Blindagem de uma Grade Computacional utilizando TPM e *Sandbox*

Luiz Fernando Rust da Costa Carmo, Roberto Paes Nemirovsky

Núcleo de Computação Eletrônica – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 23.24 – 20.010-974 – Rio de Janeiro – RJ – Brasil
rust@nce.ufrj.br, robertopaes@posgrad.nce.ufrj.br

Abstract*. This work aims to exploit all the power emerging from the possible use of Trusted Platform Module (TPM), as well as from virtualization techniques (Sandbox), to provide a safe environment for both grid nodes and users' codes. In this way, this paper presents an architectural proposal for Grids based on a join use of TPM and Sandbox, intending to provide a logically sealed environment for the processing of sensitive information.*

Resumo. *Este trabalho busca explorar todo o poder do Trusted Platform Module (TPM), bem como das técnicas de virtualização para fornecer um ambiente seguro tanto para os elementos de computação da grade (nós) quanto para os códigos de usuários a serem executados. Para isso é apresentada um proposta arquitetural para a concepção de uma grade integrando estes dois elementos (TPM e Sandbox), de forma a criar um ambiente logicamente blindado para o tratamento de informações sensíveis.*

1. Introdução

Dois grandes problemas da computação em grade são a proteção dos nós que a compõem contra códigos maliciosos submetidos para execução; e a proteção de códigos submetidos para a grade contra nós comprometidos. O primeiro problema pode ser resolvido através de técnicas de virtualização, onde um ambiente logicamente isolado do sistema principal é criado para abrigar o sistema de gerência da grade e a execução dos trabalhos submetidos para a mesma. Com isto, mesmo que o ambiente virtual da grade seja comprometido, não haverá risco de o sistema principal ser afetado. Já o segundo problema é extremamente complexo, pois, em última instância, o sistema operacional principal possui acesso irrestrito a todos os processos executados sobre ele, bem como a todo *hardware* presente.

O objetivo desde trabalho é investigar, explorar e sugerir formas de evitar que o nó de uma grade computacional seja comprometido, através da utilização do TPM e técnicas de virtualização, e, caso isso ocorra, impedir que o trabalho sendo executado na grade seja afetado ou capturado.

2. Trabalhos Relacionados

Uma grade computacional pode ser composta por algumas unidades de máquinas trabalhando cooperativamente, ou milhares destas. Da mesma forma, estas máquinas podem estar fisicamente localizadas dentro de uma sala altamente protegida ou

espalhadas por todo o mundo, em forma de máquinas de usuários expostas a vírus, *worms*, cavalos-de-tróia, *backdoors* e demais ameaças.

O projeto Daonity [Mao, Yan e Chen 2006] utiliza primitivas de criptografia executadas em *hardware* do *Trusted Platform Module* (TPM) para proteger o ambiente da grade computacional.

Analisando os problemas de confidencialidade e integridade existentes na computação em grade, [Mao, Jin e Martin 2006] identifica e discute diversas inovações que a tecnologia de *Trusted Computing* oferece para aumentar a segurança em grades.

[Berger et al 2006] descreve uma proposta de virtualização do TPM, ou seja, disponibilizar um TPM virtual para máquinas virtuais mantendo todos os requisitos de *Trust* requeridos pelo TCG, ou seja, *Measurement*, *Attestation* e *Sealing*.

3. Arquitetura

A arquitetura proposta neste trabalho visa fornecer um ambiente confiável para execução de códigos em uma grade computacional, protegendo os dados sensíveis contidos na grade em caso de comprometimento do nó onde o código é executado, e também a situação oposta, protegendo o nó de possíveis códigos maliciosos enviados para a grade.

3.1. Módulos funcionais

O modelo de arquitetura pode ser decomposto em módulos com funções específicas para garantir os requisitos desejados, como mostrado a seguir:

- Gerente da grade - responsável pelo gerenciamento dos recursos dos nós que compõem a grade, bem como pela distribuição dos código a serem executados;
- Gerente do nó - recebe requisições do Gerente da grade, e cria Sandbox a ser utilizada na computação;
- Cliente da grade - recebe o código do Gerente da grade e realiza a computação.

