República Federativa do Brasil
Ministério do Desenvolvimento, Indústria e do Comércio Exterior
Instituto Nacional da Propriedade Industrial

(21) PI 0809281-8 A2

(22) Data de Depósito: 07/03/2008
(43) Data da Publicação: 02/09/2014
(RPI 2278)

(51) Int.Cl.:
G06F 9/06
G06F 9/22

(54) Título: "SISTEMA DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL E MÉTODO DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL"

(57) Resumo:

(30) Prioridade Unionista: 13/04/2007 US 11/786,874

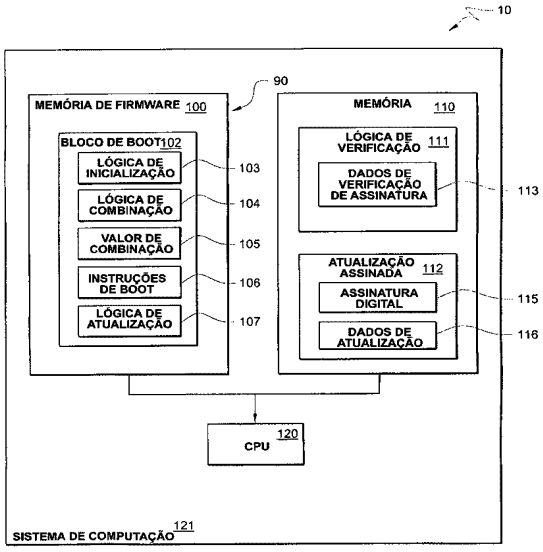
(73) Titular(es): HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.

(72) Inventor(es): BORIS BALACHEFF, LAN WANG , VALIUDDIN Y. ALI

(74) Procurador(es): Antonio Mauricio Pedras Arnaud

(86) Pedido Internacional: PCT US2008003188 de 07/03/2008

(87) Publicação Internacional: WO 2008/127523de 23/10/2008



"SISTEMA DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL E MÉTODO DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL".

Antecedentes

Os fabricantes de computadores geralmente necessitam um modo para atualizar os conteúdos de um componente confiável, tal como uma memória flash de sistema básico de entrada/saída (BIOS), para reparar bugs e/ou fornecer novas capacidades. Entretanto, permitir a memória flash do BIOS ou outro componente confiável ser modificado torna o componente confiável suscetível à corrupção por lógica maliciosa ou não autorizada.

Descrição resumida dos desenhos

Para uma compreensão mais completa do presente pedido de patente, dos objetivos e vantagens da mesma, referência é feita agora às descrições seguintes tomadas em conjunção com os desenhos anexos, nos quais:

A figura 1 é um diagrama ilustrando uma configuração de um sistema de atualização de componente confiável; e

A figura 2 é um diagrama ilustrando uma configuração de um método de atualização de componente confiável.

Descrição detalhada dos desenhos

A figura 1 é um diagrama ilustrando uma configuração de um sistema de atualização de componente confiável 10. O sistema de atualização 10 permite um componente confiável verificar que uma atualização proposta para o componente confiável é fornecida por uma fonte que tenha sido identificada previamente como uma fonte confiável e que a atualização proposta do componente confiável não tenha sido adulterada. Na configuração ilustrada na figura 1, o sistema de atualização 10 compreende um sistema de computação 121 tendo um componente confiável 90, uma unidade de processamento central (CPU) 120 e memória de sistema 110. Na figura 1, o componente confiável 90 compreende uma memória de firmware [programa residente de inicialização] 100 tal como uma memória flash de firmware. O sistema de computação 121 pode compreender um computador notebook, um computador desktop, um servidor,

um dispositivo para jogos, um dispositivo de áudio, um dispositivo de vídeo, ou qualquer tipo de dispositivo compreendendo um componente confiável. Na figura 1, a memória de firmware 100 provê funções de boot para o sistema de computação 121 quando o sistema de computação 121 é pela primeira vez energizado, reiniciado e/ou religado. Por exemplo, a memória de firmware 100 pode inicializar a configuração da CPU 120 antes de um instante quando a CPU 120 começa a executar instruções, tais como permitindo e/ou desabilitando certas capacidades da CPU 120 e definindo uma taxa de ciclo ["clock"] na CPU 120.

