myTrustedCloud: Trusted Cloud Infrastructure for Security-critical Computation and Data Managment

David Wallom, Matteo Turilli[†]
Oxford eResearch Centre,
University of Oxford,
Oxford, UK
{david.wallom,matteo.turilli}@oerc.ox.ac.uk

Gareth Taylor, Nigel Hargreaves
Brunel Institute of Power Systems,
Brunel University,
London, UK
{gareth.taylor,nigel.hargreaves}@brunel.ac.uk

Andrew Martin, Anbang Raun
Department of Computer Science,
University of Oxford,
Oxford, UK
{andrew.martin,anbang.raun}@cs.ox.ac.uk

Alan McMoran
Open Grid Systems Ltd,
Glasgow, UK
alan@opengridsystems.com

Abstract. **Cloud Computing provides an optimal infrastructure to utilise and share both computational and data resources whilst allowing a pay-per-use model, useful to cost-effectively manage hardware investment or to maximise its utilisation. Cloud Computing also offers transitory access to scalable amounts of computational resources, something that is particularly important due to the time and financial constraints of many user communities. The growing number of communities that are adopting large public cloud resources such as Amazon Web Services [1] or Microsoft Azure [2] proves the success and hence usefulness of the Cloud Computing paradigm. Nonetheless, the typical use cases for public clouds involve non-business critical applications, particularly where issues around security of utilization of applications or deposited data within shared public services are binding requisites. In this paper, a use case is presented illustrating how the integration of Trusted Computing technologies into an available cloud infrastructure – Eucalyptus – allows the security-critical energy industry to exploit the flexibility and potential economical benefits of the Cloud Computing paradigm for their business-critical applications.**

Keywords-component. Cloud Computing, Trusted Computing, Cloud Security, energy industry.

I. INTRODUCTION

Many communities have already adopted the 'cloud', a flexible computational platform allowing scalability and a service based provision model. Unfortunately, there are currently significant limitations when using a cloud infrastructure to perform security-critical computations and/or storing sensitive data. Specifically, at the moment there is no way to guarantee the trustworthiness of a Virtual Machine (VM) in terms of its origin and identity and the trustworthiness of the data uploaded and managed by the Elastic Block Storage or the Simple Storage Service (S3).[1] These limitations are due to the necessity to rely on an untrusted infrastructure – the public cloud – in order to instantiate VMs, mount a block device or return a data object through a Web Service. Users are currently not able to attest that the VM they are using has been instantiated from the correspondent image uploaded in the S3 service or to attest that the images they uploaded into the S3 are the same as those stored in the S3 repository. Analogously, users are not able to attest that the data contained inside their EBS volumes or those returned by the S3 service have not been accessed or altered by other users or processes.

Because of these limitations, public Cloud Computing uptake by business-critical communities is limited. A number of communities whose emerging information models appear otherwise well suited to Cloud Computing are forced either to avoid the pay-per-use model of service provision or to deploy a private cloud infrastructure. Deploying a private cloud is rarely a desirable solution. It requires an extended time frame and relevant investment in hardware, management and software resources. These limitations also apply to the deployment of a private cloud based on open source software because while licencing costs are eliminated, the bulk of the investment in hardware and support resources is still required. Moreover, the available open source cloud solutions – Eucalyptus[3], OpenStack[4], OpenNebula[5], Nimbus[6] – currently suffer from various limitations, especially as it relates to accounting, resource management, reliability and scalability.

The myTrustedCloud project investigated these limitations by focusing on a use case involving the highly business-critical UK energy industry. The requirement for Smart Grids implementation [7] means that the energy industry will have to leverage increased computational support for organisational

[†]Joint first authors and correspondent contacts.

[1] Or other services with equivalent functionalities offered by different cloud providers.

intra- and inter-operability of data based on emerging information exchange standards such as the Common Information Model (CIM). Ultimately, the computational and data infrastructure required by such a support will have to allow for the exchange of data produced by the Advanced Metering Infrastructures at a distribution network level in order to foster Enhanced State Estimation and On-line Condition Monitoring [8]. This infrastructure will have to serve multiple operators, often in a competitive market environment, in a context of highly regulated and constrained relationships.

Cloud Computing addresses at least two fundamental requirements of the UK energy industry community. First, accurate network simulations require highly variable quantities of computational resources depending on the contingent situation of energy delivery or on the type of energy delivered. Renewable energy output is typically much less predictable than the constant output offered by conventional generation sources, such as coal, oil, gas or nuclear. For this reason, running simulations on the cloud allows for dynamic scaling of the required computational and data resources. Second, many operators do not have the infrastructure to support the growing need for accurate predictive and historical simulations imposed by the adoption of renewable energy sources and the on-going development of smart grids. Cloud Computing allows these operators to reduce or avoid over-investment in hardware resources and their associated maintenance.

