication of users and external Contrail components, authorisation, and confidentiality and integrity. The description is an update of previous Contrail deliverable D7.3 [24].

3.2.1 IaaS Resource Account Mapping

A user identity is organised on three levels: federation; provider – VEP; IaaS. The management of the identities is done internally at VEP. VEP has its own database with the ID of the users at the provider level and it handles the mapping with the Contrail Federation User IDs. The mapping is needed because the Provisioning Manager (PM) and SLA Manager at provider level (SLAM-P) identify users with their federation ID and the same federation ID can be used in SLA and SLA templates.

The new createParty call will be invoked by the SLAM at the provider level as part of its process of registering federation users and create a username UUID to enable the use of OAuth for the request of a delegated certificate.

The map between the User ID at the VEP with the one of the IaaS is managed directly by VEP. VEP generates randomly a username and password which are hidden to the user so that the user cannot access directly IaaS resources.

3.3 Virtualisation

The VM itself needs to be bootstrapped with a certificate, which is VEP's responsibility, possibly indirectly. This certificate is used for mounting a GAFS volume and to set up the IPSec tunnels to establish a VPN with VIN. The problem is that VEP may not know the hostname of the new machine being set up (it is known to the resource controller), but only VEP is trusted by the online CA which signs the dynamic host certificates. It is therefore necessary to have a callback mechanism within the VM startup which calls back to VEP to obtain the certificate. This step is required for the VIN security and to mount GAFS volumes. The step is implemented using the OAuth2 delegation code.

Similarly, it is important that VM Images are stored securely, i.e., the *integrity* of a VM Image is guaranteed. As VM Images are stored in GAFS (D5.2 [12], section 6.3), see section 3.7 for a further discussion of integrity of GAFS.

3.3.1 Contextualisation

The process that configures the VMs at boot time, starting from a master image passing arbitrary data, is known as contextualisation. VEP allows VMs contextualisation: the user can specify contextualisation values as property in the OVF document and others are added automatically by the Provisioning Manager and VEP. The user and the Provisioning Manager can only pass key-value data, i.e., strings. On the other hand, VEP can add files to the VM and these files include certificates and a init-script to allow networking and remote storage configuration. VEP does not have its own contextualisation mechanism, hence it uses the features provided by the IaaS it works with; Contrail currently supports OpenNebula and OpenStack. All the contextualisation data sent to VEP are encrypted as all the message that VEP receives; hence, these data are passed through the appropriate APIs to the IaaS. Then, the IaaS processes and adds them to the VMs.

3.3.2 Access to resources

The need for regulating accesses at VEP level through an adequate authorisation support has been stated by the requirement WP7-ATZ-02, and it is an high priority

requirement. Moreover, the requirement WP7-ATZ-06 states that also the usage of Cloud resources must be regulated by a proper support. The integration of the Usage Control System with the VEP v1.1 satisfied the previous requirements. With the new version of VEP v2.1 the authorisation control is managed by the Provisioning Manager (PM), thus, all actions on the VEP are performed by the PM upon verifying the policy.

The security relevant actions that are performed on the VEP are

- manage (creation,deletion) of Constrained Execution Environments (CEE)
- manage (creation,deletion Application
- deploy VMs using VM templates individually
- run an application or individual VMs
- stop an application or individual VMs
- suspend an application or individual VMs
- resume an application or individual VMs

VEP also provides some data concerning the status of the resources it manages; this is required to enable the support of the authorization by the PM. The status is represented as resources' attributes, such as the total number of running VMs or the number of VMs running on behalf of the user that requested a new access.

Some possible security policies that could be enforced from the PM to VEP are the following:

- a user can deploy a VM using a VM template if the number of current active VMs is less than a given threshold
- a "gold" user can store 20 VMs, while a "silver" user is allowed to store only 10 VMs, where "gold" and "silver" are possible values of the group user attribute.
- a user is allowed to run a VM as long as his reputation is greater than a given threshold

3.3.3 Confidentiality and integrity

Provisioning Manager services acting on behalf of the Contrail federation services communicate with a provider's VEP instance over a RESTful interface using a HTTPS channel to provide data confidentiality. Moreover, an Internal Access Control Module [12] provides internal access control checks on whether a resource object's owner is the same as the subject on behalf of which the action is

being processed (or that the subject has appropriate rights to execute the action). Data integrity is guaranteed with proper hashing.

VEP's data is stored within a local database which cannot be accessed outside VEP. As such there is no need for special data integrity and confidentiality mechanisms to be in place at the database level.

3.3.4 Firewalls

Firewalls are an important part of providing security in any IT infrastructure, particularly one connected to the Internet. Any cloud platform must support this and a commercial cloud provider will have measures (firewalls, network monitoring) in place to protect both their infrastructure and the users' VMs. In part, the obligation is on the "owner" of the VMI to set the initial firewall policy – it is considered good practice to close all ports for incoming connections, although port 22 (ssh) is sometimes left open for administrative purposes (or could be opened in the contextualisation), and other ports may need to be opened to support the hosting platform (*e.g.* the Sunstone port for OpenNebula). The main issue is, however, that it will be tempting for users to upload images which are not firewalled and which have preloaded keys in them (indeed, in the case of Contrail's OpenNebula resources, it will be necessary to preload an ssh key for root, or for a user who can `su` to root). In the case of Amazon, firewalls are managed in the hosting environments, so the setting inside the image is less critical. In the case of our OpenNebula platform, we could make use of the hypervisor and libvirt to firewall the instances but tend not to do that as it introduces extra complexity in the deployment.

So what is presently missing is:

- A means of testing VMIs (or pre-start VMs) for firewall settings;
- A means of loading ssh keys into VMs during the contextualisation;
- Additional network monitoring at the platform level (OpenNebula).

3.3.5 Virtual machine images repository

VMI metadata, discovery, and marketplace In general, many end users would be put off creating their own images to make use of IaaS cloud services – creating an image takes a bit of effort and they may need to run special Contrail components such as the VIN agent. Instead, it is easier if users can locate pre-defined images which are already loaded with the applications that they need – they will just need to obtain the data via GAFS, or upload to the image. For this to work, there must be a repository which stores the VMIs, and it must satisfy the following which we discuss briefly:

- Image integrity must be preserved – a malicious modification by an attacker, or even unintentional but unauthorised change, must be discovered by the checking system prior to launching the VM.

- Endorsement – how would the user know whether the VMI is "safe." One model used is that an endorser will check the image – it may or may not be the same person who created the image – and, using a digital signature, or some similar means, will assert to the safety of the image. Note that no single person can ever check a full operating system or all the applications and intricacies. Instead, the builder will normally build from a known, trusted, base, and add only known and trusted applications (including, of course, Contrail agents for VIN and, if necessary, monitoring...)

- Patching – when users start running an image, it is important that it is patched, particularly when VMs update their patches over the Internet and therefore need to be connected to the Internet when they are booted up. An image which is several months old could probably be updated – there would be a short at-risk period after it is booted up and until the patches have all been applied (and, if necessary, the system has been rebooted), and there are examples of systems being compromised within such a short (*e.g.* half hour) time window. It would seem prudent to occasionally update an image with patches, and then re-sign any digital signatures. Indeed, one might consider an image unsafe even if the digital signature is valid, if it was made sufficiently long time ago (say, six months).

- Virtual machines could be monitored when they are started up, to check for malicious code – some commercial cloud providers monitor networks for unauthorised activities – *e.g.* DDoS attacks, port scanning, known exploits. As a corollary of this, one cannot in general run active security tests on these systems without violating the Acceptable Use Policy (AUP) (and most likely trigger an automated response system.)

- As mentioned above, one should be very cautious with predefined accounts in VMIs. An ssh key is not in itself harmful – it is a public key with a presumably well protected private key somewhere, but then the ssh connection is usable only by the person with the ssh private key. Instead, image managers resort to building VMIs with a known, and often insecure, root password (such as "password") – obviously this is highly unsafe, particularly for systems connected to the Internet, and the account needs resetting as soon as possible.

Note that, at present, much of this falls outside the scope of Contrail itself, but will need to be taken into account in an infrastructure making use of the Contrail

code.

3.4 IaaS providers and platforms

3.4.1 OpenNebula

OpenNebula is the platform of choice for Contrail (and for many other projects); it provides a mature platform for IaaS which is easy to set up and configure, yet has powerful features. OpenNebula orchestrates storage, network, virtualization, monitoring and security technologies to deploy Cloud services by means of virtual machines settled in a distributed infrastructure. In fact, the idea of OpenNebula is a pure private cloud, in which users actually log into the head node to access all Cloud services, and therefore OpenNebula has a great level of centralization. However, there is a huge amount of customizability that is permitted. For instance, users are free to choose among available hypervisors, e.g. KVM, Xen or VMware.

It provides the following security features:

- OpenNebula comes with a native authentication module and also it provides the ability to use an external module that takes care of these duties. Authentication module has support for user/password and rsa private/public key authentication. Authentication module can be easily modified or completely replaced by a third-party authentication module.

- To access Cloud services, all requests should be authorized. The OpenNebula admin (oneadmin) can perform any operation on any object (VM, network, host or user) while regular user is created by oneadmin and is able to manage only the users own objects (VMs and networks).

3.4.2 OpenStack

OpenStack is the other platform currently supported by Contrail. From the security point of view, two components are crucial: the authentication manager and the cloud controller. An authentication manager is one of the compute components and provides authentication and authorization services. A cloud controller uses several protocols to communicate between different *nova compute storage* components.

The Contrail project has integrated the IdP mechanism with OpenStackÕs Keystone for federated identity management extension. Keystone is the OpenStack project that provides Identity services among others.

3.4.3 Amazon

Amazon provides a controlled services environment with monitoring and group controls of firewalls of virtual machine (*i.e.* they are controlled in the hosting environment rather than from within the VM.) Before using Amazon Web Services (AWSs), Cloud users should sing-in to Amazon secure web site and generates security credentials. Password-based and multi- factor authentication models are available for authentication.

The credential used for authentication depends on the type of API (types: REST, Query or SOAP). The REST and Query API use access keys as the credential for authentication and authorization. The SOAP APIs use X.509 certificates. To access Amazon EC2 instances (launch and terminate instances, change firewall parameters, etc.) the Cloud user has to use key pairs.

Access control within Amazon EC2 is provided on multiple levels. The goal is to ensure that data contained within Amazon EC2 cannot be intercepted by unauthorized systems or users and that Amazon EC2 instances themselves are as secure as possible. The Amazon administrators do not have any access rights to customer instances and cannot log into the guest OS. To avoid security breaches in multi-tenant setting, Amazon EC2 provides a complete firewall solution.

3.4.4 EGI

EGI is the European Grid Initiative, an umbrella organisation of national grid initiatives (NGIs) in Europe. It is relevant in this context because EGI runs a federated cloud task force (now activity). Based on StratusLab, EGI offers a marketplace of VMIs which have been endorsed and checksummed by an identified person.

3.5 Networks/VIN

The Virtual Infrastructure Network (VIN) is the component that is responsible for managing all communication within a Contrail application. An application running in a Contrail Cloud generally consists of multiple VMs, and these VMs generally need to communicate with each other, with the public Internet, and with the user.

The security of VIN is based on the use of certificates to establish the tunnels between end-points [19]. The certificates enable authentication of the end-points, i.e., the VIN agents. As the certificate is likely tied to a private IP address, it should really be identifying each individual host by IP address (or possibly hostname, if a DNS is available for the private network). This would allow VIN participants to verify that their peers have been properly enrolled in the VIN by the central VIN

controller, by matching the IP address in the VIN certificate to the source IP address of the peer. Tying the certificate to the host would add some protection that a stolen or compromised certificate would be more difficult to exploit by a malicious service, and it would allow hosts to be removed from the VIN by revoking their certificates. On the pragmatic side, one can serve VIN with normal certificates or set up a CA as-a-service for distributing such certificates. As described in Section 3.3 VEP sets up the certificates using the Oauth2 delegation code.

Contrail applications typically require one or more dedicated communication channels between the VMs participating in the application. These channels are implemented as Virtual Private Networks (VPNs), and if necessary they use tunneling and/or encryption. The VIN is responsible for the operation of these VPNs, i.e., for setting up, running, monitoring, dynamically changing, and tearing down the VPNs. A detailed description of the VIN component can be found in deliverable D4.3 [20].

The creation of a VIN requires a communication channel between the VIN Agent and VIN Central Controller at the bootstrapping stage. This is needed because they exchange information for setting up the VIN network. Security information can be passed and this requires the channel to be secured by means of a shared secret which the VIN Agent will use to authenticate with VIN Central Controller's OAuth Authorisation Service. The VEP will be used as trusted node to enable authentication of the VIN agent and thus obtain a certificate to be used to setup IPsec tunnels. The tunnelling feature (D4.2 [19], sections 3.1-3) is useful and can potentially provide strong security between all participants (once fully implemented), provided certificates are used to secure the endpoints, and a naming scheme in the certificates is used which is consistent with the (possibly temporary) names of the hosts and/or their IP addresses. VIN uses StrongSwan to establish IPSec tunnels.

3.6 ConPaaS

ConPaaS is deployed in an untrusted environment: a cloud with thousands of users sharing the same network is a serious security threat. In order to understand ConPaaS security architecture and how does ConPaaS provide isolation by components, we need to understand its architecture first. ConPaaS services consist of three main entities: front-end, the manager, and agents. Front-end handles user requests and is responsible for spawning initial service's VM, namely the manager. The manager is responsible for maintaining agents (VMs) comprising the service. An agent resides on each of the other VMs started by the manager. Therefore, a ConPaaS service consists of one manager and multiple agents.

In order to maintain isolation of all the components, ConPaaS provides its own solution for authentication of the components and secure networking. To secure

ConPaaS, control sequence between front-end, managers and agents are secured. All the commands sent between instances of ConPaaS (starting new VMs, gather information about the agents/ managers, etc.) are issued in isolation (secure communication using SSL between all instances). Moreover, each service needs to embed mechanisms enabling privacy and isolation from other infrastructure instances and outside world.

Here we describe mechanisms provided by ConPaaS services enabling internal security. Each service uses HTTPS protocol to send commands between the front-end and the VMs (agent/managers):

- ConPaaS restricts requests only from the front-end, or from other VMs owned by the same user sending requests from the front-end and the same user's service.
- ConPaaS encrypts all traffic exchanged between the user, front-end and the agents.