Na configuração ilustrada na figura 1, o sistema de computação 121 compreende duas seções de memória que são separadas por "confiança" e geralmente residem em componentes fabricados separadamente. Por exemplo, em algumas configurações, a memória de firmware 100 é uma memória flash compreendendo um bloco de boot de memória confiável 102 (p.ex., para prover funções de boot para o sistema de computação 121 quando o sistema de computação 121 é pela primeira vez energizado, reiniciado e/ou religado), enquanto a memória 110 pode não compreender memória confiável e/ou pode compreender uma memória não de firmware. Entretanto, deve ser entendido que uma memória confiável também pode residir em um local outro que a memória de firmware 100. Como usado aqui, "confiança" ou "confiável" significa a expectativa de operação consistente dentro de um conjunto pré-definido de regras que são impostas por hardware e/ou software de computação, tal como a definição de "confiança" como registrada na TCG Specification Architecture Overview Specification [Especificação de visão geral de arquitetura da especificação], Revisão 1.2 (Trusted Computing Group [Grupo de Computação Confiável], 2004). Por exemplo, garantir que os conteúdos de uma certa seção de memória no sistema de computação 121, tal como o bloco de boot 102 de memória firmware 100, contenha somente

informações produzidas por uma fonte identificada anteriormente, definida como uma fonte confiável, habilita a confiança daquela certa seção de componente do sistema.

A memória de firmware 100 é acoplada à CPU 120 para executar operações de boot e a memória 110 para ler dados usados pela memória de firmware 100. Na configuração ilustrada na figura 1, a memória de firmware 100 compreende memória flash, não volátil. Em algumas configurações, a memória de firmware 100 compreende um sistema básico de entrada/saída (BIOS). Em algumas configurações, a memória de firmware 100 compreende uma Interface de Firmware Extensível (EFI) ou uma EFI Uniforme (UEFI). Entretanto, deve ser entendido que a memória de firmware 100 pode compreender qualquer sistema provendo funcionalidade de boot para um sistema de computação. A memória 110 pode compreender memória volátil, memória não volátil e armazenagem permanente, tal como um acionador de mídia digital (DMD). Na configuração ilustrada na figura 1, o sistema de computação 121 é mostrado como compreendendo uma única CPU 120, embora deva ser entendido que uma quantidade maior de CPUs podem ser usadas.

Na configuração ilustrada na figura 1, a memória de firmware 100 compreende o bloco de boot 102. O bloco de boot 102 é geralmente a lógica inicial executada na memória de firmware 100 quando o sistema de computação 121 é primeiro energizado, reiniciado e/ou religado. O bloco de boot 102 é uma lógica confiável porque o bloco de boot 102 é travado dentro da memória de firmware 100 e é protegida de atualização durante operação normal do sistema de computação 121 (isto é, o bloco de boot 102 é atualizado por métodos confiáveis a partir de uma fonte confiável que pode ser validada, por exemplo, por métodos criptográficos).

Sistemas confiáveis e métodos confiáveis são sistemas e métodos tendo um nível suficiente de segurança para

impedir mudanças não autorizadas para os conteúdos da memória confiável. Por exemplo, os sistemas confiáveis e os métodos confiáveis podem usar métodos fortes de segurança, tais como algoritmos criptográficos incluindo RSA Verify, Criptografia de Curva Elíptica (ECC), Algoritmo de Assinatura Digital (DAS), o Algoritmo de Combinação Segura 1 (SHA-1) e SHA-2 para garantir que uma fonte confiável produziu um certo arquivo digital e/ou que o arquivo digital não foi alterado ou adulterado desde que a fonte confiável o produziu. Sistemas confiáveis e métodos confiáveis garantem que o arquivo digital pode ser confiável antes de usar o arquivo digital para modificar a memória confiável. Desta maneira, lógica que não é autorizada pela fonte confiável, bem como vírus, podem ser mentidos fora da memória confiável, apesar da memória confiável ser modificável.