In order to capitalise on the opportunities offered by Cloud computing to the UK energy industry, myTrustedCloud has integrated Trusted Computing into Eucalyptus [2] thereby enabling a cloud solution to meet the stringent security requirements of the UK energy industry. A significant community has developed around Trusted Computing and its associated hardware components, trusted Platform Module (TPM) [9], TPM enabled devices and software such as trusted virtualisation [10]. This is now being extended into different application areas such as networks and distributed systems design. Scarce availability of components has meant that trusted computing has, until now, been mostly confined to theoretical investigations. The technologies are now becoming main-stream enough that physical and systems integration research is timely and, thanks to the availability of commodity components, it is now feasible to design and build pilot demonstrators, upon which different research areas may test their full capabilities [11].

II. Use Case Analysis

The use case of myTrustedCloud refers to data exchange between the fourteen UK Distribution Network Operators (DNOs), the Scottish and Offshore Transmission System Operators (TSOs), and National Grid (NG) the GB transmission system operator. The DNOs and TSOs provide data to NG in strict accordance with the GB Grid Code, Distribution Code and Service Capability Agreements [12]. Presently, these codes call for an annual submission regarding the DNO's network apparatus and data. This normally amounts to one major update on week 24 and some minor revisions estimated at one per month. These data, though only uploaded once with a subsequent model update, may be then reutilised by other NG departments and DNO user representatives.

DNOs' data contain a large volume of information about the state of their electrical networks. A typical data set includes data about connectivity arrangements, electrical loads and power injections reporting the values for parameters such as Power (MW), average Voltage (MVAr), Power Factor (PF), Sub-transient currents, transient currents and reactance:resistance (*x:r*) ratio at each grid connection point. These data are often submitted in an arbitrary format (e.g. spreadsheet, PDF file, MS Word document) and the NG Data & Analysis team passes them through a number of manual processes in order to arrive at a merged GB node and branch model with branch parameters of loads. This model is then converted into a switch level model and disseminated to the control, planning and network design departments. Such information exchanges enable essential operational activities such as coordinated outage planning, evaluation of fault in-feeds and loads, simulation of short and medium term scenarios and evaluation of planned updates against historical network status. An increase in the frequency of such exchanges of information and data based upon emerging standards will be essential to enable fully interoperable smart grid functionality at both a national and international level.

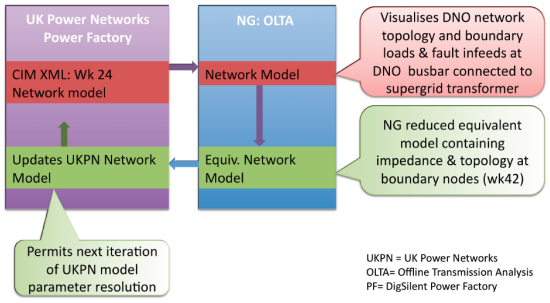

Figure 1. Schematic of CIM data file exchange between UK Power Networks and National Grid.

Currently, NG and UK Power Networks (UKPN) both operate the PowerFactory software suite produced by DIgSILENT GmbH [13] for power system model management purposes. PowerFactory allows for exporting and importing network models (developed by utility engineers) in the Common Information Model (CIM)[14] XML format, corresponding to release 14v2 of the CIM as specified by the ENTSOE-UCTE profile. An example of this data exchange is shown in Figure 1. The adoption of a common format when exchanging network models, gives the opportunity to aggregate the DNOs' models into a global, more refined and accurate representation of the overall state of the national power network.

At present, a typical UKPN CIM data set describing their reduced network to the High Voltage (HV) grid interfaces,

[2] Eucalyptus is a readily available, open source implementation of a cloud infrastructure produced by Eucalyptus Systems.

would amount to about 3.7Mb, including both network diagrams and schedule data. These files would then be merged to form a reduced NG GB network model and exchanged back to UKPN for further calculations aimed at refining their initial load and fault parameter estimations. With current network operation procedures there are no other clear calls for uploading these data apart from those already mentioned. Nonetheless, with the prospect of an increasingly large number of embedded and distributed generators, it is likely that a greater frequency and increased level of data exchanges will be necessary in order to maintain the secure operation and stability of the UK electrical network.

From the description of the use case, three security requirements clearly emerge:

1. Different operators (DNOs i.e. UKPN, TSOs i.e. NG) require a diversified access model to the data sharing infrastructure. Each operator has to upload its own data into a dedicated location that then has to be shared, with different privileges, across a known set of other operators;
2. The data exchanged among operators are potentially highly sensitive as they describe the state of portions of the national transmission and distribution system network, an infrastructure of interest for national security;
3. Ownership and integrity of the data set must be trusted in order to produce reliable aggregated models.

In order to match these requirements with the opportunity to adopt Cloud Computing, some features of the Trusted Computing platform have been implemented into Eucalyptus.