In order to achieve this, each VM receives a certificate that binds it to the ID of the user to which it pertains, to the service it is part of and to other specific parameters. With this certificate, the VM is able to send/receive commands to/from all the other VMs pertaining to the same user and the same service.

There are few assumptions made by the ConPaaS in order upper statements are true during the ConPaaS services execution.

- The cloud on which ConPaaS is deployed, is secure. Beyond that, all communication channels between the VMs are encrypted and both sides of the communication channels are authenticated (preventing MITM attacks).
- ConPaaS code is secure (the front-end's). User must trust the provider of the ConPaaS services. If the code is compromised, then services cannot be fully secure. In ConPaaS' security model, the manager plays a special role, because it is considered secure while the agents could become compromised by the application.

ConPaaS's framework is considered trusted. Service's security should be provided by the service itself and not by the ConPaaS. Applications inside ConPaaS are organized as collections of services. Applications are as trustful as services running the applications are.

Key/Value store Scalarix is a distributed transactional key/value store. It provides scalability and built-in fault tolerance by design. Scalarix provides access through HTTP and JSON interface. All external communications establish secure channels using X509 certificates provided by the application or ConPaaS itself, internal communications are also secured by using secure socket layer.

Task Farm (Bag of Tasks) The ConPaaS TaskFarm is the main component used for simulation, with Map/Reduce being important for analysis of large data volumes. The user needs to provide a list of independent tasks to be executed on the cloud. For this purpose XtreemFS service is used for data input/output. Additionally, a file system location needs to be provided where the tasks can read input data and/or write output data to it. TaskFarm service uses XtreemFS for data input/output. Moreover, the actual task code can also reside in the XtreemFS. The user can optionally provide an XtreemFS location which is then mounted on TaskFarm agents.

MapReduce The MapReduce service provides the Apache Hadoop framework in ConPaaS. The ConPaaS service provides front-end with useful links to the Hadoop namenode, job tracker and to the user interface where user has control over MapReduce jobs. As for all ConPaaS services is true, MapReduce is meant to be used from the user interface only and as soon one of the machine in the cluster is compromised, the security measurements are the same as for Apache Hadoop's framework.

Hadoop's security framework consists of:

- authentication by node validation and client validation (user)
- role based authorization of the users
- auditing system (logging)
- file encryption
- CA facility (key certificate server with central key management serving and managing different keys)
- secure communication between nodes, applications and interfaces

Web Hosting ConPaaS web hosting service supports PHP-based and servlet based applications. The default ConPaaS configuration creates strong sandboxing so that malicious applications can not open arbitrary sockets, access infrastructure's filesystem and can not execute arbitrary commands that could harm other applications/services. This makes the platform relatively secure against malicious applications.

MySQL and Galera service By default, database services bind to local interfaces provided by the service defined network.

ConPaaS Database Federation user database provided by the Contrail is now integrated with ConPaaS's database. ConPaaS provides federated log-in using

SAML2 protocol provided by the SimpleSAMLphp instance running as an SP. Using SAML request/response mechanism Contrail IdP provides ConPaaS Front-end with the Contrail Users's ID (UUID) and maps that ID to ConPaaS user. After federated user is mapped to ConPaaS user, from that point on, user is ordinary ConPaaS user with UUID attribute which is used to issue requests to the federation's API. Therefore, ConPaas provides other authentication mechanisms using Contrail IdP.

ConPaaS database itself is located on the same filesystem as the front-end runs therefore it is as secure as the ConPaaS infrastructure.

3.6.1 Other PaaS – platforms and providers

Some of the most well-known PaaS offerings are Amazon BeanStalk, Microsoft Azure and Salesforce Heroku (for a comparison with ConPaaS please refer to Deliverable D9.3 and D16.2 [21, 31]. As seen before from the case of ConPaaS, there strong trust is needed by the user towards the PaaS in order the user is willing to use the platform. It is evident that security within PaaS is multi-tiered and many of the underlying security features are outside of the customer's control.

There are some additional security considerations to take into account before customer start using a platform provider: certifications. SSAE 16 (formerly SAS 70) [5], PCI DSS or ISO 27001 [14] are often indicators of the provider's thorough security processes. Moreover, a customer should as whether the provider has completed assessments against common cloud security control models, like FedRAMP [2] or the CSA Cloud Controls Matrix [6], which are both emerging industry standards for cloud security industry.

3.7 XtreemFS/GAFS

The XtreemFS service provides POSIX compatible storage for ConPaaS. To ensure security in an untrusted, worldwide network, all network traffic can be encrypted with SSL connections, and users can be authenticated with X.509 certificates. An XtreemFS instance consists of multiple DIR, MRC and OSD servers. The OSDs contain the actual storage, while the DIR is a directory service and the MRC contains meta data.

By default, authentication in XtreemFS is based on local user names and depends on the trustworthiness of clients and networks. Additionally, for the authorization XtreemFS supports the standard UNIX permission model, which allows users for assigning individual access rights to file owners, owning groups and other users. Authentication and authorization are policy-based, which means that different models and mechanisms can be used to authenticate and authorize users.

To encrypt all network traffic, services and clients can establish SSL connections. However, using SSL requires that all users and services have valid X.509 certificates. Therefore, XtreemFS uses X509 certificates for authentication and authorization purposes. XtreemFS distinguishes between two types of certificates: user-certificates and client-certificates. User certificates are used for user authentication and authorization before mounting a volume by the user. Client certificates are used by the services in order to authenticate instances among each others. Both types of certificates can be used as administrator-certificate if it contains special attribute. Certificates are only valid for the service that was used to create them. It is not possible to mix SSL-enabled and non-SSL services in an XtreemFS installation.

3.8 Messaging

In Contrail messaging is being used mostly for monitoring and accounting components. The messaging infrastructure provides the publish-subscribe mechanism of message passing, providing asynchronous communication channel. Messaging is based on the RabbitMQ package, a well known AMQP implementation.There are two levels of messaging systems in Contrail. On the top level, the messaging system is installed on the federation between the federation nodes. On the bottom level, each provider has a separate provider messaging system. Connection between the federation and providers messaging systems is established by the Federation HUB component. The component on the federation can subscribe on a specific provider for specific messages.

Security aspects for messaging system:

- default Contrail installation uses system packages for RabbitMQ, which are protected by the username and password,
- username and password are used by the Contrail components and should not be visible to the *outsider*,
- RabbitMQ can be put under VPN for further security, however, this is domain for the cloud provider administrator to implement.

Security aspects for the Monitoring HUB:

- provides standard REST interface, protection does not differ from other Contrail components with REST API,
- cross-domain messages are sent over web sockets.

Potential threats:

- not securing the RabbitMQ channel gives access to all the messages passed between them,

- monitoring HUB uses Kestrel and Redis, which need to be secured as well.

3.9 Accounting

Accounting provides usage data for the applications and also for the infrastructure. Federation accounting is accessed from Federation Web by the user. On the other hand, Provider accounting is accessed only by the Federation components and is not intended for the human interaction. All access to the components is exposed through REST interface. The data is stored on the provider in the Accounting database. Communication between the components is performed over secured channel.

Security aspects:

- default configuration for database access is protected by standard username-password scenario, the database is as secure as MongoDB system package allows,
- REST API is protected with authorization and authentication constraints as other REST services,
- to access Federation accounting, user obtains user credentials through Federation Web login process,
- validness of the user OAuth token needs to be approved by the AS,
- communication between Contrail accounting services is secured with service certificates over secured and encrypted channel.

Potential threats:

- the data in the database is not encrypted,
- having proper access tokens and information about other instances, properly directed call to the Accounting service can be made to obtain information about usage of other users as well, however, to achieve this one has to obtain information about other users and instances first,
- if messaging security is breached, one can inject arbitrary data into the system about their running instances.

4 Isolation against threats

4.1 Protection against other federation users

In order to protect federation users from other federation users, we propose the authorization support which guarantees that only legitimate federation users are

able to access and exploit continuously some Contrail resources. The authorization support is non-by-passable and access rights are determined by a security policy that takes into account the attributes of users, resources, and environment. Besides regulating the access to resources, the authorization support revokes access rights and terminates the resources usage in case the policy is violated.

Our authorization support is based on the UCON model [28]. It introduces new features in the decision process w.r.t. traditional access control models, such as (i) *mutable attributes* of subjects and objects and, as a consequence, (ii) the *continuity of policy enforcement*. Mutable attributes describe features of subjects and objects that change due to the decision process, e.g., users' reputation and resources' workload. Mutable attributes leads to the need of continuously monitor their values, and re-evaluate the security policy to guarantee that the right of a subject to use the resource holds while the access is in progress.

This model can be successfully adopted in case of long-standing accesses because the decision process is performed continuously during the access time. The *pre decision* phase corresponds to traditional access control, where the decision process is performed at the request time to produce the access decision. The *ongoing decision* phase, instead, is executed after the access is started and implements the continuity of control that is a specific feature of the UCON. Continuous control implies that policies are re-evaluated each time mutable attributes change their values. The UCON decision process evaluates *authorizations* (predicates over subject's and object's attributes), *conditions* (predicates over environmental or system status), and *obligations* (actions that must be performed along with the access). If the decision process detects a policy violation while an access is in progress, this access is revoked and resources are released.

We exploited the U-XACML policy language [9, 22] to encode UCON policies. The U-XACML extends the XACML language [25], the widely used access control language, with constructs for usage (continuous) control and attribute updates occurred as a result of the access. Here are examples of security policies regulating access to and usage of Contrail federation resources:

- GUEST users are allowed to execute their applications (`execute-app`) as long as the Cloud federation load is lower than a given threshold α.
- GUEST and BRONZE users are allowed to store their OVF files (`store-ovf`) as long as their reputation is greater than a given threshold β.

Below is the U-XACML policy which encodes these policies:

```
<Policy PolicyId="uxacml-policy"
 RuleCombiningAlgId="first-applicable">
  <!-- pre-authorization --!>
 <Target></Target>
  <Rule Effect="Permit" RuleId="fed:rule-1">
```

```
   <Target>
    <AnyOf><AllOf>
     <Match MatchId="string-equal">
      <AttributeValue DataType="string">GUEST</AttributeValue>
      <AttributeDesignator AttributeId="federation:subject:group"
        Category="subject-category:access-subject" Issuer="Federation">
      </AttributeDesignator>
     </Match>
    </AllOf></AnyOf>
    <AnyOf><AllOf>
     <Match MatchId="string-equal">
      <AttributeValue DataType="string">OVFAPPLICATION</AttributeValue>
      <AttributeDesignator AttributeId="resource:resource-id"
        Category="attribute-category:resource" Issuer="Federation">
      </AttributeDesignator>
     </Match>
    </AllOf></AnyOf>
    <AnyOf><AllOf>
     <Match MatchId="string-equal">
      <AttributeValue DataType="string">EXECUTE-APP</AttributeValue>
      <AttributeDesignator AttributeId="action:id"
        Category="attribute-category:action" Issuer="Federation">
      </AttributeDesignator>
     </Match>
    </AllOf></AnyOf>
   </Target>
   <!-- continuous control --!>
   <Condition DecisionTime="On">
    <Apply FunctionId="less-then">
     <Apply FunctionId="one-and-only">
      <AttributeValue>ALPHA</AttributeValue></Apply>
     <Apply FunctionId="one-and-only">
      <AttributeDesignator AttributeId="federation:load"
        Category="attribute-category:system" Issuer="Federation"/></Apply>
    </Apply>
   </Condition>
  </Rule>
</Policy>
```

4.1.1 Usage Control System Architecture

The Usage Control system architecture for the Cloud federation is shown in Figure 2. It extends the common authorization systems architecture [25] to deal with a continuous policy enforcement. The main components are described in the following. The *Policy Enforcement Point (PEP)* is integrated within the Application Execution and Management (*AEM*), that is the component of the Federation Core implementing the security relevant actions regulated by the security policy. The PEP must intercept these actions, and asks he Usage Control system to evaluate the security policy and enforce the resulting decision. The *Context Handler (CH)* is the front-end of the Usage Control system, that triggers the access decision process. The *Policy Information Point (PIP)* retrieves the mutable attributes needed to perform the access decisions process. The PIP contacts the Attribute Manager(s) (AMs), to obtain the fresh values of the attributes required for the decision process. The *Policy Administration Point (PAP)* stores and manages U-XACML

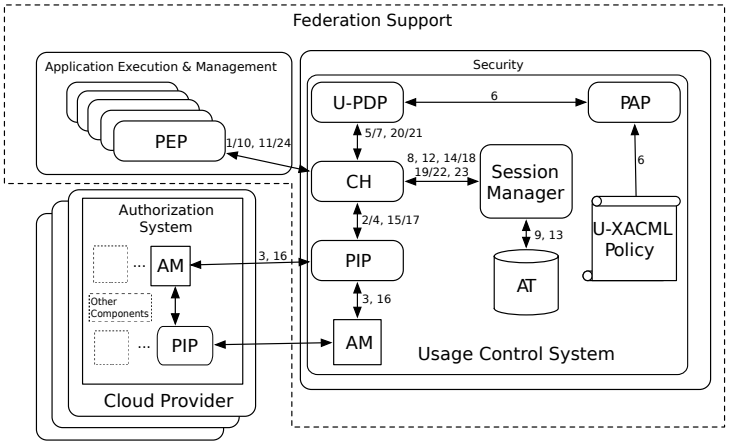

Figure 2: Integration of the UCON System within the Contrail Cloud Federation

policies. Conversely, the *Policy Decision Point (U-PDP)* evaluates U-XACML policies to produce the decision for each access requests. The *Access Table (AT)* keeps meta-data regarding accesses in progress, i.e., usage sessions. It contains a table of the current sessions with their statuses (i.e., pending, active, ended, revoked), a table of IDs of the attributes needed to service each session, and a table that caches the latest values of these attributes. Finally, the *Session Manager (SM)* is the main novelty of the architecture, and it manages usage sessions. The SM creates a new entry in the AT for each new access request that is allowed by the U-PDP, i.e., for each usage session that starts. The SM also manages the ongoing decision phase. It performs the periodical retrieval of mutable attributes. When the values of some attributes change, the SM triggers the access re-evaluation of all usage sessions that exploit these attributes. In such sessions, where a decision turns to “deny" must be revoked. The SM repeats the attribute retrieval until there is at least one active session.