Na configuração ilustrada na figura 1, o bloco de boot 102 compreende lógica de inicialização 103, lógica de combinação 104, um valor de combinação 105, instruções de boot 106 e lógica de atualização 114. A lógica de inicialização 103 compreende lógica, dados e instruções que são processadas e/ou executadas dentro do bloco de boot 102 da memória de firmware 100. Geralmente, quando a memória de firmware 100 completa o processamento das instruções encontradas nela e/ou referenciadas pelo bloco de boot 102, a memória de firmware 100 pode executar quaisquer outros procedimentos de boot antes de transferir o controle do sistema de computação 121 para um sistema operacional. A lógica de atualização 114 compreende instruções executáveis para sobrescrever seções da memória de firmware 100, tal como o bloco de boot 102.

A lógica de combinação 104 é executada para verificar a integridade de um arquivo digital executando uma função de combinação, a qual é uma operação matemática produzindo um valor de combinação como um resultado. Em

algumas configurações, a lógica de combinação 104 compreende o algoritmo SHA-1, embora outras funções de combinação possam ser usadas alternativamente ou em adição. O valor de combinação 105 é um valor de validação de integridade computado, o qual é um número usado para validar que um arquivo digital (p.ex., um programa executável de computador) não foi adulterado. Em operação, a lógica de combinação 104 é executada em um arquivo digital para calcular um valor de combinação que é comparado com um valor de combinação 105 armazenado na memória de firmware 100. Assim, se o valor de combinação calculado corresponder a e/ou de outra forma combinar com o valor de combinação 105, o arquivo digital é considerado confiável. A lógica de combinação 104 e o valor de combinação 105 são armazenados no bloco de boot 102, o qual reside na memória de firmware 100. Em algumas configurações, a lógica de combinação 104 e o valor de combinação 105 são colocados inicialmente no bloco de boot 102 quando o sistema de computação 121 é fabricado, reformado, atualizado ou reparado. Assim, o sistema de atualização 10 é capaz de estender confiança do bloco de boot 102 para arquivos digitais armazenados fora da memória de firmware 100, criando assim uma cadeia de confiança.

Na configuração ilustrada na figura 1, a memória 110 compreende lógica de verificação 111 e uma atualização assinada 112. A atualização assinada 112 compreende uma atualização para o componente confiável 90, tal como uma atualização para o bloco de boot 102, proposta por um parceiro confiável, tal como o fabricante do sistema de computação 121. Por exemplo, a atualização assinada 112 pode compreender um reparo de bug representado em dados de atualização 116. A lógica de verificação 111 compreende lógica, instruções e dados, incluindo dados de verificação de assinatura 113, para verificar que a atualização assinada 112 foi produzida por uma fonte confiável, tal como o fabricante do sistema de computação

121 ou um administrador de rede de computadores, e adicionalmente que a atualização assinada 112 não foi adulterada desde quando foi produzida. Cada uma de a lógica de combinação 104 e a lógica de verificação 111 compreende meios para verificação de confiança. Entretanto, a lógica de verificação 111 pode ser confiável, apesar de armazenada fora da memória de firmware confiável 100, porque a lógica de combinação 104 permite a memória de firmware 100 estender confiança da memória de firmware 100 para a lógica de verificação 111. Os dados de verificação de assinatura 113 são usados para verificação de integridade (p.ex., garantir nenhuma adulteração) e verificação de origem (p.ex., garantir que a fonte confiável identificada anteriormente é o fabricante) permitindo a lógica de verificação 111 calcular independentemente certas porções de uma assinatura digital 115 na atualização assinada 112. Na configuração ilustrada na figura 1, a lógica de verificação 111 é disposta em e/ou pode ser executada a partir de a memória de sistema 110. Entretanto, deve ser entendido que em algumas configurações, a lógica de verificação 110 pode residir em e ser executada a partir de a memória de firmware 100.