III. Integration of Trust & Cloud

Trusted Computing enables Eucalyptus users with the capability of verifying the integrity of the Virtual Machines (VMs) and the EBS volumes they own on the cloud. In this scenario, the Trusted Computing framework [15] proposed by the Trusted Computing Group (TCG) is integrated with Eucalyptus so as to provide remote attestation services to cloud users.

Considering the Eucalyptus infrastructure, as depicted in Figure 2, the integrity of a VM relies on three components: VM, Node Controller (NC) and Storage Controller (SC). Attestations of these three components should be made separately in order to provide proofs for the integrity of the entire VM's lifecycle.

- **Attestation of VMs.** Ensures that only expected programs with expected configuration files (including the data to process) are loaded inside the VM.
- **Attestation of NCs.** Ensures that the expected VM has been instantiated and that only an expected software stack can manipulate its state. With "expected" we denote that the VM the user is currently connecting to, is genuinely loaded by the genuine hypervisor with specified parameters, including the kernel and root images, the networking parameters and the virtual storage binding parameters.
- **Attestation of SCs.** Ensures that the VM is binding to the expected virtual storage, and that the state of the virtual storage can only be manipulated by an expected software stack. In this context, "expected" denotes that the virtual storage connected to the user's VM is genuinely loaded and managed by the genuine EBS software with the specified parameters.

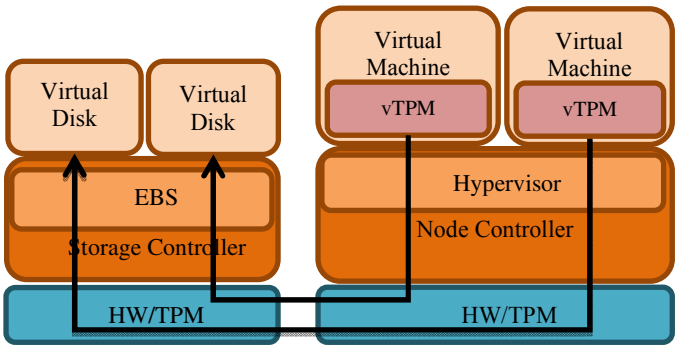

Figure 2. Logical layers of the dependencies of VMs on Eucalyptus.

A. *Iterative Attestation*

The attestations of the VM, NC and SC can be performed separately. Users initiate three different attestation sessions, verify the integrity of all the components and combine the results to form an overall attestation. The implementation of this process has to take into account at least two problems. First, exposing the internal infrastructure of the cloud to the users widens the attack surface and violates the principle of isolation between VMs and hypervisor. Second, if for every client the attestations are delegated to a Trusted Third Party, a single-point-of-failure is created that makes the whole infrastructure very difficult to scale.

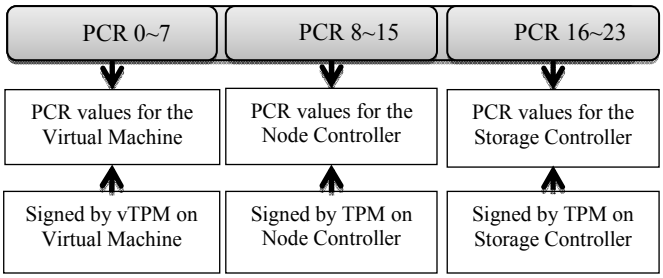

Figure 3. Attestation ticket for the three-layer deep-quote.

In the myTrustedCloud use case, an iterative attestation schema has been adopted, also referred to as deep-quote [16]. In this approach, users initiate only one attestation session for the VM they are connecting to. The returned attestation ticket incorporates three separate parts, representing respectively the integrity of the VM, that of the NC hosting the VM and that of the SC hosting the EBS volumes. As depicted in Figure 3, the first part of the returning attestation ticket contains the Platform Configuration Register (PCR) values recorded in the Trusted Platform Module (TPM) of the VM; the second part of the ticket contains the PCR values of the NC hosting the VM;

while the third part contains the PCR values of the connected SC. The three parts of the ticket are signed respectively by the VM, NC and SC TPMs, each at the same time as the others.

Thanks to this approach to attestation, users can first verify whether the PCR values are generated by genuine TPMs, and then deduce the genuine configuration of their VM and of the underlying Cloud infrastructure.

It should be noted that in order to avoid replay-attacks, a cryptographic nonce generated by the user is specified for the attestation session. Moreover, proofs for the bindings of these components should also be provided so to avoid that a tampered NC or SC is able to forward the attestation request with the provided nonce. Furthermore, in order to avoid a Man-In-The-Middle attack, when a genuine NC/SC receives a new attestation request, checks should be made to examine whether the requesting VM is running on, or is connecting to that specific NC and SC. As the genuine enforcement of this checking is also reflected in the PCR values, we can safely assume that NC/SC with the expected PCR values entail these strong bindings. In addition, NC/SC will hash the PCR values of the VM with the specified nonce as the new nonce for their attestations.