4.1.2 Authorization Workflow

Security relevant actions requested by users must be authorized before and during their execution. The workflow described in the following is general enough to be valid for any of the security relevant actions, which are explained in the next sections. The steps of the workflow authorization refer to Figure 2 and are described below:

Step 1 The authorization workflow starts when the Cloud federation receives a request from the user. The PEP embedded in the Federation Core intercepts invocations of security relevant actions. Then, the message *tryaccess* is sent by the PEP to the CH.

Step 2 The CH retrieves the values of the attributes that could be relevant to the decision process by sending the *attr query* message to the PIP.

Step 3 The PIP, in turn, contacts the relevant AMs exploiting their specific protocols.

Step 4 The PIP sends back these values to the CH through the message *attr value*.

Step 5 The CH then sends the access *request* that includes the attributes previously collected, to the U-PDP.

Step 6 The U-PDP loads the U-XACML policy (discussed in the next subsection) from the PAP.

Step 7 The U-PDP evaluates the policy and replies with the *response* to the CH.

Step 8–9 For the sake of clarity, let us suppose the policy permits the execution of the requested action. Then, the CH sends the *create entry* message to the SM for creating a new entry that represents the new usage session in the AT.

Step 10 The *create entry* message contains attribute updates which should be performed by the SM before the usage session starts. After, the CH replies with the *permitaccess* message to the PEP.

Step 11 When the access has begun (e.g., a user started an application), the PEP sends the *startaccess* message to the CH.

Step 12 The CH sends the message *update entry* to the SM.

Step 13 The SM contacts the AT to change the status field of the database entry related to this usage session from *pending* to *active*, and triggers the evaluation of the ongoing access for the first time.

Step 14–18 The SM starts the continuous policy re-evaluation loop and sends the *attr query* message through the CH to the PIP to get the fresh values of attributes that are relevant for this access.

Step 19 If one of the collected values differs from the one cached in the AT, the SM contacts the CH sending the *policy reevaluation* message.

Step 20–21 The CH translates the message in the right format and sends the *request* message to the U-PDP that performs the re-evaluation of the access right.

Step 22 If the decision included in the *response* message is *permit*, then the CH forwards this answer to the SM.

Step 23 The SM performs ongoing attribute updates contained in the U-PDP's reply and continues the policy enforcement by collecting fresh attributes and triggering the access re-evaluation (steps 13–22). Instead, if the content of the *response* message sent by the U-PDP is *deny*, the SM sends the *re-*

vokeaccess message. This includes the data to identify the right PEP to the CH.

Step 24 The CH forwards it to this PEP forcing the access revocation.

4.1.3 Protection Against Race Conditions

The authorization system in Contrail exploits mutable security attributes for evaluation of access and usage rights. It might happen that several ongoing access decision processes read and update concurrently the same set of mutable attributes. For instance, this could happen when multiple Usage Control system are adopted, e.g., to speed up the policy evaluation phase. A specific kind of race condition might appear if an access decision process checks authorization predicates when another process acts on attributes involved in the former process.

For example, the policy-1 allows the creation of a new virtual machine (VM) only if the total number of VMs belonging to a user s is less than X. Let us suppose that s tries to create two distinct VMs at the same time. This triggers two policy decision processes. Each policy decision process follows these operations: *1)* retrieves the total number of VMs n belonging to s; *2)* compares the value of the attribute n with X; and *3)* if the current value of the attribute n is less than X, increases the attribute n by 1 and allows creation of a new VM.

A concurrency problem, called *race condition*, arises when the resulting access decisions depend on interleaving of operations performed by these decision processes. *Lost update* and *inconsistent retrieval* are examples of the race condition problem. Lost update happens when all decision processes read the old value of the attribute simultaneously while decide on and update the new attribute value independently. Assume that the second decision process retrieves the value of the same attribute after the first process performed the operation 1 but before the execution of the operation 3. As a matter of fact, if the current value of the total number of data objects belonging to s is N-1, both decision processes will allow the creation of new data objects, while only one data object can be created according to the security policy. Inconsistent retrieval occurs when a policy decision process retrieves one attribute before another decision process updates it and reads another attribute after the same decision process has updated it.

The authorization system in Contrail adopts concurrency control mechanisms from the theory of databases for avoiding race conditions. It is the responsibility of security components of the federation (or providers, if the access to provider-level resources should be authorized) to control the simultaneous usage of security attributes. A user, i.e., the requester, is not involved in the concurrency control explicitly. However, without such control in the place the user might cause a policy violation by issuing access requests which require attributes that interfere with other requests.

The main element of the concurrency control is a *transaction*, a number of operations that read and modify a shared resource with other transactions. Each transaction begins with the $start$ operation which is followed by a number of arbitrary operations op which change the shared resources. The transaction ends with the $commit$ operation when everything went well, or $abort$ if opposite.

In our scenario, an attribute constitutes a shared resources, the policy decision process represents the transaction, op corresponds to the attribute retrieval and update. We assume that the policy enforcements ends with $commit$ even in the case the access is denied and no updates are required, or with $abort$ in the case of errors.

A *serializable* execution of concurrent transactions guarantees that there is no interference between transactions and it produces the same effect as some sequential, i.e., one by one, execution of theses transactions. The two-phase locking protocol (2PL) is the solution we choose to address concurrency issues in attribute reads and updates.

To avoid concurrency issues, each attribute is paired with a lock, and any operation on the attribute value requires acquisition of the lock on this attribute. When the decision process requires to read and update one attribute or more, all these attributes are locked before performing the operations. If the attribute is already locked, i.e., there is another concurrent decision process (or more) that involves the same attribute, the decision process is suspended until the lock is released by the other decision process. As soon as the decision process has been performed, the involved attributes can be unlocked.

To avoid deadlocks, the policy decision process must lock attributes in a predefined linear order. The order of unlocking is arbitrary. We can assume that is possible to define the attribute ordering because the set of attributes that are exploited in the security policies is known a priori because the Policy Information Points must be statically configured to be able to retrieve these attributes. Subjects and objects are ordered based on the id. If the policy requires same attributes of different principals the attributes are ordered based on the holders' ids.

4.2 Traceability

Traceability refers to the ability to trace the user who originally authenticated using information available from a log file. There are several cases where one would want to trace a user back to their original identity:

- Where users have misused the system unintentionally, and need to be warned that their services are being suspended – the main issue here is really with contacting the user.

- Where users are billed for the services they use. In this case, it is essential that users are not billed for services that someone else consumed, and it is highly desirable that users are billed for the services they actually used. In the latter case, there must be non-repudiation (that the user cannot claim not to have used the services), and reliable identification (that the user cannot claim that it is some other user with, say, the same name.)
- Where users have had their credentials compromised, and action need taking immediately.
- Where users have (intentionally) misused the service, and action – possibly involving legal action – needs to be taken. In this case, the important distinction is that the misuse or breach of licence is sufficiently severe that IdPs are forced to release data to comply with the law, namely to investigate a potentially criminal case.

We basically have two distinct use cases above: contacting the user, and tracing the user to their original identity (possibly involving legal action). The general process for contacting the user in Contrail depends on the email address published as an attribute. The email address can be configured to be carried as a user attribute in the credential, from where it can be extracted, by the Provider, in case it is needed. In other words, Contrail supports the use case, but it needs to be configured. In general, it is expected that the central federation database will retain email addresses for the registered users, so if the address is not published, it could be obtained via the federation. So the process for providing contact data for users requires:

- Users provide email address at registration time, or the IdP publishes it (in which case the multi server updates the database).
- The User CA is configured to add the email address to the attribute assertion which is passed along to the Provider in the user certificate.
- The Provider (or other service) extracts the attribute assertion (which they would anyway for authorisation decisions), and records the email address in case the user needs contacting.

In contrast, if the user needs tracing to their original identity, a stronger evidence chain is likely to be required. Let us consider the (worst) case where a user has misused the system, and the logs are consulted to find out who it was:

- The Provider admin consults the log file and notes that an illegal action was taken by a user with a given distinguished name (DN).
- The Provider admin consults the federation admin to find data about the user in question: in particular the federation admin can consult multi server

logs to find the originating identity provider (IdP), and any additional information necessary to uniquely identify the original user. Depending on the type of IdP this can be a session id, a principal, or a targeted id.

- Here, either the federation admin must pursue the investigation, or the Provider admin will obtain the information and pursue the investigation directly with the IdP.
  The latter may seem like the better choice, but it should be noted that in an external identity federation, the agreement between the IdP and the Contrail multi server (which appears as an SP in this federation) may prohibit the Contrail federation from sharing information with third parties, in which case the federation admin must proceed with the investigation.
  Indeed, one might consider the Contrail Providers as "extensions" of the multi server as a Service Provider in the identity federation. However, this approach is complicated by the fact that the Provider may be located in a different country, and would not automatically be considered as a part of the agreement.

The purpose of this section is really to emphasise that credentials that offer traceability potentially offer a higher level of assurance, and they define scenarios which must be taken into account when deploying a Contrail federation. The management of personal data in an investigation of misuse must be considered. Similarly, IdPs that offer traceability would generally be configured with a higher level of assurance in the federation database.

4.3 Using LoA

The Contrail authentication support provides to its users several way of authentication. As an example, users could exploit their Google accounts (that provide them an user names and a passwords) to authenticate themselves on the Contrail platform, or they can use the Contrail authentication support based on PKI (i.e., the certificate provided by the Contrail user CA). The security level of each way of authentication is represented by a number (from 0 to 3), where higher values represent more secure authentication means.

The value of the "Minimum Level of Authentication" (MinimumLoA) QoP term paired to a user defines the minimum level of security that the authentication mean, exploited by the user to access the Contrail platform, must have. This value is defined at negotiation time, and the Contrail security support must check that, at application execution time, the user exploits only those authentication means whose security level is equal or greater than the MinimumLoA.

In order to enforce the MinimumLoA QoP term, we defined a solution based on the existing authentication and authorization supports. However, this solution

is quite general, and could be exploited for the enforcement of other QoP terms.

In the proposed solution, the QoP term MinimumLoA is represented by an attribute of the user. The value to this attribute is assigned at negotiation time, as soon as the user and the SLA support have agreed on the QoP terms. The MinimumLoA attribute is stored in the Federation Data Base, where other users' attributes are stored.

Another user attribute is defined, CurrentLoA, that represents the security level of the authentication mean that the user exploited to enter in the Contrail platform. The value to this attribute is assigned when the user authenticates himself on the Contrail platform. This attribute is embedded in the user certificate that is produced at login time (in the same way as the other immutable attributes, such as group and role). In this way, if the same user logs in twice, for each of the two accesses a distinct certificate is produced, and this certificate embeds the value of the attribute CurrentLoA that refers to the authentication mean that has been exploited for that access (in other words, there is an instance of the attribute CurrentLoA for each access). The user certificate travels across the Contrail components along with the user request (e.g. create a new application) it refers to.

During the authorization phase, the PEP extracts from the certificate all the user attributes (as usual), including the CurrentLoA one, and send them to the PDP. The security policy that is evaluated by the PDP includes a rule stating that the value of the CurrentLoA attribute is equal or greater than the value of the MinimumLoA attribute:

```
<xacml3:Policy
 PolicyId="level_of_authentication"
 RuleCombiningAlgId="first-applicable" Version="1.0">
 <xacml3:Description />
 <xacml3:Target />
   <xacml3:Rule Effect="Permit"
     RuleId="level_of_authentication.rule_1">
       <xacml3:Description />
       <xacml3:Target />
       <xacml3:Condition>
         <xacml3:Apply FunctionId="function:any-of-any">
           <xacml3:Function
                   FunctionId="function:integer-greater-than-or-equal"/>
             <xacml3:AttributeDesignator
               AttributeId="urn:contrail:names:qos:subject:currentLoA"
               DataType="http://www.w3.org/2001/XMLSchema#integer"
               Category="subject-category:access-subject"
             />
             <xacml3:AttributeDesignator
               AttributeId="urn:contrail:names:qos:resource:minumumLoA"
               DataType="http://www.w3.org/2001/XMLSchema#integer"
               Category="attribute-category:resource"
             />
         </xacml3:Apply>
       </xacml3:Condition>
   </xacml3:Rule>
   <xacml3:Rule Effect="Deny"
     RuleId="level_of_authentication.rule_2">
```

```
        <xacml3:Description />
        <xacml3:Target />
    </xacml3:Rule>
</xacml3:Policy>
```

The PDP retrieves the current value of the MinimumLoA attribute through the PIP from the Federation DB, and it exploits the two attributes in the decision process. This solution exploits functionalities that are already implemented by the Contrail and does not require ad-hoc modification.

4.4 Acceptable Use Policy

The Acceptable Use Policy (AUP) is a policy which defines the acceptable and unacceptable behaviour when using resources such as networks, or computing resources. It tends to be implemented in one of three ways:

- End users are deemed to have accepted the AUP by using the resource – like a privacy policy – possibly there is some text in small print in a corner saying "by using this resource, you agree to this AUP" and then provide a link to the text. The advantage of this approach is that it is easy and many users would expect that there are some policies associated with the use of the service and may investigate if they are in doubt. However, the user might later deny having seen it.
- Organisations ensure that "their" users are made aware of AUPs. If an organisation signs up they typically make an agreement with the Provider – or more likely the federation – that their users will "behave." It is then up to the organisation to ensure that their users are made aware of the policy. As organisations already have IT security policies, this is not too onerous.
- Users must accept an AUP before using the service: they must review a statement and tick a box saying "yes, I agree to these terms."

Ideally, Contrail would support all of these. What are the requirements associated with this? In the first case, there should be a link to the provider policies – or, better still, a single AUP for the whole federation which all the providers would accept. This is easy enough to do with the present infrastructure: the portal just needs a link to the policy and maybe some text saying there is one.

In the second case, the provider needs to have a means to verify organisational membership. This can be done with the Shibboleth and OpenID IdPs, provided that they are run by organisations and are configured to only authenticate people who are members of those organisations. Contrail will need to be configured to record the organisational membership attribute (*e.g.*, *eduPersonScopedAffiliation*) and to publish it to the Providers.

In the final case, we will need an attribute in the database saying the user has accepted a certain policy – this attribute will need to be published to the providers which need it, or perhaps the federation doesn't "work" at all till the user has accepted the policy.

