Trust in Distributed small sized Data Centers

Nicolai Kuntze, Carsten Rudolph
Fraunhofer Institute for Secure Information Technology (SIT)
Rheinstrasse 75, 64295 Darmstadt, Germany
{nicolai.kuntze|carsten.rudolph}@sit.fraunhofer.de

Abstract

Peer-to-peer (P2P) networks have large advantages over server-based solutions in terms of efficiency for bandwidth consumption and server workload. This is particularly relevant for the distribution of large volume content like multimedia data. This paper proposes a security architecture that provides essential security functionality for a platform enabling commercial P2P applications. One main characteristic of such a platform is, that the devices of the P2P network are not under the physical control of the owner.

The proposed security and trust architecture includes solutions for integrity protection of data as well as for software on the device, exclusion of manipulated nodes from the network, and isolation between applications by different stakeholders residing in parallel on the same platform. All solutions can be build on existing secure hardware anchors as provided by the Trusted Platform Module (TPM) and its certification infrastructure.

1. Introduction

Content distribution and service offering in the Internet has relied on a client-server model. This model is dominant for most of the known applications like web sites, file transfer protocols, or electronic mail. Aside this approach of service delivery distributed solutions arose like distributed caching, Content Distribution Networks (CDNs) [4], and more recently peer-to-peer (P2P) networks.

For the network operators (NO) it is desirable to establish distribution networks based on nodes located at the edges of the network in the households of the customers. This allows for optimised bandwidth utilisation in the core network by applying optimised caching and data propagation methods and allows for less complex server systems to be maintained. Another aspect is an improved Quality of Experience (QoE) considering a video on demand use case. There, the CDN is able to cache the data expected to be viewed on the device located at the user or in his neighbourhood. To accomplish this task certain information about the user and his habits are required to be known at least to the individual device to manage caching algorithms.

Today's users expect a high integration of different services offered by one platform. Therefore different offerings to the user are hosted in the same device requiring isolation properties to each other as these are provided by different stakeholders. Sharing of one environment between different stakeholders is widely adopted. Web based storage services or resizeable computing capacity as offered for example by Amazon are examples for it. Grid and cloud computing concepts are building the base for such services.

Introducing architectures aimed at establishing a trustworthy base in the domain of the end user are not new to the content industry. Existing approaches like set-top boxes for pay TV [6], DRM used in Microsoft ZUNE or iTunes are examples for systems owned by the end user but enforcing the rights of the content owners. In the scientific community distributed computing approaches like BOINC [1] are established providing a good level of reliability by processing a certain part on different clients to prevent forging of results by single clients.

The EU funded project NanoDataCenters (NaDa) aims at developing a platform supporting access to virtual goods like videos or music by establishing a P2P network build up on nodes located at the households of the end users. These nodes belong to a NO and are under the administrative control of him. The NO is not offering the content on his own but offers a platform for content and service providers. These establish software on the platform distributing and offering the respective content to the end user. Therefore each node is regarded as a multi-stakeholder system.

The architecture has to respect that it is to be applied on a variety of low profile nodes. The hardware available in this area bears limitations in terms of computational power and the abilities of the chip-sets. For example hardware assisted virtualisation as it is known from the PC domain is not an option nowadays. One important requirement given is the isolation of the stakeholders and their data stored from each other and from attackers outside the node.

Another underlying decision is that the NO owns the nodes. The end user only provides for an environment, namely energy and connectivity, where the node can run. Media created and provided by a content provider are distributed using a P2P network. The content is dispersed to the nodes and redistributed. It will happen that content is stored on devices not delivering the content directly to a specific end user but storing it for others. The decision on the storage strategy is based on providing an optimal QoE.

As the content stems from different stakeholders each of them has different methods and requirements for the security of the their content. Those protection mechanisms have to be adequate for the various possible types of content like videos, music, or games. The hosting node provides for a common base with a defined security, monitoring, and communication architecture shared by the stakeholders. These common services offered to the applications of the stakeholders define the runtime environment.