B. Implementation of the iterative attestation procedure

Figure 4 depicts the trusted architecture of the NC. The myTrustedCloud project has focused mainly on the QEMU [17] and the Kernel Virtualization Module (KVM) [18] virtualization implementation and on the Canonical [19] packaging of Eucalyptus in their Ubuntu [20] operating system. The NC initialisation process has been implemented as follows:

1. TPM support is enabled in the BIOS in order to measure the initial state of the physical system.
2. Trusted Grub is then installed to bootstrap the environment of the NC and to measure the bootstrapping configuration, the kernel and initrd images, and the kernel arguments.
3. The kernel – with the IBM Integrity Measurement Architecture (IMA) [21] patches applied – measures all the applications, kernel modules and possibly configuration files loaded at boot time.

This implementation builds a trust chain rooted from the BIOS guaranteeing that all the software components of the NC are measured before being loaded.

The Trusted Core Service Daemon (TCSD) is used to export the trusted services of the physical TPM into the upper layers of the virtual environment. Moreover, in order to enable attestation inside a VM, the QEMU emulator is patched with TPM support. The TPM support consists of a backend vTPM and frontend TPM driver. The backend vTPM – implemented by means of the LibTPMs – emulates one TPM for each VM. The frontend driver exposes the implemented TPM to the VM Operating System.

The trust chain is extended to the VM by using a virtual BIOS - the SeaBIOS in this case - patched with the TPM support. In this way, the BIOS can initialise a vTPM for the VM and take the measurement of its virtualized hardware components. In the Eucalyptus NC, the kernel of the VM is directly loaded by QEMU so no boot loader exists. This is a problem because the kernel of the VM cannot be measured before being loaded and, as a consequence, its hash value cannot be stored into the vTPM.

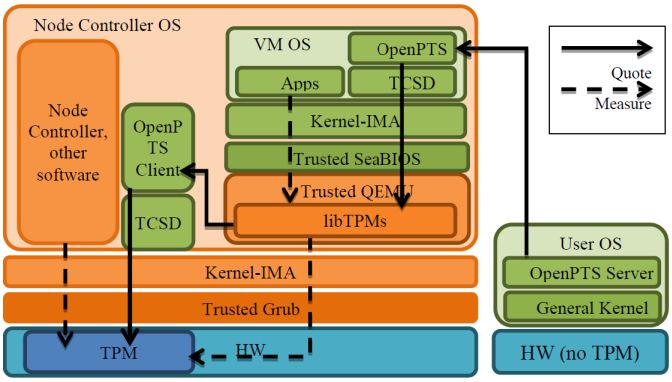

Figure 4. VM & Node Controller Attestation.

This problem has been addressed by modifying the QEMU configuration firmware so as to store the memory address of the loaded kernel. SeaBIOS extracts this memory address by interrogating and reading the I/O ports of the QEMU configuration firmware and uses it to measure the loaded kernel and obtain its hash value. SeaBIOS then stores the kernel hash value into the vTPM while QEMU measures and stores into the vTPM the hash value of the VM root image. In this way, SeaBIOS is able to obtain also the hash value of the VM root image from the firmware image of QEMU.

The Open Platform Trust Services (OpenPTS) [22] are used to implement the VM attestation procedure. The OpenPTS is divided into client and server sides. The client is installed into the VM while the server is usually hosted on the user platform or on a third party verifier platform (see Figure 4, right panel). The OpenPTS server maintains a trusted list of measurements so as to examine whether the measurements reported by the OpenPTS client and the corresponding Stored Measurement Logs (SMLs) are trustworthy. From an implementation point of view, the OpenPTS client is deployed inside the VM instantiated by the NC and used to invoke the TPM_Quote instruction of the vTPM. This instruction fetches the list of measurement values of all the software components of the VM signed by the vTPM. Once fetched, this list is reported by the OpenPTS client to the OpenPTS server for matching with the SMLs.

Once the attestation of the VM loaded by the NC has been achieved, the trustworthiness of the NC must be attested too. Otherwise, the risk is instantiating a trusted VM by means of an untrusted infrastructure. As a consequence, the attestation of the VM would be invalid. The process of attesting both VM and NC is called ‘iterative attestation’.

It should be noted that an OpenPTS installation can act simultaneously as client and server. This feature is exploited in order to implement the iterative attestation of the Eucalyptus NC and SC. The iterative attestation uses the TPM support patched into QEMU so that:

1. The vTPM of the VM receives the TPM_Quote request from the OpenPTS client of the VM;
2. The OpenPTS client of the VM, acting as an OpenPTS server, requests that the OpenPTS client of the NC issues a TPM_Quote request to the hardware TPM of the NC.