ConSec supports the first case trivially: one would merely need to add some text to the portal to indicate the acceptable use of the service (at a federation level, assuming a common AUP is sufficient for the whole federation – this is generally one of the features of federations.) The second case, where organisational IdPs assert the user's acceptance of AUPs, is also trivially supported in ConSec, by only trusting, or by assigning a higher LoA to, IdPs that are members of federations that enforce suitable AUPs, such as national Shibboleth federations.

The third case, where an AUP attribute needs to be asserted, is supported by filtering SLAs according to whether the attribute is present, or by checking for its presence in the Provider access control. At present there is no means for users to have the attribute set other than by the federation administrator, or by an external IdP. It would be fairly simple to add this to ConSec, so roles other than federation administrator could manage these attributes – see the following section.

4.5 Attributes and community membership

It is often the case for scientific collaborations (UC3) that users collaborate in groups, or on proposals. In these groups (in the grid world, known as "virtual organisations"), there are typically a number of administrators who manage the memberships and roles of the groups (as opposed to federation administrators or provider administrators). The responsibilities of the community administrator(s) are:

- To manage membership of the community: who joins, who leaves.
- Community roles: what are the roles and what are they allowed to do. As far as the federation is concerned, the name just needs to be unique so it doesn't clash with those of another community (*e.g.* a URN), because in this case users may have roles assigned by one community which is misinterpreted by a resource as belonging to another.
- Defining the community AUP.
- Collaborate with investigators of incidents or breaches of AUP which involve community resources or members.

As regards the Contrail release, there must be support for non-federation roles, and there must be defined community administrator roles and access control policies in the federation must be defined to allow people with those roles to grant

and revoke the relevant attributes (community membership, community roles) to others. This additional support code is being implemented as of Jan. 2014.

4.6 Protection and Isolation in the Cloud

One of the barriers to uptake of cloud resources is the trust issue – how can I feel that my stuff is safe if it is "out there" in the cloud? These and other security questions are relevant to the use of clouds, and depend on the security features of the cloud provider. It would not be appropriate to incorporate a full analysis of the cloud security features from a large number of providers; instead we choose to focus on the issues that are exacerbated or ameliorated by Contrail: what are the potential disadvantages and advantages of having Contrail manage your cloud access?

To investigate this (and the technical details can be found in the appendices of the present document), we need a short review of the general isolation issues associated with using cloud resources.

4.6.1 Protection against other tenants

One of the features of clouds is the case for multitenancy. As with any cloud, or indeed any shared resources, multitenancy can give rise to availability issues. Contrail seeks to address these issues at least in part by allowing the allocation of resources that are co-hosted (assuming the provider supports it) – while contention could still arise, it is now largely under the control of a single tenant.

4.6.2 Protection against external attackers

Any place on the Internet is likely to get under "attack" from attackers, be they script kiddies or a large scale resourceful attack from a competitor[6]

4.6.3 Malware, Intrusion and intrusion detection

A virtualisation environment has the capability, if so constructed, to help protect against malware, as does network monitoring. For example, some cloud providers offer firewall controls in their hypervisors, and they will monitor networks for unauthorised activity. In Contrail, we do not offer services to this effect – users

[6]We disregard here the case where the "attacker" is a government, *cf.* the Snowden revelations, as such an attacker will be "inside" the cloud and almost impossible to protect against. In the context of Contrail's use cases, the presence of governments as a likely scenario is not considered a threat *per se* except that it may affect the overall trust in public clouds, particularly data centres based within the US or the UK.

are responsible for firewalling and protecting their own infrastructure. What we do offer is the ability to advertise the capabilities in the SLA negotiation. A provider might charge more for monitored services, and using the Contrail SLA negotiation mechanism, users would be able to make an informed choice[7]. Likewise, inside a VM, added security services could offer extra features. Like the VIN agents inside the VM, additional services could be offered to users by customising the VMIs. Similarly, an intrusion detection system could be offered. Some cloud providers offer services to automatically patch the OS inside the VMs, and it is but a small leap to also offer security services like actively checking for intrusions and modified binaries within the host. As with the external services, such offerings could be expressed in the SLA negotiation.

4.6.4 Phishing

Phishing is the special case where users are prompted for their existing password and they enter it, into a form which is controlled by an attacker. In the case of Contrail, such an attack is more likely than in a normal cloud because we rely on (external) federated identities: if users are "trained" to enter their "home" password into any form that asks for it, users are more likely to have it stolen, and the impact of the theft is bigger because the federated identity is, by definition, used across a whole federation.

However, the identity provider will usually be aware of the risks and make users aware of the dangers of phishing, as it is also in their interest to prevent loss of passwords and unauthorised use of the account. In this sense, the assurance associated with federated identities is generally higher than with targeted identities[8]. Overall, it is our judgment that the use of federated identities, when done securely, enhances the usability and security of the infrastructure.

A related attack is the cross-site scripting attack. Contrail guards against these by having the infrastructure components mutually authenticate with X.509 certificates from trusted (pre-defined) certification authorities; and access to user certificates is only when authorised via OAuth, and requires, in addition, a pre-shared secret (namely, the OAuth client id.)

4.6.5 Sniffing and other passive attacks

The use of mutual authentication between infrastructure hosts helps guard against loss of information in the infrastructure, and each virtual machine (VM) is treated

[7] Of course, in this case users might choose to run the cheaper, less secure, service, unless there were penalties for running an insecure system, or for an intrusion.

[8] Also for other reasons – that users are less likely to share, or forget, the credential in general, and the fact that usually federations will require policies applied to IdPs if they aren't already.

as an independent entity. In Contrail, it would be perfectly easy to place trusted host certificates onto the VM: normally the difficulty is that the federation doesn't know the hostname to issue certificates to, and the platform isn't trusted (or indeed designed) to create host certificates. However, Contrail provides the means to use delegation to allow the contextualisation process to "call back" to the federation with a Certificate Signing Request with the appropriate hostname: a combination of the trust in VEP and the delegation mechanism allowed for the generation of host certificates. We decided against this, though, since VIN, when fully secured, would provide fully adequate protection of the host, via the VIN agents. The delegation/contextualisation process is now used only for the delegated user certificate – which in turn us used to securely access GAFS.

Contrail provides the means to protect the control and data messages on the wire – certificates connect the infrastructure, as in grids, between "static" nodes in the infrastructure, and the dynamically created resources are in turn protected with delegated credentials.

So far the design and architecture. In the implementation, we note that VIN security hasn't been fully implemented, so it is not at present providing the level of protection we would expect. The impact of this can be lessened by using the delegated user certificate to secure communications, but this would be left to the user – and would probably also require of the federation a means of securely distributing the User-CA certificate. Also, certificates in the test infrastructure tend to be generated ad-hoc and not by trusted CAs: of course, in a production infrastructure, particularly one distributed across sites, trusted CAs (such as commercial CAs or IGTF, or other Terena TACAR) should be used.

For further details of ConPaaS components and their security, please refer to section 3.6

4.6.6 Protection against PaaS and IaaS resource system administrators

In a Linux system, it is possible to have not just encrypted partitions but also encrypted `root` partitions (*i.e.*, not just the "home" directories but also encrypt the operating system). Such a system would offer a good protection against malicious system administrators, at the cost of needing activation remotely when the VM boots up. Similarly, if only encryption of the home partition were used, it would need unlocking before us. In Contrail, we have taken another approach, and offering GAFS as a secure means of providing data to the VMs, outside the reach of the resource system administrators. Of course, as everything passes through the hypervisor, a truly malicious cloud resource provider could retain a lot of control over the customer's resource, but the case is more with individual system administrator (cf. the CSA "malicious insider" threat.) While this does not offer perfect security – an attacker could suspend or page out the VM and inspect its memory

– it still offers pretty good security, and it does not need unlocking at boot time because the contextualisation installs a delegated user credential. . . but the credential is stored in the image and could be stolen by a knowledgeable administrator inspecting the image – we can offer levels of protection but not perfect protection.

There are cases where secure clouds have been built by using Trusted Platform Module, or TPM, a security device found on some motherboards[9]. The price for this is to give up some elasticity: the "cloud" is locked to machines with pre-registered TPMs. There is other work (non-cloud) using TPMs to secure virtual machines, *e.g.* from Royal Holloway's information security group[3]. As ever, there are compromises when implementing security. What Contrail has demonstrated is that security features can be advertised up front, and users can choose.

4.6.7 Protection against federation administrators

Federation administrators have little control on the resources on the infrastructure. However they are the means for a user to access IaaS services and they need to be trusteed for negotiating the deployment of the application on behalf of the user. Little protection mechanisms from the security point of view can be enforced for controlling federation administrators for misusing resources on behalf of the user. Contrail implements monitoring and accounting mechanisms so that a user can verify.

Federation administrators also need to be trusted for credentials the user might want to store to access either shared volumes or public clouds in the case of cloud bursting.

4.6.8 Protection against infrastructure system administrators

Clouds in general make the protection against IaaS administrators harder: it is quite easy to access a disk on a VM and make a copy of it, using IaaS management tools. Indeed, this is a feature, when the VM is able to migrate,

4.6.9 Other threats – User Choice

We have touched upon, in sections 4.6.3 and 4.6.6 the advantages of the ability to advertise security features. How do users know what to choose? Most "normal" end users do not understand security details, but know that they are needed. To this end, two things help: either we make use of external certifications of security (such as data centre security: Microsoft have SAS70 type II; Amazon claim they are aiming to get it), or we introduce helpful security classifications ourselves,

[9] *E.g.*, http://www.oerc.ox.ac.uk/projects/mytrustedcloud

to aid the selection – these would be fixed within the federation but would not necessarily be meaningful outside the federation.

A classification could look like this:

- High – operating systems are patched by the cloud provider, there are active intrusion detections and VIN built into the VM; the hypervisor implements its own firewall and hard disk encryption mechanisms under the control of the customer, and networks are actively monitored for unauthorised activity.
- Medium-high – default VMs come up with a high security level but measures can be disabled by the customer.
- Medium-low – default VMs come up with no additional security features (other than those built into the guest OS), but they can be added by the customer;
- Low – only security features in the guest OS itself are available.

Such a scale may be a bit misleading as it is perfectly possible to run a very secure infrastructure in a "Low" IaaS, by implementing the security in the VMs. For users who are not security experts, and for those with sufficiently small collaborations that they cannot afford to have three students work six months on implementing the security, there is an opportunity for cloud providers to offer extra security features.

5 Security evaluation and assessment

In this section we evaluate and assess the security in Contrail. We first evaluate the security by feature with respect to usability and integration with existing platforms. Then, we present detailed performance results of the authorization UCON model. Finally, we conclude this section with an assessment based on general testing.

5.1 Evaluation by feature: security, usability, and performance

A IT security infrastructure can be very secure by design, it can demand multi-factor authentication, have logging, and controls and whatnot. However, if the effect of the security is that people bypass it by sharing credentials, by setting up back doors into the system, or by otherwise compromising the design, then all the work in setting up and running the strong security has been in vain. Security works best when it doesn't get too much in the way – it is sometimes visualised as a scale with usability at one end and security at the other. And, indeed, some

of the projects with the highest security requirements also tend to have users with the highest aversion to strong technical security controls up to a point.

The features affecting the usability of the system, that we identified in Contrail either as part of the design, or in experiences with the implementation and integration, are the following:

Lower the initial barrier to uptake – Contrail achieves this by accepting existing external identities that users already have, like Google accounts, or national Shibboleth (or other SAML) identities (the latter being very strong requirements for UC3).

Account generation For the initial user, accounts are generated (by the multi server) on the fly, at the initial registration, but, in general, without carrying any authorisations. Thus, a federation administrator can implement a policy whereby any authenticated user can access the Contrail-based federation – a default authorisation. Further rights could then be granted later by an administrator. The ability to get working straight away without having to wait for rights to be granted by an administrator is one of the key features of cloud resources.

Integration with external portals. This requirement mainly arose out of the need to deploy ConSec (Contrail Security component) within the EUDAT (European Data Infrastructure) project testbed [1], and the need to support rather diverse communities participating in EUDAT. While ConSec was designed to accept credentials the users already have, further requirements arose to integrate with existing user portals. Each community has its own set of portals, and EUDAT has (or will have) a portal as its main entry point. For the end user, it is best if they have a means of logging in once, and ideally see both their community features and their Contrail (resp. Eudat) features.

In collaboration with EUDAT, we designed a few different models, involving sharing components between different parts of the infrastructure – *e.g.* common database, using the community portal as an OAuth/CA client, or using the portal as a CA. Matters are further complicated by the Shibboleth IdPs where the main identity attribute is `eduPersonTargetedId`, so each of the portals – Contrail, EUDAT, community – would see a different identity for the user. Another complicating factor was the need for the communities themselves to contribute work to the integration effort, as they know their own portals better than we, and the people in the communities with the skills and abilities to do this tend to be few and to be heavily committed to a number of other related tasks. Thus, in the end we opted for a very simple integration model, invented by EUDAT, where users access both portals but the IdP remembers the login for a period of time. On the first redirection (accessing the first portal), users are asked to select their identity provider, but on subsequent accesses *even to a different portal*, there are enough cookies in the browser to automatically redirect and log the user in to the other portals. Needless to say, such a feature needs to be used with some caution to

avoid stealing credentials from the user, but the portals are generally trusted and identify themselves to the ConSec federation using not just an X.509 host certificate but also an OAuth client id (which works like a shared secret).
Performance The performance of the system is also likely to be a factor – if the system is slow to react, people will be unhappy. Most people are used to a few seconds delay in loading, as they would see anyway with social media (which use the same security technologies as Contrail). For certain parts of the infrastructure, we have explicit performance measurements (see 5.2 below; indeed, the implementation of XACML measured was chosen partly because it had better performance than other implementations we tested earlier); for other components the performance is easy to see because it is visible immediately on the portal – login, redirections, generation of certificate, SLA negotiations – are immediate, and on a level with what can be expected from any online service.

5.2 UCON performance measurements

This paragraph describes the performances achieved by the prototype of the authorization system for the Contrail Cloud Federation, realized according to the architecture described in Figure 2 of Section 4. In order to execute the performance evaluation, the Contrail Cloud Federation front-end and the authorization system were hosted in a VMWare VM with Linux Ubuntu 12.04 and Java 1.6 support. The VM was configured with 1 CPU (the native machine had Intel Xeon Processor, 3.20 GHz) and 4 GB memory.