Em algumas configurações, a assinatura digital 115 compreende uma sequência alfanumérica ligada à atualização assinada 112 pelo autor da atualização assinada 112, a qual identifica unicamente o autor e também fornece uma base para garantir a integridade de arquivo com um algoritmo de combinação. Os dados de verificação de assinatura 113 são dados usados pela lógica de verificação 111 para executar cálculos independentes usando a assinatura digital 115 e outras porções da atualização assinada 112. Por exemplo, em algumas configurações, os dados de verificação de assinatura 113 compreendem uma combinação de dados de atualização 116 assinados e/ou criptografados por uma chave privada do parceiro confiável. A lógica de

verificação 111 combina pelo menos uma porção da atualização assinada 112, por exemplo os dados de atualização 116, descriptografa a assinatura digital 115, e determina se a combinação de dados de atualização 116 corresponde à assinatura digital descriptografada 115. Se os dois valores corresponderem, a atualização assinada 112 é considerada a ser uma réplica verificada de um arquivo autorizado pelo parceiro confiável que não foi adulterada desde quando foi produzida. Em algumas configurações, a lógica de verificação 111 compreende RSA Verify, que é um algoritmo adequado para verificação de assinatura digital.

Os dados de atualização 116 compreendem informações a serem usadas para modificar os conteúdos da memória de firmware 100, tal como os conteúdos do bloco de boot 102, lógica de combinação 104, valor de combinação 105 e instruções de boot 106. Em algumas configurações, a atualização assinada 112 pode adicionalmente ou alternativamente compreender informações de modificação para a lógica de verificação 111. Em algumas configurações, a atualização assinada 112 compreende um programa de remendo executável capaz de modificar os conteúdos do bloco de boot 102 sem usar a lógica de atualização 114.

Em algumas configurações de operação do sistema de atualização 10, um fabricante do sistema de computação 121 ou outra entidade gera lógica de verificação 111 com dados de verificação de assinatura 113 que identificam unicamente a entidade como uma fonte confiável para atualizações futuras para a memória de firmware 100. A entidade combina toda ou uma porção da lógica de verificação 111 com uma duplicata da lógica de combinação 104 para calcular o valor de combinação 105. A lógica de combinação 104, valor de combinação 105 e instruções de boot 106 são armazenadas no bloco de boot 102 dentro da memória de firmware 100, e a lógica de verificação 111 é armazenada na memória 110. Em algumas configurações, cada

vez que o sistema de computação 121 passa por boot, as instruções de boot procuram por uma indicação que uma atualização, tal como a atualização assinada 112, está disponível e/ou esperando (p.ex., esperando na memória 110, num disco rígido ou outro meio de armazenagem) para ser usada para modificar os conteúdos da memória de firmware 100. Por exemplo, durante um uso anterior do sistema de computação 121, a entidade pode ter enviado uma mensagem eletrônica através de uma rede de computadores colocando a atualização assinada 112 na memória 110.

Se a atualização assinada 112 é encontrada, a lógica de combinação 104 valida a integridade da lógica de verificação 111 combinando toda ou uma porção da lógica de verificação 111 e compara o resultado da combinação com o valor de combinação 105 para garantir que a lógica de verificação 111 não tenha sido adulterada. Se a lógica de verificação 111 for válida (isto é, for considerada a não ter sido alterada baseado no valor de combinação da lógica de verificação 111 corresponder ao valor de combinação 105), o bloco de boot 102 faz a execução da lógica de verificação 111 verificar a autoria da assinatura digital 115 usando os dados de verificação de assinatura 113. Por exemplo, em algumas configurações, a lógica de verificação 111 é usada para verificar o arquivo de assinatura digital (p.ex., assinatura digital 115) associado com a atualização assinada 112 usando um método criptográfico tal como o algoritmo de verificação de assinatura digital RSA. Assim, em algumas configurações, a assinatura digital 115 representa um valor de combinação pré-calculado de pelo menos uma porção dos dados de atualização 116 que foram criptografados por uma chave privada de um parceiro confiável. Em operação, a lógica de verificação 111 descriptografa a assinatura digital 115 (p.ex., usando dados de verificação de assinatura 113, tal como uma chave pública emparelhada com a chave privada do parceiro