The service for the end customer is the content delivered by the stakeholder to him. This content is protected by the system using watermarks or other means. As the content is not delivered directly by the content provider but using P2P schemes protecting the content at the side of the content provider's server is not applicable. The node has to provide for the mechanisms. It is to be noted that such a platform is not only applicable for multi media distribution but also interesting for e.g. distributed computing for medical purposes or backups. Besides the content providers also the user is represented in the node. Each stakeholder, including the end user, stores and controls specific information describing an identity. In case of the end user certain accounting and usage information may be stored allowing for payment processes or service customisation.

This paper presents the security architecture for the NaDa project. Currently successful application of P2P technology is concentrated on the distribution of free content. This architecture considers the unique challenges that derive from a distributed system that is operated in the end user's domain but using a hardware anchor for trust establishment. We show how this trust anchor in combination with a node architecture allows for the sufficient level of trust.

In section 2 security requirements are discussed and trust model is presented. Section 3 presents the required basics in Trusted Computing (TC) that provides the required roots-of-trust for the security architecture presented in section 4. This paper concludes in section 5.

2. Security Requirements Analysis

In this section the underlying trust level is presented and the relevant security requirements are discussed on an architectural level and on the level of the business process of the use case scenario (see also [7]). Perils resulting from e.g. faults in the software are not covered within this analysis. The NaDa security architecture has to cover the full business process to allow all involved stakeholders to put trust into the platform. Trust on a technical level just says that a system will always behave in an expected manner. This trust definition does not include any judgement on the actual behaviour of the system e.g. in terms of good or bad behaviour. If there is a failure in the system this definition of trust only states this failure is not the result of a malicious change to the system. The system will behave exactly in the implemented way including the failure.

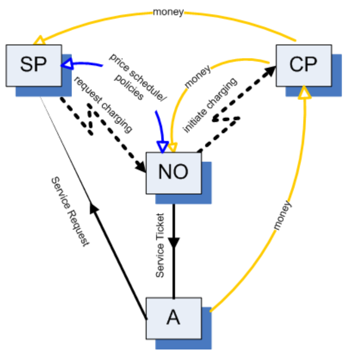

Figure 1. Roles in the scenario

In the above introduced use case of content distribution using P2P networks mainly the roles of the NO, content/service provider, and the end user are relevant. An identity and charging provider are additionally required to describe the business process. Figure 1 presents a conceptual view on the roles. The Agent (A), representing the customer, consumes services offered by the service provider (SP). In our case there are SP_1 to SP_n. The NO mediates between them and provides identities for the As acting here as an identity provider. The charging provider (CP) receives the charging requests and transfers money between the stakeholders. The actual transfer of money is not within the scope of this paper. A similar scheme is also presented in [5]. Due to the various applications possible running on a node different security requirements can occur. Only a subset of these requirements can be provided by the platform and the P2P communication system. Other requirements need to be taken care of by the applications themselves. These requirements are either identified based on static trust relations on the organisational, contractual, or infrastructure level, or for single services that are part of the implementation. In the case of decentralised distribution structures it is not trivial to derive the correct combination of security mechanisms that satisfies all requirements of the different partners. Therefore, it is necessary to precisely define security requirements. A full formal framework for the specification of security requirements is out of the scope of this paper, but we can refer to the generic framework for security requirements in [3]. In this paper we concentrate on those requirements that can be supported by the underlying platform and that provide a security basis used by applications to build their own security system.

Aspects of information governance like authenticity of a node acting within the network, integrity of this node, integrity or confidentiality of data that is transported between nodes, or the enforcement of certain licenses belonging to the content distributed are included in the set of security requirements for which basic security mechanisms can be provided. We concentrate on the central requirements, namely the security properties of isolation of the different stakeholder applications, the identity of each node, status of the node, and the integrity of the content and introduce them.

Each stakeholder hosted has his own interests and stores sensitive information on the node to properly run his software. Different types of information require different levels of protection. Some information is crucial, like keys, for the operation of the system. Others might reveal important facts of the stakeholder. Especially for the end user the protection of his privacy relevant data like gathered usage data is of importance. Aside the *isolation* on the level of data access and information flows the architecture has to provide for mechanisms to establish isolation on the level of the usage of resources. This second dimension of isolation is required to guarantee the assured service level for each stakeholder. It is to be noted that isolation is always only provided between the stakeholders. The underlying system of the node is able to manipulate the stakeholder applications. Therefore each stakeholder has to trust in the owner of the node.