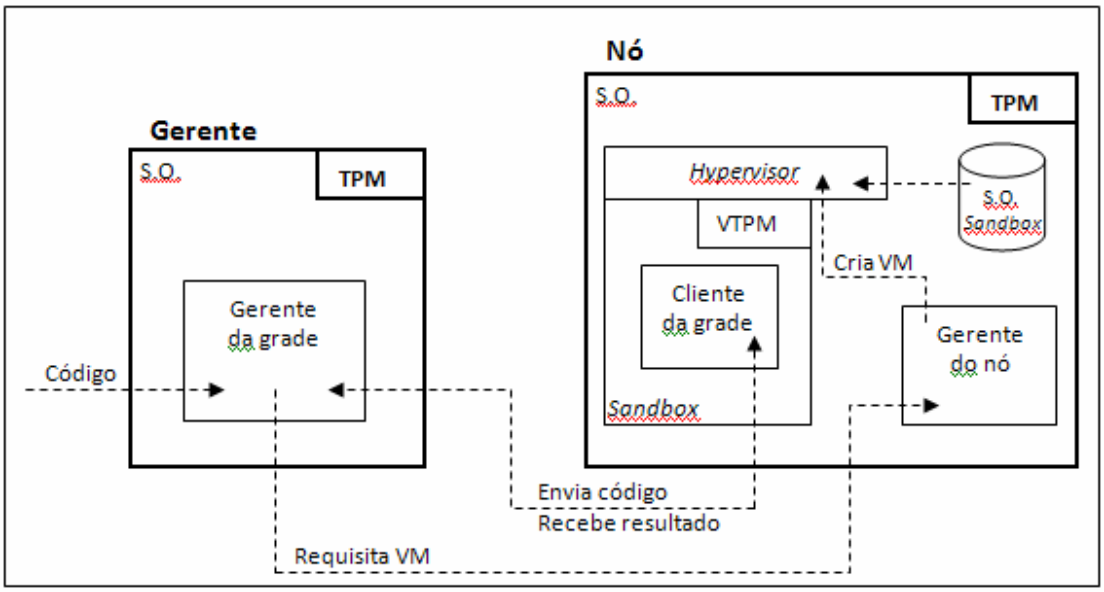

Figura 1: Arquitetura geral do sistema

3.2. Funcionamento

Ao receber uma requisição de um cliente para execução de código na grade, o Gerente da grade elege um nó para tal, de acordo com seus critérios. Neste momento, o Gerente da grade conecta-se ao nó selecionado, onde ocorre a primeira verificação de segurança: tanto o Gerente da grade quanto o Gerente do nó verificam mutuamente a autenticidade da outra ponta, através da análise dos certificados digitais de ambos, ou seja, verifica-se que não há um atacante fazendo-se passar por um elemento da grade.

Após esta etapa, o Gerente da grade requisita uma *Sandbox* para a execução do código enviado pelo cliente. Este ambiente é criado a partir de um arquivo, representando a imagem de um Sistema de arquivos, assinado digitalmente pelo Gerente da grade, logo, caso o mesmo seja modificado, o Gerente do nó irá detectar a alteração.

Para que haja uma verificação efetiva da integridade inicial da *Sandbox*, é necessário que um ambiente novo seja sempre criado e inicializado para cada código a ser executado. Somente desta forma o TPM poderá realizar a atestação de toda a cadeia de *boot* do ambiente virtual, verificando a integridade do BIOS, *Boot loader*, *kernel*, módulos, etc. Assim, tem-se a certeza que todos os componentes do ambiente virtual foram carregados e funcionam conforme esperado pelo Gerente da grade. A este processo de verificação do TPM dá-se o nome de *Attestation* [TCG 2006].

Caso haja uma tentativa de se criar uma *Sandbox* a partir de um arquivo que não o determinado pelo Gerente da grade, a *Sandbox*, através do mecanismo de TPM Virtual apresentado em [Berger et al 2006], irá detectá-la e o processo será interrompido. Mesmo que se crie um ambiente virtual paralelo, de forma que não haja bloqueio do processo de *boot* em função de alterações na *Sandbox*, no momento em que o Cliente da grade tentar se comunicar com o Gerente da grade, este irá detectar que a cadeia de atestação da *Sandbox* não corresponde ao esperado, a um estado íntegro, pelo fato desta comunicação ser validada remotamente pelo TPM do Gerente da grade. A este processo de verificação remota dá-se o nome de *Remote Attestation* [TCG 2006].