The same iterative attestation process is applied to the Eucalyptus SC. The OpenPTS installed on the VM acts as the server requesting the attestation of the SC.[3] The three parts of the PCR values returned by the three TPM_Quote requests and depicted in Figure 3, are fetched and returned to the VM OpenPTS client in a single attestation session. In Figure 4, the solid arrows denote the TPM_Quote sequences when the OpenPTS client in the VM receives an attestation request from the user,[4] while the dashed arrows denote the measurement sequences.

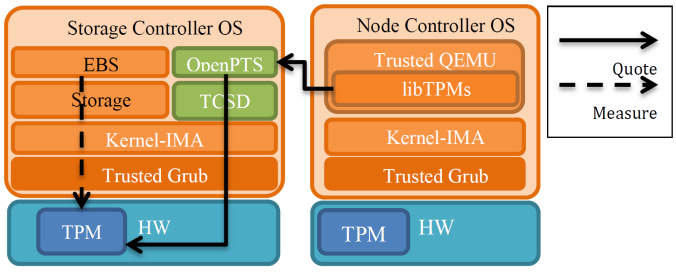

Figure 5. Storage Controller Attestation.

Figure 5 depicts the architecture of the SC. Most components are the same as on the NC. As shown with the dashed arrow, the SC and EBS volumes are measured by the Kernel-IMA, together with all system programs and configuration files. An OpenPTS client is deployed in the SC to report the attestation ticket of the SC to the OpenPTS on the VM during the user attestation session (shown with the solid arrows in Figure 5).

IV. Use Case Implementation

Figure 6 and 7 illustrate the myTrustedCloud use case. A group of Distribution Network Operators (DNOs) upload, validate and store a data set into the storage facility offered by the chosen cloud infrastructure (Figure 6). Another operator, the National Grid (NG) in this case, uses the same cloud facilities to collect, merge and convert the data set uploaded by the DNOs (Figure 7). As described in Section II, currently this operation is repeated every month but future requirements would see this timing reduced to one merge less than every ten minutes.

The use case adopts the Open Grid Systems' Cimphony application as an intermediary between data origins and recipient. Cimphony is a network model data visualization and analysis tool. As the tools used for power network analysis often work with differing data formats, the Siemens PSS/E file format [23] was chosen as a data format for export, due to its wider usage.

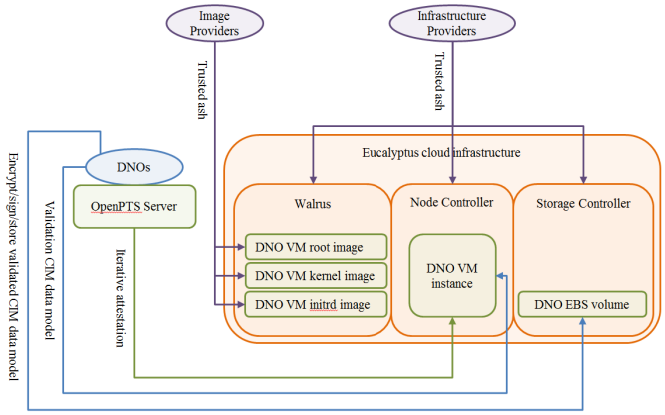

Figure 6. myTrustedCloud DNO use case.

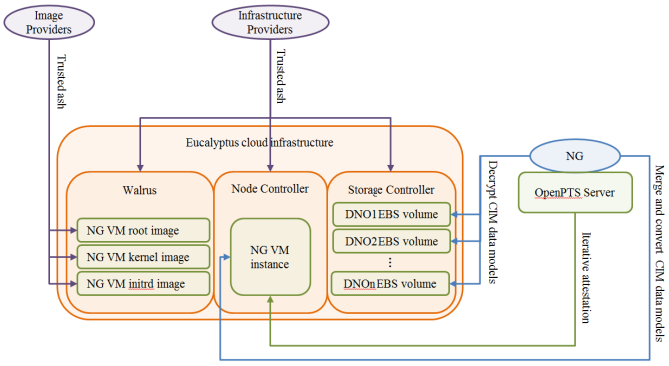

Figure 7. myTrustedCloud NG use case.

In the use case, Cimphony performs three operations upon two different test data models:

1. Validate two different network models presented in CIM format;
2. Merge these data models;
3. Transform these data models from CIM to PSS/E format ready for export.

The PSS/E files exported from the myTrustedCloud platform may then be imported into the power analyses tools used by NG. The process exemplified here however, would be common to the needs of other Transmission System Operators (TSOs) and similar to the needs of pan-European transmission network coordination such as Coreso [24] or TSC [25].

Two sets of root, kernel and initrd images are uploaded into a trust-enabled Eucalyptus cloud infrastructure. One image is to be used by the DNOs, the other by the NG. Both root images contain the Cimphony application.