Authenticity of a particular action is satisfied for one node N if N can deduce that this action has occurred from its knowledge about the global behaviour of the CDN and the view of the current behaviour. The *identity* of a node issuing requests to other nodes that lead to actions is the anchor for the authenticity of this action. Therefore, this identity needs to be protected against forgery or manipulation even from a attacker who has physical access.

In addition to the identity of the node also the *state* of the node needs to be considered in the decision of the authenticity of the corresponding action. All involved nodes are autonomous operating located at the end customers and are not under the physical control of the owner. Possible attacks on the software either by direct physical access or by exploiting weaknesses of applications or OS are to be considered. A measurement system for the analysis of the actual state of the running software is required first to allow the individual node to detect tampering and second to report this state securely to the connected nodes. If the node is tampered it has to be assured that relevant information stored at the device is not accessible. Communication has to assure that based on this reported state only nodes in the allowed state are part of the network.

The *integrity* of the content stored on each node by the NO, stakeholders, and the end user has to be secured against first, undetected modification by direct physical attacks should not be possible. Such a physical attack could e.g. be direct access to the storage of the node in a power down mode of it. Second, attacks by external attackers using software vectors to modify the content. A third scenario is the introduction of malicious or manipulated content by e.g. the legitimate end user with the aim to distribute illegal content. Content is distributed in data chunks between the involved nodes, so that the file arrives from different nodes at the target node and is then assembled to one file or stream. Many applications allow disruption of a data stream, e.g. time shift or skipping of parts of the play back. Thus, integrity protection relying on a linkage between the individual chunks is not always possible and can therefore can not be part of the architecture. Individual applications can introduce additional integrity protection on the application layer.

From the view of the architecture established on the nodes the only *confidentiality* requirement is concerned with the isolation between different applications running on the node. On each node different kind of types of information are stored and require different levels of protection. The used platform and its OS include data concerning the network, its interaction, and other parameters required to run the network. For each stakeholder certain secrets are stored. If a stakeholder considers certain data as to be protected he has to take appropriate measures. The proposed security architecture provides mechanisms for confidentiality protection that can be used and applied by applications. This can guarantee the isolation between different applications.

3. Trusted Computing

As shown in section 2 protection of the node's identity and the secure reporting of the system state are important security requirements. Furthermore, the lack of control on the physical access to the node induces strong requirements on the protection level. One possibility is to root security mechanisms in strong hardware security anchors. TC [10] offers such a hardware root of trust providing certain functionalities designed to approach these requirements. In this section these functionalities are introduced.

TC as defined by the Trusted Computing Group (TCG) are computer systems extended by additional components which shall bring trust to the computing environment. Trust means that components of the system always work as implemented. Most important hereby is the specification of the TPM providing a hardware chip hard-wired to the computer platform. It implements basic cryptographic functionality like SHA-1 calculation, message digest creation, random number generation, creation of RSA key pairs, and a RSA engine for encryption and signing. Realized as an independent hardware module, it can provide protected capabilities allowing to shield secret data efficiently. This implementation also allows for in depth testing and validation of the soft- and hardware. The TCG defines three different roots of trust on which the trust to the whole system is built on.

The Core Root of Trust for Measurement (CRTM) [9] is implemented e.g. as an extension of the BIOS. Its duty is to perform measurements of system components involved in the boot process. Measured components then can perform measurements of other components involved in the next stage of booting. Through this principle of transitive trust, trust in the correctness of the measurement values can be passed on to the OS and the software executed.

The second root of trust is the Root of Trust for Reporting (RTR). One of the aims of TC is to enable computer systems to proof to other platforms that it is in a trusted state. Therefore the results of measurements of system components have to be presented to the remote platform. To guarantee the genuineness of these data, they are signed. For this purpose every TPM contains a 2048 bit RSA key pair, the Endorsement Key (EK), which is generated before shipping. The EK, together with an EK Credential, represents the identity of the platform. Pseudonymous representatives of the EK, so called Attestation Identity Keys (AIK) are used for signatures, for example of measurement results used by the remote party to verify the correctness of the desired state (Remote Attestation or RA) [2].

