Mecanismo de Verificação de Integridade de Software Baseado em BIOS UEFI

Marciel de Liz Santos, Michelle S. Wangham, Cesar A. Zeferino

Universidade do Vale do Itajaí (UNIVALI) – Itajaí – SC – Brasil

marciel.dls@gmail.com, {wangham,zeferino}@univali.br

Abstract. *This paper describes the proposal for a verification mechanism that takes advantage of UEFI BIOS resources to attest the integrity of the software of embedded systems used in IoT. This mechanism is composed by an integrity verification application called AVIS UEFI, which is executed in the Pre-Boot Applications phase and uses digital signature and keys stored in cryptographic devices to verify if the software has been tampered with. According to the result of the verification, the system is initialized or shut down. The next steps of this work will be to finalize the implementation of the prototype, present and evaluate the test results to show its applicability in a real scenario.*

1. Introdução

Devido à ampla adoção do paradigma Internet das Coisas (IoT), há uma grande quantidade de dispositivos conectados à Internet e uma previsão de maior crescimento nos próximos anos. Este crescimento na quantidade e diversidade de dispositivos com sistemas embarcados conectados amplia a preocupação com a segurança, uma vez que estes são mais vulneráveis e mais propensos a ataques. Além disso, com o avanço tecnológico da eletrônica digital, os sistemas embarcados vêm aumentando seu poder de processamento e suas funcionalidades ao incorporar novos componentes em sua estrutura interna, como BIOS UEFI. Estes novos recursos conferem flexibilidade e permitem criar aplicações mais poderosas. No entanto, geram novas vulnerabilidades e preocupações com a segurança, principalmente em situações em que pode haver interesse de determinados indivíduos em alterar o software dos dispositivos para obter benefícios [Abomhara e Koien 2015].

Existem diversas abordagens para garantir a integridade de softwares em situações em que entidades homologadoras ou fabricantes não confiam no usuário. Estas abordagens, no entanto, apresentam limitações e não são apropriadas para sistemas embarcados complexos [Liu 2015].

2. Solução Proposta

A proposta consiste em um mecanismo de verificação que aproveita os recursos do BIOS UEFI para atestar a integridade de software de sistemas embarcados utilizados em IoT. Este mecanismo é composto por uma aplicação de verificação de integridade chamada AVIS UEFI, que é executada na fase de aplicativos de Pré-Boot da inicialização e utiliza assinatura digital e chaves armazenadas em dispositivos criptográficos para verificar se o software foi adulterado. De acordo com o resultado da verificação, o sistema é inicializado ou desligado. A Figura 1 ilustra os componentes do

mecanismo e como estes interagem durante o processo de verificação de integridade. Estes componentes são: o software a ser verificado, que se encontra armazenado em memória secundária; a AVIS UEFI, que é a aplicação de verificação de integridade incorporada à área de Aplicativos de Pré Boot do BIOS e executada sob controle do mesmo; e o USB Key, um dispositivo criptográfico externo e seguro para o armazenamento da assinatura e das chaves necessárias para a verificação. A Figura 1 mostra, ainda, as entidades que interagem com a solução proposta: (i) o fabricante, que projeta e fabrica o dispositivo, e instala o software a ser homologado; (ii) o homologador, que é a entidade que deve atestar a integridade do software instalado; e (iii) o usuário, que é a entidade que utiliza o dispositivo e pode ter interesse de modificar o software para obter benefícios ilícitos.