Network models are uploaded by the DNOs into EBS volumes, one volume for each model uploaded. Each volume is tagged with the name of the DNO and the time and date stamp of the upload. Each DNO has read/write access only to its volumes while the NG has read access to all the volumes owned by the DNOs. From the Eucalyptus point of view, this means that the NG is the administrator of the cloud infrastructure. Other cloud platforms – e.g. Amazon WS –

[3] Please note that all the Eucalyptus controllers could be iteratively attested in the same way.

[4] It should be noted that the user will place such a request through an OpenPTS server hosted on the user platform or on a third party verifier platform.

may offer a richer authentication and authorisation model but from a functional point of view, the choice of the cloud infrastructure makes no difference for the design of the use case.

From a security point of view it is assumed that:

1. Each DNO and NG own a private/public key pair;
2. NG and DNOs share their public keys;
3. NG and DNOs have access to a trusted hash value of their respective root, kernel and initrd images. The hash value is published encrypted and signed by the images provider(s);
4. NG and DNOs have access to a trusted hash value for the NC and SC of Eucalyptus provided by a trusted Eucalyptus distribution authority;
5. The Eucalyptus infrastructure is integrated with the Trusted Computing technology as described in Section III. In this way, the hash value of an instantiated VM and its kernel are available to the VM owner; the DNOs and NG can run/interrogate an OpenPTS server in order to get the hash value for the NC, SC, Cimphony and every other relevant software installed either in their VMs, or on the systems running the NC and SC.

Each DNO workflow is as follows:

1. instantiate a VM from a root image stored in the Eucalyptus Walrus storage service;
2. Verify that the VM, NC and SC can be trusted by means of the OpenPTS server and the trusted hash values published by the image providers and the cloud distributor;
3. Upload the network model;
4. Validate the network model by means of the Cimphony software;
5. Encrypt and sign the data set of the validated model respectively with the NG public key and DNO private key;
6. Create and attach an EBS volume to the current VM tagging it with the name and time/date stamp.
7. Store the encrypted model into the EBS;
8. Unmount and detach the EBS volume;
9. Destroy the VM.

The NG workflow is as follows:

1. Instantiate a VM from a root image stored in the Walrus storage service;
2. Verify that the VM, NC and SC can be trusted by means of the OpenPTS server and the trusted hash values published by the image providers and the cloud distributor;
3. Mount each DNO EBS volume tagged with a time/date stamp less than an hour old;
4. Decrypt and verify the signature of each model contained in each mounted DNO EBS volume;
5. Merge all the DNO models;
6. Convert the resulting model from CIM to PSS/E format;
7. Elaborate the model on the VM or download it through an encrypted channel;
8. Destroy the VM.

It should be noted that all the operations involving trust verification, encryption, decryption, signing and signature verification, creating, attaching, tagging, mounting and unmounting EBS volumes can be automated. Furthermore, Cimphony has been modified to provide feedback information to the users about its own trustworthiness and that of the infrastructure on which it is running.

The trustworthiness of the VM and its kernel is assessed by comparing three hash values: that of the running VM/kernel, that of the VM/kernel images stored in the Walrus storage service and that of the VM/kernel image as supplied by the image providers. This comparison is secured by the public/private key infrastructure and allows trusting the running image even on a potentially insecure cloud infrastructure. While a trusted cloud infrastructure can still be vulnerable to attacks and bugs, the possibility to verify that the running VM has not been altered from the one supplied by the VM image providers guarantees that the VM itself has not been compromised independently from whether the underlying cloud infrastructure has been compromised.

V. Performance Analysys

The integration of trusted computing and cloud computing introduces performance overheads in the system bootstrapping and remote attestation procedures. The bootstrapping of a trusted operating system includes the building of a chain-of-trust. This is a time consuming operation in which the IMA-enabled kernel measures the hash value of each executable, kernel module and relevant configuration file during the bootstrapping procedure. In the case of the prototype presented in this paper, a chain-of-trust must be built for the physical machines on which the NC and SC are executed and for each VM instantiated on the NC. The added overhead on their boot time is measured with bootchart [26]. In the case of the VM, the bootstrap time is calculated from when the QEMU command is invoked to when the logging console of the VM is ready. The time taken to copy the VM images from the SC to NC is ignored as it is unaffected by building the chain-of-trust. The physical machines of our testing infrastructure are equipped with Intel Core i5-2400 Quad, 16GB of ram, a TPM module integrated into the motherboard and deployed with Ubuntu 11.04.

Table 1 shows the results of the performance measurements. The bootstrapping delay of trusted NC and SC is significant, requiring around three times more than their untrusted counterparts. However, the operational impact of such overhead on a typical deployment environment is negligible as it is experienced only when a physical machine is rebooted. This is usually a scheduled event and high-availability and service redundancy are usually adopted in order to manage unscheduled outage. Conversely, the bootstrapping delay measured when instantiating a VM is minimal. This is due to the adoption of software-TPMs, within which the TPM_Extend operation is much faster than its hardware-TPM counterparts.