confiável), combina uma porção correspondente dos dados de atualização 116, e compara o valor de combinação com a assinatura digital descriptografada 115. Se a verificação da assinatura digital 115 tiver sucesso, o bloco de boot 102 executa a lógica de atualização 114 para modificar os conteúdos da memória de firmware 100, tal como o bloco de boot 102, usando os dados de atualização 116. Assim, configurações do sistema 10 permitem um fabricante ou outra entidade confiável reparar bugs e/ou de outra forma modificar memória de firmware 100 enquanto minimizando o risco de mudanças não autorizadas ou maliciosas para a memória de firmware 100 por uma entidade não autorizada.

A figura 2 é um diagrama ilustrando uma configuração de um método de atualização confiável de memória de firmware 200. O método 200 é descrito com referência ao sistema de atualização 10 da figura 1, embora deva ser entendido que o método 200 pode ser usado com configurações alternativas.

No bloco 201, o bloco de boot 102 determina se uma atualização assinada 112 está presente (p.ex., presente na memória 110 ou em algum outro lugar). Se nenhuma atualização assinada 112 estiver presente, o método prossegue para o bloco 222, onde o bloco de boot 102 continua a fazer o boot do sistema de computação 121. Se a atualização assinada 112 estiver disponível ou presente, bloco de boot 102, a lógica de combinação 104 combina a lógica de verificação 111 para produzir um valor de combinação recém calculado no bloco 204. No bloco 206, o valor de combinação recém calculado é comparado com o valor de combinação 105 armazenado na memória de firmware 100 para determinar a validade da lógica de verificação 111. Se, no bloco de decisão 208, o valor de combinação recém calculado não corresponder ao valor de combinação 105, o componente confiável 90 não será atualizado com tal atualização, e o bloco de boot 102 ao invés continua a fazer o boot do sistema de computação 121 no bloco 222. Se, no bloco de decisão 208,

o valor de combinação recém calculado corresponder ao valor de combinação 105, o método prossegue para o bloco 212.

No bloco 212, a lógica de verificação 111 é executada e usada para verificar a assinatura digital da atualização assinada 112 combinando os dados de atualização 116. No bloco 214, a lógica de verificação 111 descriptografa a assinatura digital 115 usando os dados de verificação de assinatura 113 e compara a combinação dos dados de atualização 116 com a assinatura digital descriptografada 115 no bloco 216. Se, no bloco de decisão 218, a combinação dos dados de atualização 116 não corresponder à assinatura digital descriptografada 115, a assinatura digital da atualização assinada 112 não é verificada, e tal atualização não será carregada, instalada ou implementada, mas pelo contrário o bloco de boot 102 continua a fazer o boot do sistema de computação 121 no bloco 222. Se, no bloco de decisão 218, a combinação dos dados de atualização 116 combinar com a assinatura digital descriptografada 115, a assinatura digital da atualização assinada 112 é considerada verificada válida, e o componente confiável 90 é atualizado com os dados de atualização 116 executando a lógica de atualização 114 no bloco 220. O bloco de boot 102 continua a fazer o boot do sistema de computação 121 no bloco 222.