The third root of trust is the Root of Trust for Storage (RTS) with the purpose to establish a secure storage for cryptographic keys. The RTS is implemented by the Storage Root Key (SRK), a second RSA key pair stored in the non-volatile memory of the TPM. The SRK never leaves the shielded location of the TPM. That allows building a hierarchy of keys, with the SRK as the root, in which direct successors are protected with the SRK. These keys on their part can protect any number of other keys. These keys allow to bind data to a device or even to a particular state of it.

4. Concept / Architecture

The aim of NaDa is to provide a trustworthy base for the distributed offering of services. To meet this goal each involved stakeholder must be able to lay trust in the overall architecture provided by the project. As the overall network is build upon and by the nodes located at the homes of the customers the architecture can not assume each node as secure and trustworthy per se. Therefore, the architecture consists out of platform design, mechanisms required in the communication, and methods used in the particular use cases. In this paper the use case considered is the distribution of virtual goods like multimedia content. Other use cases, e.g. the distributed processing of computational tasks, can also be based on the underlying architecture proposed. Additional means securing the confidentiality and integrity of the service are stakeholder and use case specific.

The tasks for the platform used to establish the distributed environment is twofold. First, the stakeholders represented by their software need to be isolated from each other in terms of access and resource control. Second, a tampered or malicious device needs to be excluded from the P2P network. A node is defined as not acceptable if either it can not provide a valid identity or is not able to provide proof that it complies to a defined state. The identity of a node is derived from its TPM identity, the EK.

The detection of changes to the state of the node is based on the Remote Attestation abilities offered by TC. As introduced in section 3 the chain of trust is established in the CRTM and spans from there into the running system. Within this authenticated boot all components before the start of the OS are included e.g. the loader and the running kernel. The OS is responsible to extend the chain of measurements into the application domain. One challenge here is the application of hash based algorithms to the different execution orders possible as each permutation of the execution results in different measurement values. One approach feasible is to measure a whole system including all executable files or even the whole partition resulting in one measurement value and then to start this system from this partition. Applying this scheme leads to a unique and reproduceable value for the whole system that later on can be stored in a data base (DB) of reference values.

To exclude a node from the network if it does not comply to a defined state two approaches are possible. First, the device detects tampering and deactivates itself. However, considering a skilled attacker with physical access it is possible that the node will not be able to produce a reliable decision or any decision at all. This can be achieved by either manipulating the reference values used or by attacking the decision algorithm. the second approach uses the RA mechanisms that allows an external entity to get evidence on the actual system state and to verify it. As the hardware core roots of trust are very difficult to attack and local attackers cannot manipulate the reference values and decision procedures on the remote machine, the verification of measurement values in RA is very reliable. In combination with means for a secure platform design both on the hard and software level the required level of trust can be provided.

As the measurement of a system provides proof of the state it is required to provide a system design that supports the RA on the one hand and also provides a secure and trustworthy base. In the following first platform design patterns are presented. Then impacts on the communication protocols between the nodes are discussed on a high level. Furthermore, the integration and use of the identity given by TC into the architecture is important. This section concludes with a brief discussion of the specific use case of distribution of virtual goods.

The security of the latter presented design approaches for the platform depend on the OS underlying them. It is to be noted that we assume that this part of the system is well tested. To gain the required isolation properties protecting the stakeholders from each other and also ensuring the system behaviour one approach is to define the shape of the environment of the stakeholder application. By defining the environment resources consumed by the applications are restricted and communication can be limited. This approach requires precise policies for each stakeholder application.

As the node can not derive these policies on its own they are required to be provided either with the application or by a certain central authority. A second factor are resource utilisation contracts between the stakeholder and the NO. In such a contract the NO assures to a certain stakeholder the availability of specific resources. Policies and contracts are signed either by the stakeholder or the NO and need to be enforced by each node in the network. Based on the signature the node decides if a certain application can run on the platform. In this decision also existing other applications and their requirements are to be considered. These policies and contracts are to be included in the attestation of the system forming a list of measurements on its own as they induce modifications to the behaviour of a node.