Concerning the overhead of the remote attestation procedure, the three-step iterative attestation schema adopted in myTrustedCloud takes around 1:18 minutes (N = 15, SD = 2.5s). The attestation delay is due to the time required for generating and verifying the integrity reports, together with the time required for quoting and verifying the PCRs. Assuming that the attestation is performed regularly, as the states – i.e. the loaded software components – of the controller nodes of a production cloud infrastructure seldom change [27], the PCRs and integrity reports are going to be the same most of the times. As a consequence, the attestation throughput can be improved by first quoting and verifying the PCRs and then comparing them with the latest values. In this way, the integrity report is only generated and verified when the PCRs are changed. Adopting this approach generally allows to reduce the attestation overhead to no more than 10 seconds.

System type	Boot time (minutes)	Standard deviation (seconds)
Trusted NC	3:18	3.8
Untrusted NC	0:57	4
Trusted SC	3:05	4
Untrusted SC	1:02	0.5
Trusted VM ()	1:01	1.5
Trusted VM with disk hash (2gb)	1:07	1.3
Untrusted VM	0:57	1.4

Table 1. Boot time of trusted and untrusted systems. Number of observations: N = 15. Every measurement has been performed on a standard installation of Eucalyptus 2 as packaged and provided by Canonical for Ubuntu 11.04. No additional load was present on the infrastructure when the measurements were performed.

VI. Threats and Vulnerabilities Analysys

A full threat and vulnerability analysis for cloud computing is out of scope for this paper and a production-ready, trusted cloud infrastructure for the energy sector would have to meet several types of security requirements of which trusted computing is one important example. The presented approach concentrates upon mitigation of the risks which would arise from the insertion of rogue analysis code into the cloud, which would have consequences for the confidentiality of the data and the integrity of results.

With a Trusted Computing infrastructure, a remote party can undeniably attest the current configuration of a platform, including the version of all the loaded software components and their configurations. As a consequence, changes made by malicious administrators or privileged malicious users can be genuinely recorded. Moreover, Dynamic Root of Trust for Measurement (DRTM) [28] can be used to establish the runtime chain-of-trust so to protect critical components on the system against runtime attacks, e.g. the hypervisor. However, for unprivileged malicious users it is not feasible to measure and record the configurations of all the VMs running on the same platform with the target user VM. Nonetheless, the genuine enforcement of the countermeasures against these attacks, e.g. side-channels, can still be attested. As usual, it is assumed that hardware cannot be manipulated without being identified, i.e. the TPM cannot be tampered with, and insider physical attacks such as the cold boot attack are not considered.

The trusted computing infrastructure as presented in this paper could suffer from a well-known potential privacy leakage. The problem is that the remote attestation reveals the detailed configuration of a platform. This information can easily be obtained by attackers and used for system profiling. Property-based attestation [29] has been proposed as a solution to this problem. In this type of attestation, platform configurations would be mapped to a set of properties on which attestations would be enforced. In order to adopt a property-based attestation, a set of properties mappings for all the software components used within the cloud infrastructure should be provided. This set of properties should also be certified by a reputable Trusted Third Party.

The so called time-of-check-to-time-of-use (TOCTOU) is another well-known potential security issue of a trusted computing platform. As seen in the previous sections, in a trusted computing scenario software components are measured before being loaded. As a consequence, malicious code directly injected into memory through runtime attacks cannot be measured – e.g. stack-overflow attacks. The DRTM discussed above mitigates this kind of attacks. However, DTRM requires modifications of the cloud software stack, modifications that would have to be considered for a production-ready, trusted cloud infrastructure.

A natural piece of further work will be to build the infrastructure necessary to log and evaluate the measurement data generated and received, and to use it systematically to ensure the provenance of the reported results. There is every reason to presume that this is feasible, but only a large-scale deployment can properly validate it.

VII. Conclusions

The myTrustedCloud use case grounded on the integration of Trusted Computing Components with a cloud platform shows how Cloud Computing can be a viable solution for communities with high data integrity requirements. The security models adopted by the public, private or hybrid cloud systems currently available are not suitable for the requirements of the energy industry and their use cases. As such, we have developed a prototype of trust-capable cloud infrastructure based upon a publically available cloud infrastructure solution. The solution developed through the project has been integrated in such a way that it interacts with the lowest level of virtualisation software. This has meant that there has not been a need to alter any of the Eucalyptus software and so the scheme chosen will work for any of the previously mentioned open source cloud solutions.

The implementation of the use case described in Section IV showed that a trusted-enabled Cloud Computing platform is a viable solution for the security requirements of the UK transmission and distribution network business sector. In a model based on the greater interoperability between active distribution and transmission networks the trusted cloud

infrastructure that we have developed allows in principle a dynamic, fast and secure management of the information generated and elaborated on both sides. Further research and development is required in order to translate the proposed prototype into a production infrastructure.