Portanto, configurações do sistema de atualização 10 permitem atualizações de campo serem feitas para a memória confiável. Deve ser entendido que no método descrito, certas funções podem ser omitidas, realizadas em uma sequência diferente daquela representada na figura 2, ou executadas simultaneamente. Também, deve ser entendido que o método representado na figura 2 pode ser alterado para abranger qualquer das outras características ou aspectos como descritos em qualquer outro lugar na especificação. Adicionalmente, configurações podem ser implementadas em software e podem ser adaptadas para operar em diferentes plataformas e

sistemas operacionais. Em particular, as funções implementadas pela lógica 104, lógica 111, e lógica 114, por exemplo, podem ser fornecidas como uma listagem ordenada de instruções executáveis que podem ser configuradas em qualquer meio lido por computador para uso por ou em conexão com um sistema, aparelho, ou dispositivo de execução de instruções, tal como um sistema baseado em computador, sistema contendo processador, ou outro sistema que possa ir buscar as instruções do sistema, aparelho, ou dispositivo de execução de instruções, e executar as instruções. No contexto deste documento, um "meio lido por computador" pode ser qualquer meio que possa conter, armazenar, comunicar, propagar ou transportar o programa para uso por ou em conexão com o sistema, aparelho ou dispositivo de execução de instruções. O meio lido por computador pode ser, por exemplo, mas não limitado a, um sistema, aparelho, dispositivo eletrônico, magnético, ótico, eletromagnético, infravermelho, ou semicondutor, ou meio e propagação.

REIVINDICAÇÕES

1. Sistema de atualização de componente confiável, caracterizado pelo fato de compreender:

lógica de verificação (111) configurada para validar a integridade de uma atualização para um componente confiável (90) de um dispositivo de computação (121); e

lógica (104) disposta no componente confiável (90) e configurada para validar a integridade da lógica de verificação (111).

2. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de a lógica (104) no componente confiável (90) ser configurada para combinar pelo menos uma porção da lógica de verificação (111) para validar a integridade da lógica de verificação (111).

3. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de a lógica (104) no componente confiável (90) ser configurada para combinar pelo menos uma porção da lógica de verificação (111) e comparar um resultado da combinação com um valor de combinação pré-determinado (115) armazenado no componente confiável (90).

4. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de a lógica de verificação (111) ser configurada para validar a integridade de uma assinatura (115) associada com a atualização.

5. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de a lógica de verificação (111) ser configurada para combinar pelo menos uma porção da atualização para validar a integridade da atualização.

6. Método de atualização de componente confiável, caracterizado pelo fato de compreender:

validar a integridade de uma atualização de um componente confiável (90) de um dispositivo de computação (121) usando lógica de verificação (111); e

validar, usando a lógica (104) disposta no componente confiável (90), a integridade da lógica de verificação (111).

7. Método, de acordo com a reivindicação 6, caracterizado pelo fato de a validação da integridade da atualização compreender validar a integridade de uma assinatura (115) associada com a atualização.

8. Método, de acordo com a reivindicação 6, caracterizado pelo fato de a validação da integridade da lógica de verificação (111) compreender combinar pelo menos uma porção da lógica de verificação (111).

9. Método, de acordo com a reivindicação 6, caracterizado pelo fato de a validação da integridade da lógica de verificação (111) compreender combinar pelo menos uma porção da lógica de verificação (111) e comparar um resultado da combinação com um valor de combinação pré-determinado (105) armazenado no componente confiável (90).

10. Método, de acordo com a reivindicação 6, caracterizado pelo fato de a validação da integridade da atualização compreender combinar pelo menos uma porção da atualização.