Another approach is the evaluation of the running software. Virus scanner use patterns to detect malicious software and modifications. By this the integrity of the system is ensured during runtime by an inspection. This black-list approach cannot protect a system from unknown attacks but as the NO knows which software is allowed on the nodes a positive list can be compiled and distributed to the nodes as a reference. Instead of using patterns or other intrusion detection mechanisms a defined state can be testified by applying RA. If a node detects a compromised stakeholder environment it can be reported to the NO. Using these reports the NO is then able to determine the cause and may be able to detect previously unknown threats.

In practice it is complicated to perform these inspections on running programs. Existing protection mechanisms either in a virtualised system or within a secured OS have to be changed to allow the scanner access to all executable software. This would bring the inspecting application into a security sensitive position as it is now able to access large parts of the systems. If the inspector is compromised he could modify all systems he has access to. This security hole is not necessary if the scanner is only working on copies of the actual system state of parts of the node.

The origin and integrity of the system used can be checked by the underlying entity, e.g. hypervisor [8] or boot loader, by using OS creation certificates. Each environment used by the device origins at the operator and requires credentials stating the origin and integrity. The origin is stated by a signature that is associated with the environment and contains reference values for deployment and runtime. The reference value may be defined by a hash value on the image of the environment or by providing evidence on a file level.

As the state of the node at the deployment time may change due to changes during operation also appropriate reference values for the lifetime of the node are required. These changes may occur e.g. due to downloadable code or updates. Therefore after deployment only these values are of importance for the attestation of the system state. One possible set of reference values are the hash values of the applications in contrast to the one value of the whole image during deployment.

Separation of executable code and data provides a clear split between the parts that are to be attested and verified and the data they operating on. This split can be performed in different ways. One approach is to split the partitions and to store the executables on a write protected one. If virtualisation is available certain functionalities can be used. Virtualisation allows in principle to restart an environment in a given state. This can be done at any point in time. If the whole state of the environment is stored operation can be resumed and only a small amount of work has to be redone. Be restarting the environment from time to time the maximum attack time is limited and after restart the system is compliant again. The old environment that was terminated can then be analysed in the back ground.

Communication protocols that are developed in the project have to provide for authenticity and content protection in the communication between the nodes. To testify the integrity of the individual nodes to each other statements of integrity based upon attestation have to be included where necessary. From the security perspective update and software distribution towards the nodes are of special interest. Content protection can be based upon stream ciphers using keys that are bound to the inbound node. Bound in this context denotes that a certain information is only accessible if the node is in a certain defined state. This can be achieved by the TPM_seal or TPM_bind commands. Before a node is allowed to interact with the other nodes a proof on its integrity is required.

Direct mutual attestation between the nodes may be not wished as it is expensive in terms of computational power and time consumed. Most relevant is the functionality of the verifier as this is part of the NO functionalities. He has to vouch for the integrity of the nodes as he also defines the desired state. Verifying a specific node requires checking a DB of reference values. This DB tends to grow and needs to be kept in sync with a central DB if we assume that this verification process is also distributed. A single ticket could replace the verification of the collected data by stating that the hash value of the reference values can be regarded as correct. In this case the verification of the collected reference values is only done once.

If a certain node can be regarded as trustworthy the NO can distribute a software representative located in each node performing the verification and assertion of tickets. For explanation of the ticket system we assume node B wants to communicate with node A, e.g. in order to transfer some content data to B. Before node B considers node A as trustworthy A has to prove to B that its state is compliant to the expected state defined by NO. A produces to B a ticket issued by the NO, together with a TPM quote (signed by an AIK), and the AIK certificate. Due to the NO ticket B does not have to check the reference values but only the link between them (represented by the ticket), quote, and

AIK certificate. Therefore no local DB and no additional communication with a remote DB is required for B to verify it. The AIK (certificate) states that this is a TPM equipped device, but cannot deliver informations about the latest software changes. Due to the P2P structure of the overlay network the role of the NO representative can be included in a security (trust) super node that manages for all devices the ticket issuing. Fall back options are required here and assumed as part of the P2P protocol. To enhance the runtime behaviour of the system the attestation scheme can be limited according to policies e.g. that it is performed only once for a new communication partner or when a new content is stream to a certain node.