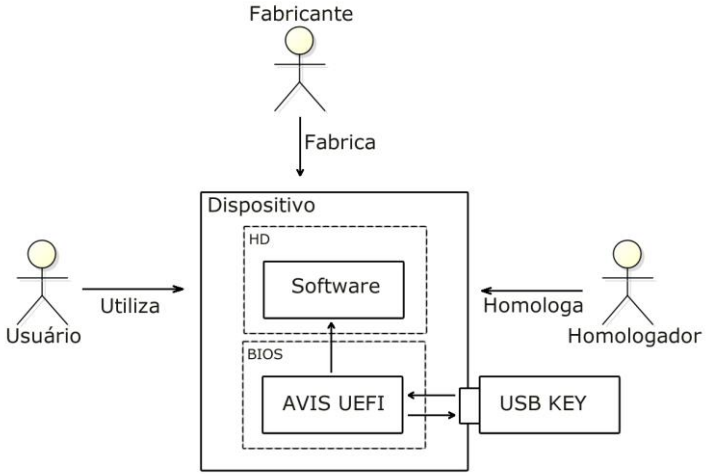

Figura 1. Componentes e entidades do mecanismo proposto

A sequência de funcionamento e a interação entre os componentes do mecanismo são: (1) O sistema embarcado é ligado e começa o processo de inicialização; (2) O módulo de Aplicativos de Pré-Boot do BIOS UEFI executa a AVIS UEFI; (3) A aplicação acessa o arquivo binário do software a ser verificado; (4) A aplicação calcula o *hash* do arquivo acessado; (5) A aplicação autentica o dispositivo criptográfico USB Key; (6) A aplicação lê a assinatura e a chave armazenadas no dispositivo USB Key; (7) A aplicação verifica a integridade do software. Para isso, descriptografa a assinatura com a chave pública para extrair o *hash* e o compara com o *hash* calculado no Passo 4; e (8) Se a verificação é bem sucedida, o equipamento é liberado para uso e o sistema operacional é carregado. Caso contrário, o sistema é bloqueado e impedido de uso.

A AVIS UEFI, aplicação de verificação de integridade, deve ser incorporada à área de aplicativos de Pré-Boot do BIOS UEFI. Esta aplicação será executada na fase de Carregamento de Sistemas de Transição, depois do disparo de boot e antes de carregar e passar o controle para o sistema operacional. Caso a verificação não seja bem sucedida, a inicialização será interrompida e o comando *shutdown* será enviado para impedir o uso do dispositivo.

A assinatura utilizada pela aplicação para verificar o software é gerada no processo de configuração por meio do cálculo do resumo criptográfico do mesmo com o

algoritmo SHA-256. Este, por sua vez, é criptografado com o algoritmo assimétrico RSA e a chave privada da entidade homologadora. No momento da verificação de integridade, a aplicação deve utilizar a chave pública do homologador para extrair o *hash* e compará-lo com o *hash* original.

A assinatura e a chave pública são armazenadas em um USB Key protegido por um mecanismo de autenticação baseado em PIN e que destrói seu conteúdo caso sejam feitas cinco tentativas incorretas. Assim, a aplicação deve conhecer o PIN do dispositivo e implementar os mecanismos necessários para estabelecer uma comunicação segura com o USB Key.

Como prova de conceito da solução proposta, um protótipo está sendo desenvolvido na linguagem de programação C, no ambiente de programação Eclipse 3.8 e com o auxílio do framework UDK versão 2017. Este framework permite implementar aplicações UEFI.

Os testes de integração para verificar o correto funcionamento do protótipo estão sendo executados em um ambiente virtual baseado no emulador QEMU fornecido pelo *framework* UDK que disponibiliza uma máquina virtual com BIOS UEFI para a realização de testes. Já a aplicabilidade e a utilidade do mecanismo em um cenário real serão demonstradas por meio de um estudo de caso, que consiste em aplicar o protótipo desenvolvido no projeto Cronotacógrafo do INMETRO para automatizar o processo de verificação do software homologado dos simuladores de pista fabricados pela empresa Moss do Brasil com a finalidade de evitar fraudes e reduzir os custos com verificações periódicas.

3. Considerações Finais

A principal contribuição deste trabalho é fornecer um mecanismo simples e eficaz que verifica a integridade do software em cada inicialização e impede seu uso em caso de adulterações, o que aumenta a segurança de sistemas embarcados e promove a redução de custos nos casos em que há a necessidade de homologação por terceiros ou entidades reguladoras. Os próximos passos desta pesquisa será apresentar e avaliar os resultados obtidos dos testes para mostrar a sua aplicabilidade em um cenário real.

Referências

Abomhara, Mohamed; Koien, Geir M. (2015) Cyber “Security and the Internet of Things: Vulnerabilities, Threats, Intruders and Attacks”. Journal of Cyber Security, River Publishers, v. 4, p. 65-88.

Liu, Hong; LI, Hongmin; Vasserman, Eugene Y. (2015) “Practicality of Using Side-Channel Analysis for Software Integrity Checking of Embedded Systems”. In: 11th EAI International Conference on Security and Privacy in Communication Networks, SecureComm 2015, Dallas, TX, USA, October 26-29.