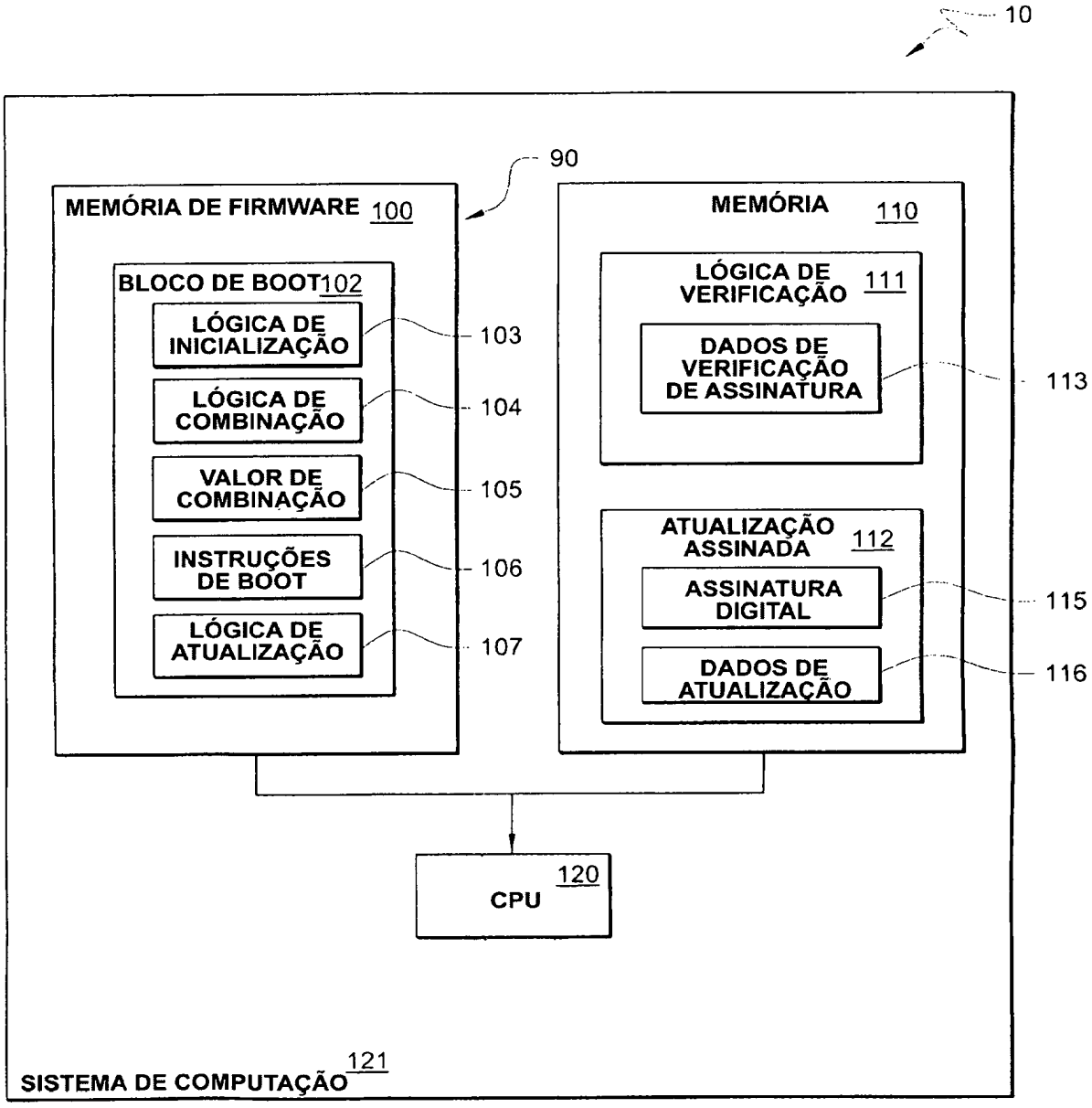

10
90
MEMÓRIA DE FIRMWARE 100
BLOCO DE BOOT 102
LÓGICA DE INICIALIZAÇÃO
LÓGICA DE COMBINAÇÃO
VALOR DE COMBINAÇÃO
INSTRUÇÕES DE BOOT
LÓGICA DE ATUALIZAÇÃO
103
104
105
106
107
MEMÓRIA 110
LÓGICA DE VERIFICAÇÃO 111
DADOS DE VERIFICAÇÃO DE ASSINATURA
113
ATUALIZAÇÃO ASSINADA 112
ASSINATURA DIGITAL
DADOS DE ATUALIZAÇÃO
115
116
CPU 120
SISTEMA DE COMPUTAÇÃO 121