The prototype has been implemented using stateful Web Services (WS) which allow to share attributes among concurrent sessions. We used the Apache Axis2 framework[10] for WS implementation and deployment. Access requests, responses, and attribute queries were encoded into messages compliant with the “SAML 2.0 profile of XACML" open standard. For the CH implementation we used OpenSAML2.0 extension[11], whilst the Sun XACML Engine[12] and WSO2 Balana XACML Engine[13] have been adopted to evaluate U-XACML policies. The information regarding usage sessions was stored by using MySQL [14].

First, we measured the overhead which occurs during the pre-authorization phase as a result of the access request construction, attributes retrieval, and evaluation of the policy against the access request. Without loss of generality, we considered a scenario in which there is a single provider which executes one application and the provider should authorize the request only once before the ex-

[10] http://axis.apache.org/axis2
[11] http://www.bccs.uib.no/~hakont/SAMLXACMLExtension
[12] http://sunxacml.sourceforge.net
[13] http://xacmlinfo.com/category/balana/
[14] http://www.mysql.com/

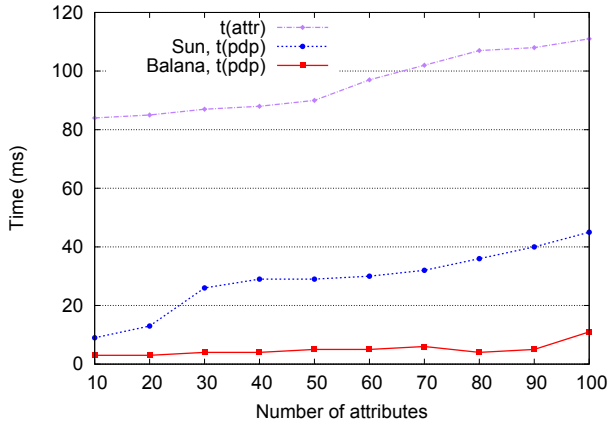

Figure 3: Overhead of Attribute Retrieval and Decision Process with Sun and WSO2 Balana XACML Engines

ecution. We varied the number of attributes which are required by the UCON system to perform the decision process. The overhead is measured as the sum of the following time intervals: t_{pepOut}, t_{attr}, t_{pdp}, t_{pepIn}.

The time t_{pepOut} is needed to build the SAML/XACML request by the PEP and to send it to the UCON system. The measured time was 10ms on average. The time t_{pepIn} is the time passed from the point when the UCON system is ready to send the SAML/XACML response until the PEP gets it and starts the enforcement of the access decision. In this case the measured time was 6ms on average. t_{pepOut} is bigger than t_{pepIn} because the size of the access request message is larger than the access response one. In fact, the former contains three attributes which describe a subject, an object, and an action while the latter contains only the access decision (permit or deny). The t_{attr} is the time which the UCON system spends for processing the access request, building a SAML Attribute Query, retrieving fresh attributes from PIP, and constructing the final XACML request. The time t_{pdp}, instead, is needed to evaluate the XACML request against the U-XACML policy and get the access response.

We noticed that the attribute retrieval contributes the most to the overhead and thus we plotted on the top line of Figure 3 the t_{attr} trend w.r.t. the number of attributes which the UCON system requests from the PIP before the access evaluation. As a matter of fact, there is a slight growth of t_{attr} with the number of attributes. Please note that in our tests attribute values were stored in a file, while in real systems an external AM could be exploited, possibly increasing the value of t_{attr}.

The middle line in Figure 3 shows how t_{pdp} depends on the number of attributes in the policy if the Sun XACML Engine is adopted for the evaluation of

U-XACML policies.

We used the number of attributes as the main parameter because our experiments shown that it impacts the most whereas the structure of the final policy (i.e., its complexity, a number of rules, a number of sub-policies combined for producing the access decision) is negligible. Indeed, t_{pdp} is almost the same for the individual policy which involves x attributes in one rule, or for the individual policy with x rules and each rule involves one attribute, or for the policy which is a combination of n rules or sub-policies where these x attributes are distributed evenly or randomly. However, in our tests we used simple types of attributes (strings and integers) and function on them.

There is a linear growth of t_{pdp} with the number of attributes. We used the Sun XACML Engine in our initial implementation of the UCON system. Then, we tried another engine, i.e., Balana, which is claimed by the developers to be more powerful comparing to the Sun engine. The bottom line in Figure 3 shows how t_{pdp} depends on the number of attributes in the policy if the Balana XACML Engine is exploited. Indeed, t_{pdp} drops drastically therefore in what follows we exploited the Balana as the primary engine for the policy evaluation.

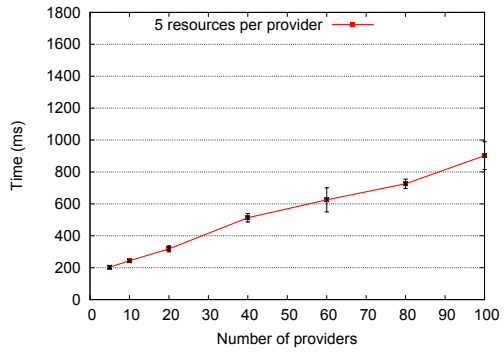

Figure 4: Time for revoking ongoing accesses, variable number of providers, 5 resources per provider

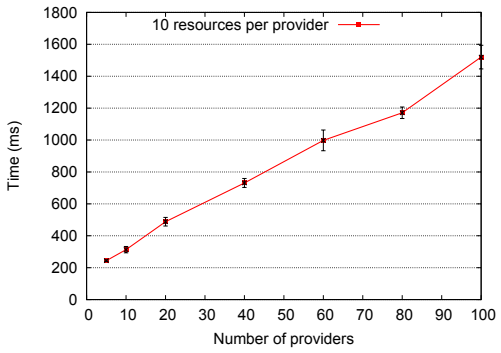

Figure 5: Time for revoking ongoing accesses, variable number of providers, 10 resources per provider

Second, we measured the scalability of the UCON system in the Cloud Federation of Contrail and considered the continuous control phase. We considered a scenario in which there are many providers, each provider executes many applications, i.e., resources, and each resource should be continuously controlled. Thus, the UCON system should service N concurrent sessions and $N = N_P * N_R$, where N_P is the number of providers and N_R is the number of resources per provider and we assume that resources are distributed uniformly among providers. Unlike the pre-decision phase, the security checks that implement the continuous control phase do not directly affect the performance of the Cloud Federation. In fact, the

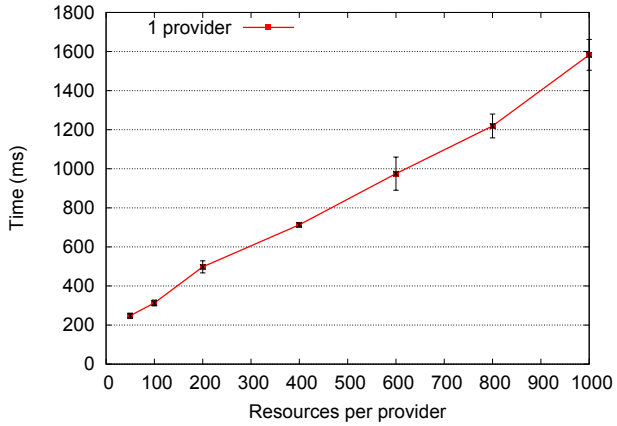

Figure 6: Time for revoking ongoing accesses with a single provider and a variable number of resources

UCON system could run on another machine rather than on machines that executes the access, i.e., the Cloud federation front-end and/or provider machines. The UCON system is responsible for the continuous attribute retrieval and policy re evaluation, so t_{attr} and t_{pdp} contribute mainly to the UCON system workload. From the prospective of the Cloud Federation, we are interested in the overhead which happens when attributes changed values and usage sessions must be revoked until the Cloud Federation starts the actual revocation. Indeed, there are inevitable delays due to the policy re-evaluation and delivery of the revoke access messages to corresponding PEPs by the UCON system.

We measured the overhead t_{revAll} which defines the time passed from the point when the PIP replied to the UCON system with the attribute value which violates the security policy until all sessions where this attribute is exploited have been re-evaluated and all PEPs have received the revoke access message. We used the same security policy for all sessions. The U-XACML policy was based on 5 mutable attributes.

We varied the number of concurrent usage sessions N from 25 to 1000. We considered different approaches for the load distribution in the Cloud Federation varying N_P and N_R. Let us consider 100 applications. All applications may be run by a single provider, or 5 providers may divide the load and every provider executes 20 applications, or each of 10 providers gets 10 applications. Figure 5 shows how t_{revAll} depends on N_P when $N_R = 10$. Figure 4 shows how t_{revAll} depends on N_P when $N_R = 5$. We see that t_{revAll} is moderate and increases with the growth of the number of concurrently running sessions N. When there are 100 providers and every provider runs 10 applications, i.e., the number of concurrent sessions is 1000, the UCON system takes on average 1520ms with 74ms of a

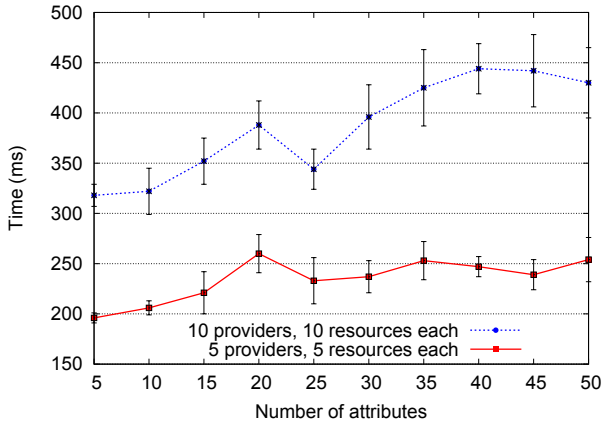

Figure 7: Time for revoking ongoing accesses with a variable number of attributes

standard deviation to get the attributes which violate the policy, to re-evaluate the policy, and to broadcast the revoke access message to all corresponding PEPs in the Cloud Federation. We noticed from the results that t_{revAll} does not depend on how the load in the Cloud Federation is distributed among providers. For instance, if there are 400 concurrent sessions t_{revAll} for $N_P = 80$ and $N_R = 5$ (726ms) is almost equal to t_{revAll} for $N_P = 40$ and $N_R = 10$ (731ms).

Then, we decided to measure how t_{revAll} changes if all applications are executed by one provider. Figure 6 shows how t_{revAll} depends on N_R when $N_P = 1$. We see that t_{revAll} is of the same range.

Finally, we measured how t_{revAll} depends on the number of attributes needed for the access evaluation. We varied the number of attributes from 5 to 50 and considered two configurations of the load distribution: (i) $N_P = 5$ and $N_R = 5$, (ii) $N_P = 10$ and $N_R = 10$. Figure 7 shows the results obtained. There is moderate growth of t_{revAll} with the number of attributes. In percentage terms, the growth is even for both configurations.

5.3 Assessment based on general testing

We have performed a penetration test using the W3AF vulnerability analysis tool, summarised in tables in Appendix A. The penetration test concentrated on a testbed system located at STFC. Refer to Appendix A for details of the testbed and the penetration tests that were performed.

No problems were observed regarding the integrity of the OAuth2, Shibboleth and OpenId protocols (see 5.4 for a more detailed discussion of the security of the OAuth implementations.) However, many vulnerabilities were found in the Contrail software. This project is a research project not intended for production

use directly. All of the vulnerabilities listed below could be eliminated by further programming work and some redesign of the functionality.

To illustrate the capabilities of the testing, here is a quick summary of the incidents found during testing.

- There is no audit-trail to log access so an intruder could use his genuine identity to connect to https via a genuine certificate trusted up to a known-good root CA without fear of his identity being revealed. See Section A.6 for a detailed analysis.
- The REST API is unfiltered and unprotected, leading to privilege escalation (observed – see Table 8 and Table 15), unauthorised access (observed – see Table 15), data injection (observed), script injection (suspected) and sql injection (suspected). This also occurs when the secure protocol https is used due to a lack of audit-trail. See Table 9 in Appendix A for details on data validation.
- Any use of the insecure protocol HTTP leads to major vulnerabilities, including a man-in-the-middle attack and trivial enumeration of users' confidential information.
- There is a lot of scope for misconfiguration by the systems administrator due to the complexity of the configuration files. This leads to problems when the insecure HTTP protocol is used in error. See Section A.6 and Table 5 for more details.
- Fingerprinting of the system was trivial and would lead to attacks targeting known vulnerabilities. The vulnerabilities that have been found would cause federations that are linked to this one to become vulnerable (Table 3)
- Automated attacks can proceed indefinitely and at full speed without detection (Table 10).
- A user account logging in via OpenId and sharing a browser session with another user on the same physical machine would be logged in as the other user, leading to credentials theft, impersonation and privilege escalation (Section A.6).
- A two-stage logout is confusing for the user and could lead to an incomplete logout from a session allowing someone sharing the physical machine to log in as the first user (Section A.6).
- A clickjacking vulnerability was found, leading to password theft (Section A.5 and Table 15)
- A bug means that OpenId sessions from more than one provider (i.e. Google versus Yahoo) become muddled, leading to privilege escalation and impersonation (Section A.6 and Table 6 and 7)

- There was a path disclosure vulnerability observed and a bypass of the login to the Tomcat management tool (See Table 15).
- A user can self-register without need to confirm his identity in any way and to get a limited login to the portal[15] They could use someone else's identity leading to an impersonation attack. But once logged-in they can also access his delegated certificate and uuid and this allows him to misuse the OAuth feature.
- There is no means of logically disabling an account, only to delete it. This would lead to a loss of data during the investigation of an intrusion.

Some of these have been fixed; some will need fixing in a production infrastructure.

Based on the security testing we feel to warn the reader and use the following message:

> WARNING - the system uses software that is intended for research purposes and is not suitable for production use. DO NOT expose this system outside of a safe environment. Internet access is exceptionally dangerous and is likely to result in a root compromise of the server and the entire network behind it.