Um importante ponto na arquitetura proposta é a separação em dois módulos, chamados Gerente do nó e Cliente da grade, dentro de um determinado nó. Esta separação se justifica pelo fato do Gerente do nó não precisar se envolver efetivamente na computação da grade, sendo assim, seu comprometimento não afeta o processamento da tarefa. O Cliente da grade, por sua vez, tem seu código protegido pelo TPM Virtual.

O código a ser executado deve levar em conta a existência de um TPM Virtual e se utilizar das funcionalidades, protegendo toda e qualquer informação sensível. Esta proteção deve se utilizar da funcionalidade *Sealing* [TCG 2006] do TPM, ou seja, a chave criptográfica deve ser atrelada ao conjunto de PCR [TCG 2006] do ambiente virtual. Desta forma, somente um código sendo executado dentro da *Sandbox* poderá realizar a decriptografia dos dados. Com isto, o ambiente virtual se torna blindado, pois nenhum código executado fora da *Sandbox* conseguirá acesso aos dados protegidos.

4. Avaliação da proposta arquitetural

O uso de Sandbox em conjunto com TPM Virtual, ou seja, proporcionar a cada ambiente virtual todo o poder do TPM de forma exclusiva, com todas as garantias de *Trust* requeridas pelo TCG, propicia à grade um ambiente altamente confiável e controlado. Desta forma, independente do estado em que se encontra um determinado

nó, pode-se ter certeza que a *Sandbox* a ser utilizada não se encontra comprometida. Além disto, a operação de *sealing* do TPM, quando utilizada dentro da *Sandbox*, vincula a criptografia de dados sensíveis da computação ao conjunto de PCR do ambiente virtual, tornando impossível a um código externo realizar a descriptografia destes dados, criando então um ambiente logicamente blindado para execução segura de códigos.

A virtualização protege o sistema operacional do nó. Isto ocorre devido ao controle da execução do código inserido no ambiente virtual, por técnicas de gerência de memória. Todo acesso ao *hardware* também é controlado pelo gerenciador do ambiente virtual. Além de proteger o sistema operacional principal, outra característica importante é o isolamento entre duas *Sandbox* executando no mesmo sistema.

5. Conclusões e Trabalhos Futuros

A combinação de TPM com técnicas de virtualização foi materializada através da proposta de utilização de técnicas de virtualização do TPM, disponibilizando todas as funcionalidades do TPM em um ambiente virtual, sem perder a cadeia de *Trust*.

Uma contribuição importante da arquitetura proposta é a necessidade de que um ambiente novo seja sempre criado e inicializado para cada *job* a ser executado, de forma que haja uma verificação efetiva da integridade inicial da *Sandbox*. Somente desta forma o TPM é capaz de realizar a atestação de toda a cadeia de *boot* do ambiente virtual, verificando a integridade do BIOS, *loader*, *kernel*, módulos, etc. Além disto, a operação de *Sealing* do TPM vincula a criptografia de dados sensíveis da computação ao conjunto de PCR do ambiente virtual, criando então um ambiente logicamente blindado.

Por outro lado, o uso da virtualização protege o sistema operacional principal do nó devido ao controle da execução do código inserido no ambiente virtual e ao controle de acesso ao *hardware* em geral. Outra característica importante é o isolamento entre dos *jobs* em execução no mesmo sistema.

Os trabalhos em andamento incluem uma avaliação quantitativa do desempenho desta proposta através de avaliações empíricas e simulações.

6. Referências

[TCG 2006] Trusted Computing Group – TCG, (2006). “Trusted Platform Module specification”, TPM Work Group, dísponível em https://www.trustedcomputinggroup.org/groups/tpm/.

[Mao, Yan e Chen 2006] Mao, W., Yan., F., e Chen, C. (2006). “Daonity: grid security with behaviour conformity from trusted computing”, Anais do First ACM workshop on Scalable trusted computing, pp. 43- 46.

[Mao, Jin e Martin 2006] Mao, W., Jin, H., e Martin, A. (2006). “Innovations for Grid Security from Trusted Computing”, Anais do The Fourteenth Global Grid Forum, USA.

[Berger et al 2006] S. Berger, R. Cáceres, K. Goldman, R. Perez, R. Sailer, e L. van Doorn. (2006). “vTPM: Virtualizing the Trusted Platform Module”, Amais do 15th USENIX Security Symposium.