FIG.1

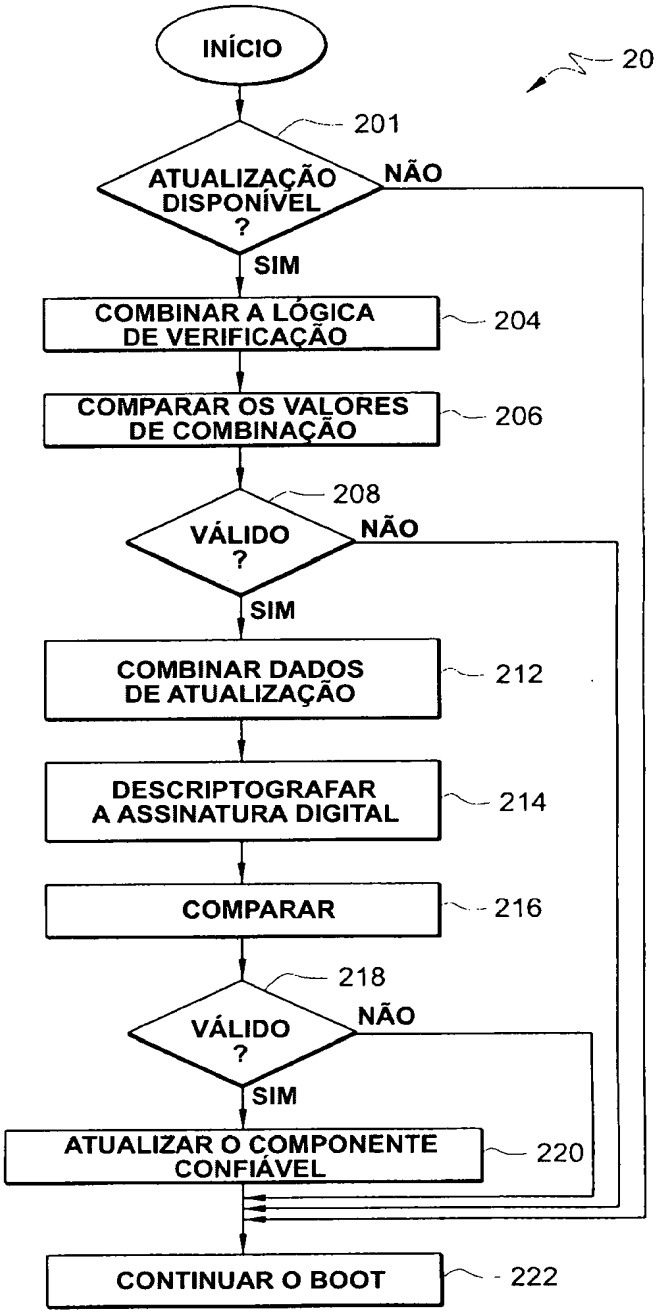

INÍCIO
20
201
ATUALIZAÇÃO DISPONÍVEL ?
NÃO
SIM
COMBINAR A LÓGICA DE VERIFICAÇÃO
204
COMPARAR OS VALORES DE COMBINAÇÃO
206
208
VÁLIDO ?
NÃO
SIM
COMBINAR DADOS DE ATUALIZAÇÃO
212
DESCRIPTOGRAFAR A ASSINATURA DIGITAL
214
COMPARAR
216
218
VÁLIDO ?
NÃO
SIM
ATUALIZAR O COMPONENTE CONFIÁVEL
220
CONTINUAR O BOOT
222

FIG.2

RESUMO

"SISTEMA DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL E MÉTODO DE ATUALIZAÇÃO DE COMPONENTE CONFIÁVEL".

Um sistema de atualização de componente confiável (10) compreende lógica de verificação (111) configurada para validar a integridade de uma atualização para um componente confiável (90) de um dispositivo de computação (121), e lógica (104) disposta no componente confiável (90) e configurada para validar a integridade da lógica de verificação (111).