REFERENCES

[1] R Lerner, Amazon Web Services, Linux journal, ISSN 1075-3583, Nº 143, 2006 , pages. 20-22.

[2] Microsoft Azure, http://www.microsoft.com/windowsazure Accessed 26th August 2011.

[3] D. Nurmi et al., "The Eucalyptus Open-Source Cloud- Computing System," Cloud Computing and Applica-tions 2008 (CCA 08), 2008.

[4] OpenStack.. http://readwriteweb.com/cloud/2010/07/openstack-rackspaceand-nasa-n.php

[5] J. Fontán, T. Vázquez, L. Gonzalez, R. S. Montero, and I. M. Llorente. OpenNEbula: The open source virtual machine manager for cluster computing. In Open Source Grid and Cluster Software Conference - Book of Abstracts, San Francisco, USA, May 2008.

[6] Sky Computing, Keahey, K., Tsugawa, M., Matsunaga, A., Fortes, J. IEEE Internet Computing, vol. 13, no. 5, September/October 2009.

[7] Massoud Amin, S.; Wollenberg, B.F.; "Toward a smart grid: power delivery for the 21st century," Power and Energy Magazine, IEEE , vol.3, no.5, pp. 34- 41, Sept.-Oct. 2005.

[8] Taylor, G.A.; Irving, M.R.; Nusrat, N.; Liao, R.; Panchandaram, S.; "Developing novel information and communications technology based solutions for smart distribution network operation," *Universities Power Engineering Conference (UPEC), 2010 45th International* , vol., no., pp.1-6, Aug. 31 2010-Sept. 3 2010.

[9] Trusted Computing Group, Trusted Platform Module Main Specification, Part 1: Design Principles, Part 2: TPM Structures, Part 3: Commands , October 2003, Version 1.2, Revision 62. http://www.trustedcomputinggroup.org

[10] Derek Gordon Murray, Grzegorz Milos, and Steven Hand. 2008. Improving Xen security through disaggregation. In Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments (VEE '08). ACM, New York, NY, USA, 151-160.

[11] Engineering Attestable Services, John Lyle and Andrew Martin, Proceedings of the 3rd International Conference on Trust and Trustworthy Computing. Pages 257—264. Springer. June, 2010.

[12] "The Grid Code, Issue 3, Revision 12," National Grid Electricity Transmission plc, available at: http://www.nationalgrid.com/uk/Electricity/Codes/gridcode/gridcodedocs/

[13] DIgSILENT Power Factory Manual, Version 14.0, DIgSILENT GmbH, Gomaringen, Germany, 2009.

[14] N. Hargreaves et al, Developing emerging standards for power system data exchange to enable interoperable and scalable operational modelling and analysis, 46th International Universities' Power Engineering Conference (UPEC2011), Soest, Germany, 5-8th September 2011 (submitted).

[15] Trusted Computing Frameworks.

[16] R. Toegl et al., "Towards Platform-Independent Trusted Computing," Proc. 2009 ACM Workshop Scalable Trusted Computing (STC 09), ACM Press, 2009, pp. 61–66.

[17] Fabrice Bellard, QEMU, a fast and portable dynamic translator, Proceedings of the annual conference on USENIX Annual Technical Conference, p.41-41, April 10-15, 2005, Anaheim, CA.

[18] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. KVM: The Linux Virtual Machine Monitor. In Proceedings of the Linux Symposium, pages 225--230, 2007.

[19] Canonical Ltd. http://www.canonical.com

[20] Shuttleworth, M. (2006). *Ubuntu: Linux for human beings*. http://www.ubuntu.com/

[21] Reiner Sailer, Xiaolan Zhang, Trent Jaeger, and Leendert van Doorn. 2004. Design and implementation of a TCG-based integrity measurement architecture. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13* (SSYM'04), Vol. 13. USENIX Association, Berkeley, CA, USA, 16-16.

[22] OpenPTS. http://sourceforge.jp/projects/openpts

[23] http://www.energy.siemens.com/hq/en/services/power-transmission-distribution/power-technologies-international/software-solutions/

[24] Coreso, http://www.coreso.eu/

[25] http://www.tso-security-cooperation.eu/en/index.htm

[26] http://www.bootchart.org

[27] John Lyle and Andrew Martin. 2009. On the Feasibility of Remote Attestation for Web Services. In Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 03 (CSE '09), Vol. 3. IEEE Computer Society, Washington, DC, USA, 283-288.

[28] Paul England. 2008. Practical Techniques for Operating System Attestation. Lecture Notes in Computer Science, Trusted Computing - Challenges and Applications. Vol. 4968, 1-13.

[29] Ahmad-Reza Sadeghi and Christian Stüble. 2004. Property-based attestation for computing platforms: caring about properties, not mechanisms. NSPW '04 Proceedings of the 2004 workshop on New security paradigms.