5.4 Analysis of OAuth as Delegation Protocol

While OAuth itself provides good security, and has itself undergone extensive formal analysis ([26], [8]), in practice both implementations and practical deployments may have security vulnerabilities [30]:

- Access token eavesdropping.
- Theft of access token via cross site scripting
- Impersonation
- Session swapping
- Force login with cross-site request forgery

We summarise below the status of these vulnerabilities in a Contrail deployment.
Access token eavesdropping. In a "real" deployment of Contrail, every endpoint is protected with X.509 certificates from one or more trusted certification authorities. Like the OAuth client, it is static (as opposed to elastic), so does not have

[15] This is a feature; as self-registered users would not in general have any default authorisations, or would only have limited rights.

to change dynamically, and thus builds on established security models (PKI, systems administrators managing systems, federation administrators). Thus, the risk of exposure and impersonation is eliminated. (Indeed, in a fully configured Contrail security infrastructure, both client and server would authenticate and check each other's distinguished name, but in the case of OAuth we only need server side authentication and identification, as the client will be sending its client id which is effectively a shared secret.)

Another related consideration is if OAuth is used for authentication, *i.e.* as an IdP from outside of Contrail – this would be the case if we supported Facebook or ORCID. However, in the present ConSec deployment, external OAuth identity providers are not supported. It is more likely that we will be supporting OpenID Connect as a part of future exploitation of ConSec.

Theft of access token via cross site scripting. Contrail supports also the automatic grant – this is essential, or users will have to authorise every single delegation. In general, the type of workflow – such as deployment of an application – can be predefined, so the Authorisation Server knows which grants to approve. As it is somewhat difficult to define the full sequence of workflows in Contrail, we have chosen in ConSec instead to provide two features to control the delegation: (a) a predefined list of trusted clients, customisable for each user – so a user could, for example, choose to disallow delegations to a particular country or provider. And (b), to provide an auditing mechanism so the user can view the actual delegations after they have occurred. Together, these features provide a higher level of control for the end user than standard OAuth, yet is fully compatible with the OAuth protocol.

Impersonation. In [30], the authors highlight three weaknesses which can lead to impersonation attacks:

- The attacker can guess the victim's SSO credentials. In our case, the credentials are generated by the multi server, so the user id will have to be guessed or sniffed. Now, even an insider would only learn another user's identity if they see the user's credential (in particular, a federation administrator). However, in the default setup of Contrail (without added features to support collaborations between users), users are *isolated* from each other and will not see each other's username. Even if they did, though, they would still need a client id to obtain a credential (and it would have to be a *trusted* client id, set as trusted by the victim.) Uids are not guessable.

- Impersonation is also possible if the SSO credential can be reused. The reader will recall that the credential is always a SAML assertion issued by the multi-server. However, when the connection between multi and the authorisation server is encrypted with end to end encryption (*i.e.* mutual authentication) as in a production setup, SSL itself provides protection against

replay. (Alternatively, or in addition, nonces could be added to the SAML assertion along with a signature.)

- Finally, impersonation is possible if the resource doesn't check the originator of the authorisation request. But, again, resources would always use SSL in a production setup, and the distinguished name of the requestor is checked in every part of the OAuth protocol using a callback to the authorisation server.

Session swapping. Session swapping is an attack where the attacker intercepts credentials from a legitimate user, yet uses it for their own session by tricking the user's browser into relaying the attackers payload. In ConSec, we are not vulnerable to this attack because end users do not access OAuth directly: they authenticate to the authorisation server, but *all the clients and resources* are Contrail-internal and protected with X.509 host certificates. An attacker would need to obtain a certificate from a trusted CA to be able to make use of a stolen credential.

Force login with cross-site request forgery. The cross-site request forgery is one where an attacker presents a web page with malicious content inside to the user's browser. Again, Contrail's OAuth implementations are not vulnerable to this attack, because of the reasons listed aboive ("session swapping"). However, this attack could still be possible with the SAML assertion passed from the multi-server to the authorisation server. An attacker would send the victim malicious content, and, assuming that the victim is already authenticated, their browser would have to be tricked into accessing the Contrail portal/authorisation server. The usefulness of such an attack is limited, as only the authorisation server accepts credentials from the multi-server. It is thus essential in a practical deployment that the portal be protected with a "browser-friendly" certificate. While we can limit the scope of such an attack considerably, it would be difficult to guarantee that it cannot happen: it is a compromise between security and usability, built in to the OAuth protocol.

6 External factors

6.1 Operations

How is a federation operated? The federation is more than a software package; it also needs people to operate it. While this falls outside of Contrail, it is required to operate a production federation infrastructure.

Specifically, the following items need to be set up:

- Response to security incidents – *e.g.* untrusions at one of the participating

sites – in many cases it becomes necessary to warn the other participants, or even coordinate an incident response.

- Support – a common helpdesk supporting the user. Because the federation is in principle (with permissions) open to all the users, a common helpdesk which can assign tickets to individual sites is essential. Also necessary is a ticket handling system (or people) which is able to assess the tickets when they are reported, and assign them to the right software team, or the right site.

- Adding and removing sites. In particular, sites not living up to their obligations need to be dealt with.

- Defining policies and processes for updating the policies for the federation (see Section 6.2). In particular, for federations which span national boundaries, it is necessary to have policies which can satisfy data protection rules and policies in the relevant countries. Some of these policies can be implemented in UCON, but some would be defined at federation level (*e.g.* acceptable use), some at community level, and will have to be managed by having organisations, communities, and end users agreeing to the policies.

- A means of managing downtime: if a site needs to perform upgrades or network maintenance, it is sometimes necessary to take the site off the network, and if this is scheduled in advance, other sites and communities can schedule around it. In a federation, other sites can pick up work from a site which is in scheduled downtime, so the requirements for site availability is lessened, thus making operations less expensive for each site. However, the disruption is obviously less for users if the downtime is scheduled and other sites pick up the workload. Also, unscheduled downtime happens: network failure, power failure – for a short downtime, some work may be lost and will have to be resubmitted; for longer downtimes, other resource providers may wish to offer more resources to compensate.

6.2 Federation policies

A federation is generally not useful unless it defines policies and practices for its members. While generally outside the scope of the Contrail project, a *real* federation will need to clarify:

- Who can be a member;

- What are the requirements on an identity provider, and which attributes do they need to publish? What is the meaning of the attributes, and which schema defines them?

Feature	Purpose
ConSec external id	Unique identification of users, SSO
ConSec multi	Support for multi-LoA authentication
SLAs	Support for SPs advertising requirements and policies via QoP
LoA support	Support for service provisioning according to LoA
LoA support in QoP	Inaccessible services are filtered out beforehand.

Table 2: Features for Federation Policy Support

- What are the requirements on service providers ("Providers" in Contrail). Which attributes can they request, and how can they use them?
- Are there any other federation participants, *e.g.*, discovery mechanisms, and what are their roles? Who is the federation operator and what is their role?

All this is beyond the scope of the Contrail project. In this section, however, we need to describe *how Contrail can support federation policies.* The easier it becomes to implement a real federation policy, the more likely the code is to be used for a "proper"federation.

Table 2 describes the features provided by Contrail that are useful in this respect, and their role in the description and management of policies and practices of a "real" federation.

6.2.1 Incident response

When a federation is providing login or access to services spread across providers, it becomes necessary to have a coordinated incident response: otherwise an attacker (who in the worst case may have stolen a credential) will be able to misuse resources at sites that have not responded to the attack. A second case arises when the federation infrastructure is provided using similarly versioned software – this is particularly the case in the early stages of a project where the versions of the software deployed at sites often have to be in sync with each other – thus, sites will need to respond to known exploits in a consistent manner.

While technically outside the scope of Contrail, a federation-as-a-service will need to take this incident handling into account. Practical experiences can be gleaned from grids and national SAML (Shibboleth) federations. For example, it may be useful to set up a vulnerability handling team, which will assess the risk to the middleware and the infrastructure, and coordinate the fix with developers, and the deployment of the fix with the site administrators/coordinators. This assessment will typically be kept confidential, to avoid revealing vulnerability information to potential attackers before the problem is fixed and the software patched. Secondly, there has to be a team dealing with an actual incident: blocking users

with compromised credentials, investigating logs – both federation middleware and host services such as `ssh` – to find out what users have done, and coordinating with each other. Finally, another lesson shows that it is useful to have occasional security service challenges, where a specific coordinator pretends to have a stolen credential and deletes files, etc., and then challenges sites to (a) report what the credential had done, and (b) block it.

6.2.2 AUP

When end users start using a service, it is often necessary to accept acceptable use policies (AUPs). Each cloud provider will likely have its own AUP, and there may be "implicit" AUPs, such as IT policies from the user's own home institute or business, as well as those of the network providers. These AUPs may describe the types of activities which are not permitted – there may be activities which are not permitted, and in the cloud world services also need to respect other tenants on the infrastructure. Whatever the content of the AUP, users need to agree to it *before* using the services, and preferably *only once* (unless it changes.)

In this model, it follows that the federation needs to track which AUP(s) the user has signed. Ideally there is a single AUP for each federation, and the user just has to sign it once (and again when it changes.) Users who have not signed it will not be able to use the federation. Experiences here from the grid world show that it is best to keep it simple: a simple AUP is more likely to be acceptable to all the sites. However, it is still worth having lawyers look through the policy, and to have processes for changing policies.

However, it is likely that a more heterogeneous federation (encompassing multiple identity providers, diverse user communities, and very different service providers) will need a more nuanced approach. Therefore, it makes sense to provide for the signing of the AUP with the SLA negotiation. If the AUP can be expressed in terms that can be parsed into an SLA template, and the user's agreement or negotiation based on this template is taken as an agreement not only of the service level but also of the AUP, then the federation becomes more useful in the sense that it can deliver very diverse services. Like the support for multiple LoAs, the support for multiple AUPs extends the range of uses of the cloud.

One problem is how it is specified: if the user accepts the AUP during the negotiation then we lose some of the benefits of the brokering done by the federation, as each new provider will require intervention from the user. If all provider AUPs can be expressed in similar ways, it may be sufficient – as with other QoP – to define beforehand which terms are acceptable to the user. A second approach is to request permission from a user whenever a new term is encountered. Getting the user's explicit consent to the acceptance of a term would help mitigate the the classic problem of whether the user can be said to have agreed to an AUP if they

have not explicity signed it, *i.e.*, if the actual acceptance is done by a federation agent.

Another useful compromise is that the user accepts the AUP of each provider once, and the federation records that this acceptance has been made: once accepted, the user is then free to use services from the provider; otherwise the federation will filter out other services.

In this section we have outlined different approaches to the AUP. Either (a) the federation records that the user has accepted a single AUP, or (b) there is a recorded acceptance for each provider. Or, (c), there is a means of defining acceptable (and agreed as far as the user is concerned) terms before resource selections takes place, or (d) during the resource selection. Each approach needs implementing in different ways: (a) is an extra attribute in the federation database; (b) is a new table with a user$\times$provider acceptance; (c) will need implementing in the user portal, based on parsing the SLA templates, and (d) will need some sort of interactive approach to the agreement which makes it clear to the user that the user is agreeing.

Code supporting this does not currently exist in Contrail, although (d) is closest to the current approach in the portal. Expressing AUP in the QoP/SLA would be the next step.

6.3 Making use of existing Identity federations

One of the main requirements from the outset of the project was to support "federated" identity management (not to be confused with the Contrail federation itself.) This means that Contrail should allow users to use external identity providers (IdPs) – endpoints which are capable of, and trusted by the federation to, make assertions about the user. Very briefly, the sort of services expected from an external IdP can include:

- Providing identity assertions about the user, *e.g.*, the user's name, or contact/name related like email address or institutional affiliation. An *identifier* is a special case of this, it is a string of some specified format (like an email address) which is likely, or guaranteed, to be *unique* for the user.

- Provide assurance that the user is the same user who logged in last time, and presented the same identifier.

- Provide authorisation attributes in the (relatively few) cases where the IdP is authoritative for the user's authorisation.

- Traceability – the ability to link the identifier back to a real-life identity of the user, potentially links strong enough to be used in a court of law.

- A certain level *cryptographic strength* and *protocol design* associated with the authentication process, along with additional security measures, together preventing the tampering, misrepresentation, unauthorised impersonation, authentication replay attacks, elevation of privileges, and other types of unauthorised and unacceptable use of the infrastructure.

Not all IdPs need provide all this information, but together they add up to a certain *level of assurance* (LoA) which is used in Contrail as a QoP parameter (see Section 4.3). The important advantage is that the infrastructure is able to assign a consistent LoA to each IdP, and to use this LoA as a basis for allocating services: this approach is essential to supporting the diversity of the use cases, where users already have existing identities from IdPs outside of Contrail, and communities with higher security requirements can choose to allow only those with the highest LoA.

Single Sign-On: One of the main advantages of relying on external identities is that it brings us – to some extent at least – *single sign-on* (SSO). SSO means slightly different things to different people, and could mean any or all of the following:

- A single account is held (somewhere, or centrally) for the user;
- A user can use the same password to access multiple independent resources (without having set the password with each resource);
- A user can log in once, and, subject to a timeout, will not have to authenticate to other services they access;
- A user is authenticated transparently using their "home" desktop, and they do not need to authenticate again as long as they are logged in to their desktop machine.

6.3.1 Shibboleth

Shibboleth is a SAML-based identity-and-resource-provider federation technology. As a rule, federations are made out of the IdPs, and SPs, plus a central WAYF ("where are you from") service which lets users select their home institute. As a web-based system, it offers users accessing resource with browsers a means of logging in with their home id. The advantage is that it offers a relatively high level of assurance, with – most importantly – users (or rather their "home" organisations) having signed up to federation policies. The disadvantage is that IdPs are different, they publish different attributes; and the fact that it is web based[16].

[16]Few European IdPs support SAML ECP.

In Contrail, we offer a service provider to Shibboleth federations. The multi server acts as a translator, which is able to take external federation credentials and translate them to Contrail federation credential. The advantage is that users, typically academic, already have Contrail accounts – we automatically generate a new account upon receiving a new credential from an external IdP (of course, users still need authorisation). The main driver for this technology in Contrail is UC3 (neutron scattering) where single sign on (*cf* PanData) is extremely important. However, supporting this technology also improves the usability of the ConSec code, as other users will have a use case for Shibboleth (*e.g.* the CLARIN project in EUDAT).

6.3.2 OpenID federations

It is possible to build federations with OpenID as well: in this case, we define the set of IdPs we trust, and set the level of assurance accordingly. For an existing federation with its own policies we would typically set the LoA for each IdP the same, and higher than public IdPs (*e.g.* Google) but lower than Shibboleth.