A new ticket granting needs to be done as part of the update procedure. The initial ticket is established during the manufacturing process of the node. There the manufacturer can directly state that this system as delivered is in the desired state. Each update or added functionality changes the system state Also the revocation of certificates requires new tickets issued by the NO or its representatives. As changes are propagated also by the P2P infrastructure a scheme should take advantage from this but allowing for a distributed scheme without a central update repository. In this scheme an update is only accepted by a node if the node offering the update can provide evidence on his trustworthiness (using the tickets and TPM quote) and a valid signature on the update that it derives from the NO. After applying the update the node offering the update also produces the new ticket for the updated node after verifying the new state by performing an attestation and if successful granting a new ticket in the name of the NO. The updated client can now propagate the update as well.

Another issue during the proposed update scheme is the protection of data stored on the particular node that is bound to a certain system state. Applying updates leads to a change of the system which renders existing content useless. It can no longer be accessed. To solve this one should keep in mind that the new state can be determined either by the issuer of the update or the node offering the update in the update process. Therefore an update has to provide for the new system state or the data needed to predict it. An update also provides for authenticity and integrity. We assume that all data bound to a certain state is symmetrically encrypted. The respective key is bound to the platform using the TC seal functionality. These symmetric keys are re-sealed to the new system state before the system is restarted into the new state. A possible roll back can be done by storing the old data structures containing the sealed keys and keeping a backup of the old system state.

5. Conclusion

The overall concept of security and trust in the distributed creation of value has the potential to spur the usage of unused resources. In the example of the presented use case of multimedia distribution the utilisation of the local bandwidth is improved. This leads to a higher revenue for the network operator and also a better energy to effect ratio.

This paper introduced a security architecture for P2P distribution of large volume content. This architecture is based upon hardware mechanisms like the introduced TC allowing for strong assertions on the security of the used anchors of trust. These assertions are required as each node of the network is not under the physical control of the owner and therefore exposed to attacks originating in this direct access. On should keep in mind that it is not possible to achieve absolute security. Thus, it is crucial to meet the requirements of the use case. The right level of protection allows for the establishment of the business scenario by keeping the costs on an appropriate level but also protecting the interests of the stakeholders.

References

[1] D. Anderson. BOINC: A System for Public-Resource Computing and Storage. In *5th IEEE/ACM International Workshop on Grid Computing*, pages 365–372, 2004.

[2] K. Goldman, R. Perez, and R. Sailer. Linking remote attestation to secure tunnel endpoints. In *Proceedings of the first ACM workshop on Scalable trusted computing*, pages 21–24. ACM New York, NY, USA, 2006.

[3] S. Gürgens, P. Ochsenschläger, and C. Rudolph. On a formal framework for security properties. *Computer Standards & Interfaces*, 27(5):457–466, 2005.

[4] B. Krishnamurthy, C. Wills, and Y. Zhang. On the use and performance of content distribution networks. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 169–182. ACM New York, NY, USA, 2001.

[5] N. Kuntze, D. Mähler, and A. U. Schmidt. Employing trusted computing for the forward pricing of pseudonyms in reputation systems. In *Axmedis 2006, Proceedings of the 2nd International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution, Volume for Workshops, Industrial, and Application Sessions*, 2006.

[6] N. Kuntze and A. U. Schmidt. Protection of dvb systems by trusted computing. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, 2007.

[7] N. Kuntze and A. U. Schmidt. *Handbook of Research on Secure Multimedia Distribution*, chapter Trust in the Calue-Creation Chain of Multimedia Goods. IGI Global, 2009.

[8] T. Mitchem, R. Lu, and R. OBrien. Using Kernel Hypervisors to Secure Applications. In *Proceedings of the Annual Computer Security Applications Conference*, pages 175–181, 1997.

[9] S. Pearson. Trusted Computing Platforms, the Next Security Solution. *Bristol UK: HP Laboratories*, 2002.

[10] Trusted Computing Group. TPM Specification Version 1.2 Revision 103. *Trusted Computing Group*, 2009.