The main driver for OpenID in Contrail was UCs 1 and 2, because every user has an OpenID account somewhere, or can easily get one. With an OpenID account, we have no high assurance of the user's identity, but we at least have some assurance that it is the same user every time. At least it is better than username and password, because users are using their passwords with more than one thing, so are less likely to forget it, and are presumably also less likely to share the account details with others.

OpenID *federations* in Contrail was the potential use case (an addendum to UC3) of climate modelling, based on supporting the Earth System Grid Federation. While using the same technology, we would assign it a higher LoA as mentioned above, because each is very tightly controlled by the organisations participating in the federation (and because of the policies.)

6.3.3 OAuth

OAuth is used in Contrail to manage the internal delegation of federation credentials. However, OAuth could also be used to manage the authentication, *e.g.* by having users authenticate with an external provider which uses OAuth. Examples include ORCID and Facebook. It would be natural for some use cases to support these means of authentication, but there were no strong drivers in Contrail's existing use cases. However, it is expected to be fairly easy to be able to adapt the existing Contrail code to support OAuth for authentication.

The drivers for OAuth (in a future sustainable ConSec) are also expected to become more noticeable if the climate modellers switch to OpenID connect which

is similar to OAuth.

6.3.4 WS-Federation

WS-Federation is based on web services federation services, WS-Trust, WS-Security, etc. While primarily tied to SOAP, there is a WS-PassiveFederation protocol as well which is similar to the way we manage some of the SAML profiles.

6.3.5 Moonshot

Moonshot is a RADIUS-based project, like eduRoam, which aims to provide federated identity management. Taking identities at the network level and bringing them to higher level services (such as portal or `ssh` login), it carries additional SAML assertions about the user. Currently the most mature Moonshot infrastructure is the pilot project run by JANET (UK's NREN). Moonshot would essentially provide authentication based on existing identities, and could potentially provide other useful attributes. However, Contrail would still need to add authorisation attributes.

As of Jan. 2014, the Moonshot infrastructure is available for testing and integration, but there is no federation as such, with policies and operations. The project is a pilot project. While Contrail is following closely the activity, there has not been enough effort, or high enough priority, for a full integration.

6.4 Software sustainability, patching, and other maintenance

In the context of software sustainability – particularly with the EUDAT project – we have advocated sustainability based on *components*. The sustainability is described in further detail in D16.6 [27].

7 Conclusions

The work presented here was started in July 2013, and the evaluations were expanded until the due date of the deliverable. The work has then been going on alongside the final development, with testing trailing behind the bleeding edge, but that also means that potential vulnerabilities have been fed into the development process, so they could be fixed.

As with any IT project, it is clear that Contrail has had to compromise between security and features – the more features are developed for the releases, the less robust and secure they become, quite simply because every feature needs testing

and evaluating and then issues fixed, which takes time away from developing other features. While it is hard to say which approach is right for a general purpose project, in general the balance has been right: delivering a good mix of features and security (when secured properly in a production infrastructure.) Nevertheless, the final releases have been integrated close to the end of the project, and it would be difficult to say that the final release is as secure as we would like for production software. Sustainability and exploitation plans will need to take into account the management and patching of the software, and addressing the outstanding issues.

As a research project, it is worth noting innovative features in security. Taking the Quality of Protection into practice, running code self-tests using temporary auto-generated certificates, using OAuth to manage the list of trusted services per-user, and tracking the delegations, have been examples of novel approaches which are nevertheless close enough to the real world to be usable in projects (as opposed to academic work which has no role outside of its own project.) Moreover, the fact that ConSec (Contrail Security) is built of components which can be deployed and used to a large extent independently also aids the sustainability and longevity of the outcome, as components can be replaced, or supported independently. The overall approach has been pragmatic, making use of existing externally developed components whenever possible, and glueing them together into a federation infrastructure, yet retaining the ability to use components independently. As a consequence, the configuration is also not as easy as we could have desired, as configuration items are held in more than one location, but the addition of admin documents plus a testing tool for the infrastructure have helped address this point.

References

[1] European data infrastructure (eudat). http://www.eudat.eu.

[2] Federal risk and authorization management program (fedramp). www.fedramp.gov/.

[3] *Trusted Mobile Platforms*, chapter Foundations of Security Analysis and Design IV: FOSAD 2006/2007 Tutorial Lectures. Springer-Verlag, 2007.

[4] Open Virtualization Format Specification, 2009. Document Number: DSP0243.

[5] AICPA. Ssae 16 overview. http://ssae16.com/SSAE16_overview.html.

[6] Cloud Security Alliance. Cloud controls matrix (ccm). https://cloudsecurityalliance.org/research/ccm/.

[7] Cloud Security Alliance. Top Threats to Cloud Computing v1.0. Technical report, March 2010.

[8] Suresh Chari, Charanjit Jutla, and Arnab Roy. Universally composable security analysis of oauth v2.0. Technical report, IBM, 2011.

[9] Maurizio Colombo, Aliaksandr Lazouski, Fabio Martinelli, and Paolo Mori. A proposal on enhancing XACML with continuous usage control features. In *proceedings of CoreGRID ERCIM Working Group Workshop on Grids, P2P and Services Computing*, pages 133–146. Springer US, 2010.

[10] Massimo Coppola. Final evaluation of the CONTRAIL system based on the second prototype. Deliverable D10.6, CONTRAIL Project, February 2014.

[11] CSA. Cloudaudit: Automated audit, assertion, assessment, and assurance. https://cloudsecurityalliance.org/research/cloudaudit.

[12] Piyush Harsh. Revised specification of computational resource management functionalities for virtual platforms. Deliverable D5.2, CONTRAIL Project, July 2012.

[13] ISO/IEC. Iso/iec 27001:2013 overview. http://www.iso27001security.com/html/27002.html.

[14] ISO/IEC. Iso/iec 27001:2013 overview. http://www.iso27001security.com/html/27001.html.

[15] ISO/IEC. Iso/iec 27017 overview. http://www.iso27001security.com/html/27017.html.

[16] Yvon Jégou. Revised specification of the system architecture. Deliverable D10.4, CONTRAIL Project, November 2012.

[17] Yvon Jegou. Final specification of the Virtual Execution Platform. Deliverable D5.3, CONTRAIL Project, February 2014.

[18] Jens Jensen. First Report of Security Analysis. Deliverable D7.2, CONTRAIL Project, April 2012.

[19] Thilo Kielmann. Revised design of a Virtual Infrastructure Network. Deliverable D4.2, CONTRAIL Project, June 2012.

[20] Thilo Kielmann. Evaluation of the Virtual Infrastructure Network. Deliverable D4.3, CONTRAIL Project, February 2014.

[21] Thilo Kielmann. Final evaluation of the Runtime Environments. Deliverable D9.3, CONTRAIL Project, January 2014.

[22] Aliaksandr Lazouski, Gaetano Mancini, Fabio Martinelli, and Paolo Mori. Usage control in cloud systems. In *7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, 2012.

[23] Paolo Mori. Security Requirements, Specification and Architecture for Virtual Infrastructures. Deliverable D7.1, CONTRAIL Project, September 2011.

[24] Paolo Mori. Revised Specification and Architecture of Security for Virtual Infrastructures. Deliverable D7.3, CONTRAIL Project, July 2012.

[25] OASIS. extensible access control markup language (xacml) version 3.0. Technical report, OASIS Standard, 2010. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf.

[26] S Pai, Y Sharma, S Kumar, R M Pai, and S Singh. Formal verification of oauth 2.0 using alloy framework. *Proc. Int'l Conf on Communication Systems and Network Technologies (CSNT)*, 2011.

[27] Antonio Salis. Final exploitation plan (conclusion report of exploitation achievements. Deliverable D16.6, CONTRAIL Project, February 2014.

[28] R. Sandhu and J. Park. The UCON_{ABC} usage control model. *ACM Transactions on Information and System Security (TISSEC)*, 7(1):128–174, 2004.

[29] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (blowfish). https://www.schneier.com/paper-blowfish-fse.html.

[30] San-Tsai Sun and Konstantin Beznosov. The devil is in the (implementation) details: An empirical analysis of oauth sso systems. *CCS '12 Proceedings of the 2012 ACM conference on Computer and communications security*, 2012.

[31] Nick Trigg. Comprehensive review of market requirements. Deliverable D16.2, CONTRAIL Project, October 2011.

A Appendix A: Penetration tests

STFC conducted a penetration test against the portal application and the called components, particularly the federation-api. This was structured around the OWASP Web Application Security Testing Cheat Sheet[17].

The test was run via the w3af vulnerability analysis tool against the testbed located at STFC.

There are additional notes on functional design weaknesses that were not tested against testbed systems.

Details of the testbed The software tested was the Contrail Security component of the overall Contrail project. The nominal version was binary package 1.3 of the contrail.ow2.org repository, however this was from the testing branch so contained many small enhancements from the 1.3 baseline package. The testing version was build number 3087, dated 7th November 2013 with the exception of the contrail-oauth-as binary package which was build number 3108 dated 11th November 2013.

The Contrail system ran on Ubuntu 12.10. The hardware used was a dual-core twin processor Intel Xeon 5130 running at 2.0 GHz, with 8GB of memory and a 223GB hard-drive.

The configuration of the software that was used was based on the defaults from the binary package but customised for different names for certificates, different server names and ip addresses, a different Tomcat connector security configuration, and increased debug logging.

The Checklist: Note that probability refers to the likelihood of an attack succeeding if it were used, not that the likelihood of an attack would be used.

[17] Available at:- https://www.owasp.org/index.php/WebApplicationSecurityTestingCheatSheet

A.1 Information Gathering

Table 3: OWASP Web Application Security Testing Cheat Sheet - Information Gathering.

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Manually explore the site	The Contrail project is open-source. Program source can be inspected for weaknesses. Standard configuration params are visible and include URLs, default passwords, private keys used for testing and example code to access it.	Misconfiguration could result in any of these being used unsafely	Some likelihood of misconfiguration	Ensure sysadmins cannot misconfigure the system through the use of a checking utility.
Spider/crawl for missed or hidden content	No problems found			
Check for files that expose content, such as robots.txt, sitemap.xml, .DSStore	No problems found but see item "Manually explore the site"			

Continued on next page

Table 3 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Check the caches of major search engines for publicly accessible sites	The website was not referenced but people connected with the project were	The person could be targeted, for example via a scripting email	Moderate likelihood	No defence possible
Check for differences in content based on User Agent (eg, Mobile sites, access as a Search engine Crawler)	No problems found			
Perform Web Application Fingerprinting	The webserver (tomcat and apache) and operating system were fingerprinted accurately. The framework (django) was fingerprinted.	Attacks can be targeted on known versions	Small possibility	Obfuscate all visible components

Continued on next page

Table 3 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Identify technologies used	By inspecting the open-source project it was found that the system used Java, Django, PHP, Apache, Tomcat, various java libraries, various php libraries particularly pysaml2.	Attacks can be targeted on known vulnerabilities	Moderate likelihood	Patching against vulnerabilities must be kept fully up-to-date
Identify user roles	No problems found			
Identify application entry points	By inspecting the open-source project all entry points were found	Greater ease of attack by knowing entry points. Attacks can be targeted.	Moderate likelihood	Obfuscate entry-points and URLs by editing the configuration files
Identify client-side code	There are example programs available from the open-source project	Misconfiguration allowing use of default shared-secret is possible	Small likelihood	Systems administrator must be prevented from using default configurations via a checking utility.

Continued on next page

Table 3 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Identify multiple versions/channels (e.g. web, mobile web, mobile app, web services)	No problems found			
Identify co-hosted and related applications	Authentication gateway into EUDAT and UK Federation	If compromised at Contrail the federations connected are at risk too	High likelihood due to known vulnerabilities in Contrail	Disconnect from other federations until vulnerabilities are removed.
Identify all hostnames and ports	Standard ports used. Hostnames were not visible.	Attacks are easier	Small likelihood	Do not publicise the hostname
Identify third-party hosted content	No problems found but see item "Identify co-hosted and related applications"			

A.2 Configuration Management

Table 4: OWASP Web Application Security Testing Cheat Sheet - Configuration Management.

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Check for commonly used application and administrative URLs	No problems found			
Check for old, backup and unreferenced files	No problems found			
Check HTTP methods supported and Cross Site Tracing (XST)	No problems found			
Test file extensions handling	No problems found			
Test for security HTTP headers (e.g. CSP, X-Frame-Options, HSTS)	No problems found but see item below "CSRF"			
Test for policies (e.g. Flash, Silverlight, robots)	No problems found			
				Continued on next page

Table 4 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for non-production data in live environment, and vice-versa	Development and test data were found but not in a live environment. But the server is exposed to the Internet so there is little distinction between live and test environment	Default data visible on the open-source project could be used to gain access	High probability	Remove test data and change default configuration
Check for sensitive data in client-side code (e.g. API keys, credentials)	No problems found			

A.3 OWasp results

Secure Transmission

Table 5: OWASP Web Application Security Testing Cheat Sheet - Secure Transmission.

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Check for commonly used application and administrative URLs	No problems found but note the test system had tools installed such as phpmyadmin which is frequently attacked			
Check SSL Version, Algorithms, Key length	No problems found			
Check for Digital Certificate Validity (Duration, Signature and CN)	No problems found			
Check credentials only delivered over HTTPS	User can use insecure http protocol to reach portal when misconfigured.	Man in the middle attack	Certain if misconfigured	Use only https. Autoforward to https.
Check that the login form is delivered over HTTPS	Login accessible thru http using the default install	Man in the middle attack	Certain if misconfigured	Use only https. Autoforward to https.
				Continued on next page

Table 5 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Check session tokens only delivered over HTTPS	Important components eg ca-server and federation-api, can be misconfigured to use http at the config files.	Man in the middle attack	High	Ensure configuration is correct. Don't allow http at apache and tomcat. Put comments into config files advising sysadmins on best practice and impact.
Check session tokens only delivered over HTTPS	Tomcat can be misconfigured to allow http thru Want param in server.xml	Man in the middle attack	High	Ensure configuration is correct. Don't allow http at apache and tomcat. Put comments into config files advising sysadmins on best practice and impact.
Check if HTTP Strict Transport Security (HSTS) in use	No.			

Authentication

Table 6: OWASP Web Application Security Testing Cheat Sheet - Authentication

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Test for user enumeration	Trivial enumeration of user accounts by unauthenticated persons or bots anywhere on the web, showing email addresses and human names even when correctly configured with https.	Social engineering attack. Spam. Account hijack. Very angry users.	Certain	Alter program functionality. Although https requires a certificate to be presented by the user's browser or a bot, it only has to be trusted to a known root CA. There is no record kept of connections so a malicious user can expose his true identity without risk to gain access.
				Continued on next page

Table 6 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for authentication bypass	When OpenID authentication is used the user remains authenticated by a session token at the portal for a period of time even though the remote OpenID account has been revoked or password altered. Password is stored in database as hash, so privileged user can steal passwords by accessing dbms.	1. Invalid access 2. Password theft. 3. Insider attack	1. High 2. Low, it depends on getting privileged access. 3. High.	1. Alter program 2. Do not store passwords in dbms. 3. Do not store passwords in dbms.
Test for bruteforce protection	None implemented. Portal responds at top speed indefinitely until a weak password is found.	Password cracking - unauthorised login	High	Invalid logins to be blocked after a number of attempts.

Continued on next page

Table 6 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test password quality rules	No problems found - passwords are set by Contrail administrator or auto-generated on registration of user			
Test remember me functionality	No problems found - not used server-side			

Continued on next page

Table 6 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for autocomplete on password forms/input	1. Not implemented within django, but can be implemented at browser level. 2. A user account is automatically logged in and given access to its portal page through OpenID so sharing a browser session with another user on the same workstation will login as the other user.	1. Unauthorised login at unprotected workstation. 2. Privilege escalation	Certain	

Continued on next page

Table 6 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test password reset and/or recovery	No problems found - passwords are reset by Contrail administrator			
Test password change process	No problems found - passwords are reset by Contrail administrator			
Test CAPTCHA	Not in use	Not relevant	Not relevant	Not relevant
Test multi factor authentication	Not in use	Not relevant	Not relevant	Not relevant
Test for logout functionality presence	A two-stage logout is used - from the portal website and from the OpenId identity used to login. The OpenId login gives immediate login to the portal without querying the user. Therefore on a shared physical desktop system some user could impersonate another	Impersonation attack	Certain	Change system functionality

Continued on next page

Table 6 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for cache management on HTTP (eg Pragma, Expires, Max-age)	No problems found			

Session Management

Table 7: OWASP Web Application Security Testing Cheat Sheet - Information Gathering

Check	What was found	Impact	Probability of successful exploit	Recommendation
Establish how session management is handled in the application (eg, tokens in cookies, token in URL)	PHP sessionid. Cookies. Certificates. No problems found			
Check session tokens for cookie flags (httpOnly and secure)	No problems found			

Continued on next page

Table 7 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Check session cookie scope (path and domain)	No problems found			
Check session cookie duration (expires and max-age)	Not investigated but likely to be a week			
Check session termination after a maximum lifetime	No problems found			
Check session termination after relative timeout	No problems found			
Check session termination after logout	Two-stage logout can be confusing to users and creates the possibility of an impersonation attack on a shared physical desktop	Impersonation attack. Privilege escalation.	Certain	Change functionality

Continued on next page

Table 7 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test to see if users can have multiple simultaneous sessions	Bug if user sessions become muddled when more than one OpenID session is authenticated.	Privilege escalation by incorrect login as a different user account	Certain	Fix bug.
Test session cookies for randomness	Not investigated but likely to be satisfactory			
Confirm that new session tokens are issued on login, role change and logout	No problems found			
Test for consistent session management across applications with shared session management	Two-stage logout problems - see item "Check session temination after logout"	Impersonation. Privilege escalation.	Certain	Change functionality.
Test for session puzzling	No problems found			
Test for CSRF and clickjacking	Clickjacking vulnerability found due to misconfiguration of django (for the portal)	Password theft	Certain	Enable django protections against clickjacking.

Authorization

Table 8: OWASP Web Application Security Testing Cheat Sheet - Authorization

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Test for path traversal	No problems found			
Test for bypassing authorization schema	No problems found			
Test for vertical Access control problems (a.k.a. Privilege Escalation)	1. Browser password autocomplete allowed 2. Shared workstation and shared browser allows use of another user's credentials.	Impersonation attack and privilege escalation	Certain	Change system functionality

Continued on next page

Table 8 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for horizontal Access control problems (between two users at the same privilege level)	Impersonation attack and privilege escalation		Certain	Change system functionality
Test for missing authorization	When misconfigured into insecure mode all authentication is bypassed	Gets root	Certain	See discussion below. Change functionality.

Data Validation

Table 9: OWASP Web Application Security Testing Cheat Sheet - Data Validation

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for Reflected Cross Site Scripting	No problems found			
Test for Stored Cross Site Scripting	No problems found			
Test for DOM based Cross Site Scripting	No problems found			

Continued on next page

Table 9 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for Cross Site Flashing	No problems found			
Test for HTML Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for SQL Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for LDAP Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for ORM Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for XML Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.

Continued on next page

Table 9 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for XXE Injection	Not investigated but not likely to be a problem			
Test for SSI Injection	Not investigated but not likely to be a problem			
Test for XPath Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for XQuery Injection	Not investigated but not likely to be a problem			
Test for IMAP/SMTP Injection	Not investigated but not likely to be a problem			
Test for Code Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.

Continued on next page

Table 9 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for Expression Language Injection	Not investigated but not likely to be a problem			
Test for Command Injection	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for Overflow (Stack, Heap and Integer)	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for Format String	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.
Test for incubated vulnerabilities	Not investigated but not likely to be a problem			
Test for HTTP Splitting/Smuggling	Not investigated but not likely to be a problem			

Continued on next page

Table 9 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for HTTP Verb Tampering	Not investigated but not likely to be a problem			
Test for Open Redirection	Not investigated but not likely to be a problem			
Test for Local File Inclusion	Not investigated but not likely to be a problem			
Test for Remote File Inclusion	Not investigated but not likely to be a problem			
Compare client-side and server-side validation rules	Not investigated but not likely to be a problem			
Test for NoSQL injection	Not investigated but not likely to be a problem			
Test for HTTP parameter pollution	Problems likely due to unfiltered Rest api. Further investigations required.	Severe compromise	Certain if this vulnerability is confirmed.	Change software to filter input.

Continued on next page

Table 9 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for auto-binding	Not investigated but not likely to be a problem			
Test for Mass Assignment	Not investigated but not likely to be a problem			
Test for NULL/Invalid Session Cookie	Not investigated but not likely to be a problem			

Denial of Service

Table 10: OWASP Web Application Security Testing Cheat Sheet - Denial of Service

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Test for anti-automation	No protection found. 1. Bots can automate password cracking attempts. 2. The portal registration page can be used to flood the system and so deny service.	1. Bots can probe any page. 2. Denial of service	Certain	Failed login attempt protection required. Registration page requires protection measures.
				Continued on next page

Table 10 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test for account lockout	No account lockout feature for protection against automated login attempts was found. However there was an inadvertent account lockout found due to a bug that overwrites the password of an account with some randomly chosen password when OpenID was used but failed to login.	Denial of service at user account level.	Certain	Fix bug.
Test for HTTP protocol DoS	No problems found			
Test for SQL wildcard DoS	No problems found			

Business Logic

Table 11: OWASP Web Application Security Testing Cheat Sheet - Business Logic

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Test for feature misuse	Problems likely. Not tested via w3af.			
Test for lack of non-repudiation	No problems found			
Test for trust relationships	Problems likely. Not tested via w3af.			
Test for integrity of data	Problems likely. Not tested via w3af.			
Test segregation of duties	Although multiple roles may be required by a single user account the portal does not support more than one.	Ease of use	Very low	Fix feature.

Cryptography

Table 12: OWASP Web Application Security Testing Cheat Sheet - Cryptography

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Check if data which should be encrypted is not	Pysaml2 reports in its logs that "Unencrypted message received" even when https used throughout and secure mode enabled.	Sniffing	Likely	Investigate and fix bug.
Check for wrong algorithms usage depending on context	No problems found			
Check for weak algorithms usage	No problems found			
Check for proper use of salting	Misconfiguration of simplesamlphp, django and php can cause the use of the default salt	Sniffing	High	Installation script to force change of salt.
Check for randomness functions	No problems found			

Risky Functionality - File Uploads

Table 13: OWASP Web Application Security Testing Cheat Sheet - Risky Functionality - File Uploads

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test that acceptable file types are whitelisted	No problems found			
Test that file size limits, upload frequency and total file counts are defined and are enforced	No problems found			
Test that file contents match the defined file type	No problems found			
Test that all file uploads have Anti-Virus scanning in-place.	No problems found			
Test that uploaded files are not directly accessible within the web root	No problems found			
Test that uploaded files are not served on the same hostname/port	No problems found			

Continued on next page

Table 13 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Test that files and other media are integrated with the authentication and authorisation schemas	No problems found - but note that a user at the portal can download his certificate and private key which could be accessible by a malicious user sharing the physical desktop machine			

Risky Functionality - Card Payment

This paragraph does not apply to us. There are no financial transactions in use.

HTML5

Table 14: OWASP Web Application Security Testing Cheat Sheet - HTML 5

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Test Web Messaging	No problems found			
Test for Web Storage SQL injection	No problems found			
Check CORS implementation	No problems found			
Check Offline Web Application	No problems found			

A.4 W3AF results

The following sections are additional to the OWasp categories.

Table 15: OWASP Web Application Security Testing Cheat Sheet - W3AF results

Check	**What was found**	**Impact**	**Probability of successful exploit**	**Recommendation**
Click-jacking	1. The url <portal>/user has no defence against clickjacking. An attacker could insert his own content once the user has logged-in and is viewing his portal home page. 2. Federation-api Rest interface has no defence.	1. Clickjacking 2. Clickjacking	1. High 2. High	Use the X-Frame-Options HTTP response header to block click-jacking.
				Continued on next page

Table 15 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
Path disclosure	Tomcat6 is disclosing the path to its binary.	Path disclosure	Low	It's not clear to me how the path is being found. Possibly through the tomcat management app. The report is unclear.
Misconfigured access to the tomcat management application	The login to the tomcat management tool can be bypassed and access can be gained to control the tomcat apps.	Privilege escalation	High	Fix the misconfiguration.

Continued on next page

Table 15 – continued from previous page

Check	What was found	Impact	Probability of successful exploit	Recommendation
SSL not used at login	When using insecure mode the login to the tomcat management app, the django portal login page and editing pages for the django portal allow credentials to be sent over the insecure protocol. If the traffic is sniffed unauthorised access is achieved.	Privilege escalation	High	Use only a SSL connection
Cookie access	A cookie without the HttpOnly flag was sent. This happened at a number of places if tomcat management app, the django portal login page	Cross site scripting	Unlikely	Use HttpOnly flag on cookies.

A.5 Explanation of W3AF results

Clickjacking Clickjacking, also known as a “UI redress attack”, is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to other another page, most likely owned by another application, domain, or both. Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker. The most popular way to defend against Clickjacking is to include some sort of “frame-breaking” functionality which prevents other web pages from framing the site you wish to defend. The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a <frame> or <iframe>. Sites can use this to avoid Clickjacking attacks, by ensuring that their content is not embedded into other sites.

Cookie without HttpOnly flag The HttpOnly flag prevents potential intruders from accessing the cookie value through Cross-Site Scripting attacks.

Cross Site Request Forgery CSRF attacks allow a malicious user to execute actions using the credentials of another user without that user’s knowledge or consent.

A.6 Functional Weaknesses

The OAuth Rest API calls have very little security on them so anyone with a user certificate issued by the recognised root ca can access it even in secure mode. A user certificate issued by the CA can be got using the create-user-cert utility that runs from the command line against the java webapp on tomcat. It can also be got by registering a new user account, logging into the portal without credentials and downloading it from the portal account settings page. To use the OAuth client credentials flow the user must be registered in advance. But it is possible for users to self-register and immediately query the federation-api to get their uuid when the federation-api has been set up to allow insecure http access from any public browser. Once his uuid is known it is trivial to make full use of the OAuth client credentials flow.

Anyone without a user certificate at any remote location can also write into the oauth database when the federation-api is set up to allow insecure http access.

Locking down the federation-api to permit only secure access is difficult and uncertain, with multiple paths through the design logic and multiple paths through the program logic.

There is no functionality to disable a user account, permanently or temporarily. It can only be deleted. Sometimes it is desirable to retain the history trail for an account but deleting it would probably involve deleting that history trail or removing the point of access (ie primary key on User table) so that it is not easily accessible or cannot be related back to other data with certainty of having the right account.

Vulnerability: Identity Theft On the Registration page of the portal it is possible to enter someone else's identity and make use of it immediately after without any need for confirmation.

On the Registration page of the portal there should be some confirmation of identity e.g. an email sent back to the email address entered asking the new user to confirm the account creation.

There is no audit trail or history trail. The activity of an authorised user or of an intrusion attempt is not logged. It is not possible to determine after the event what was done, using what resources, when it happened or what might have been compromised.

A user sharing a physical desktop machine with another user is at high risk due to cookies used by OpenId login being accessible, confusing two-stage logout, and password recall at the browser.

There is no assessment of whether or not there are any known vulnerabilities within the program source libraries used. If a serious vulnerability in a library were to be found there is no human procedure for monitoring for alerts or assessing their impact in a timely manner.

Vulnerability: Probable misconfiguration The configuration of the system is spread across multiple configuration files that are difficult to keep correctly synchronised when editing by hand. For example the files /etc/contrail/contrail-federation-web/federation-web.conf and /usr/lib/contrail/federation-web/src/federweb/settings/–init–.py have most of the parameters of the first file duplicated in the second file. Which takes precedence? Does the precedence change at different points in the software?

There are also configuration parameters contained within the database. For example the oauth-as parameters that control which client is used for login authentication and what shared-secret is used. This has to agree with parameters contained within four other configuration files.

There is little validation of the parameters in the configuration files or the

database. We found that it was unclear what parameters were required because no error was raised when one was omitted due to misconfiguration. For example we found that in configuration file /var/lib/tomcat6/webapps/federation-api/WEB-INF/web.xml there was a parameter "test-mode" that disabled the use of the Herasaf authorizer (xacml) allowing login without checking the user's credentials against the xacml file. However if this test-mode parameter were missing it defaulted to a value of True which caused the disabling of the use of the Herasaf authorizer. This is not appropriate - it should fail safe. Furthermore it is unclear where the test-mode parameter should be used in the file web.xml. We found two usages of it but suspect three exist, all of which do not fail safe.