Respostas às vulnerabilidades e sugestões de melhorias encontradas no Teste Público de Segurança 2017

Relatório técnico

Brasília, 23 de maio de 2018

Tribunal Superior Eleitoral
Secretaria de Tecnologia da Informação
Coordenadoria de Sistemas Eleitorais
Seção de Voto Informatizado

Introdução

O Teste Público de Segurança - TPS, iniciado em 2009 e já com quatro edições, é um dos marcos do processo de desenvolvimento dos sistemas eleitorais e do hardware da urna eletrônica. Ao longo dos últimos anos, a cada edição do TPS foi possível aprimorar os sistemas eleitorais que são utilizados nas eleições subsequentes, as quais passaram a contar com hardware e software mais seguros e robustos.

A edição de 2017 do TPS contou com um grande número de pesquisadores e profissionais altamente qualificados. E não por acaso, a excelência do seu trabalho contribuiu para a descoberta do maior número de achados de software para uma única edição do TPS.

Pela primeira vez, após a realização do TPS, a equipe técnica do Tribunal Superior Eleitoral - TSE teve a oportunidade de colocar à prova as correções feitas sobre os achados de uma edição do TPS no prazo mais curto possível. Imediatamente após o TPS 2017, a equipe do TSE apresentou relatório técnico com a indicação das correções que seriam implementadas[1]. Após cinco meses de trabalho, todas as vulnerabilidades encontradas em 2017 estão corrigidas. Nesse período, além de implementar as correções originalmente propostas, diversas outras melhorias de segurança foram realizadas, assim como adaptações nas propostas originais. Todos esse trabalho será apresentado neste documento.

Este relatório tem por objetivo apresentar as correções e contramedidas de software às vulnerabilidades e sugestões de melhorias apresentadas durante a última edição do TPS - é feito um breve resumo dos achados do TPS 2017 (vulnerabilidades, sugestões de melhoria e o seu impacto sobre o software) e são apresentadas as correções e contramedidas implementadas. Também é apresentado um breve relato do trabalho dos investigadores durante o Teste de Confirmação, realizado nos dias 7 e 8 de maio de 2018. Durante essa etapa, os investigadores tiveram a oportunidade de avaliar o trabalho realizado pela equipe da Seção de Voto Informatizado - Sevin, que é a unidade responsável pelo desenvolvimento do conjunto de software do Ecossistema da Urna, e também puderam repetir os testes que haviam realizado em novembro de 2017. Ao final, são feitos comentários sobre como o software da urna deve evoluir a partir daqui.

O objetivo deste documento é apresentar a visão da equipe técnica do TSE, tanto dos trabalhos realizados durante o TPS, quanto daquilo que foi feito para mitigação das vulnerabilidades. Este documento não se sobrepõe ao posicionamento da Comissão Avaliadora do TPS, tampouco aos registros realizados pela equipe de acompanhamento e pelos próprios investigadores.

Na verdade, espera-se justamente a conjunção das visões do TSE, da Comissão Avaliadora, dos investigadores e da comunidade técnico-científica para a construção de sistemas eleitorais cada vez mais seguros.

[1] http://www.tse.jus.br/hotsites/teste-publico-seguranca-2017/arquivos/tps2017-relatorio-tecnico.pdf

Vulnerabilidades e sugestões de melhoria

A seguir é feita uma breve análise das vulnerabilidades e sugestões de melhoria identificadas durante o TPS 2017.

Presença de chaves no ambiente de inspeção de código-fonte

A chave de criptografia do sistema de arquivos da urna eletrônica, que fazia parte do código-fonte do kernel do Linux, estava presente no ambiente de inspeção de código-fonte. O arquivo contendo essa chave deveria ter sido removido do ambiente de inspeção de código, mas encontrava-se à disposição dos investigadores.

Ao memorizar essa chave, o Grupo 1, liderado pelo Prof. Dr. Diego Aranha (Unicamp), foi capaz de decifrar uma das partições da flash de carga - FC, alterar arquivos e, dessa forma, explorar outras vulnerabilidades presentes no software.

Presença de chaves de criptografia no código-fonte

A chave de criptografia do sistema de arquivos da urna eletrônica fazia parte do código-fonte do kernel do Linux, como dito anteriormente. Com isso, a chave encontrava-se embarcada no binário do kernel e, por meio de técnicas de engenharia reversa foi possível recuperar essa chave.

O grupo liderado por Ivo Peixinho, perito da Polícia Federal (Grupo 4), foi capaz de analisar a memória do kernel do Linux em execução ao iniciá-lo numa máquina virtual. Pela análise da memória, o grupo foi capaz de recuperar a chave de criptografia do sistema de arquivos.

Bug na validação de assinatura de binários pelo kernel

A equipe da Sevin desenvolveu um módulo de kernel capaz de validar uma assinatura digital embarcada em arquivos ELF (*Executable Linux Format*), derivado do mecanismo já presente para a validação de módulos (arquivos *.ko*). Dessa forma, bibliotecas de link dinâmico e aplicativos da urna contém uma assinatura digital RSA de 4096 bits com SHA 512 embarcada em seu binário, que é validada pelo kernel antes de colocar o arquivo em execução. Essa assinatura é gerada durante o processo de lacração e somente a chave pública é incluída no kernel (a chave privada é destruída ao final do processo).

Na versão submetida ao TPS 2017 foi identificado um bug nesse mecanismo. Embora o cálculo do RSA estivesse correto, a função que valida a assinatura retornava um inteiro com sinal com valor negativo para indicar falha de assinatura. Esse valor estava sendo atribuído incorretamente a uma variável do tipo inteiro sem sinal. Em seguida, outra função verificava por um valor negativo sobre a variável anterior para determinar a suspensão da execução do kernel. Esse bug tinha como efeito a execução de bibliotecas adulteradas.

A presença desse bug permitiu ao Grupo 1 adulterar bibliotecas de link de dinâmico e executá-las na urna, provocando comportamento indevidos no software, tais como adulteração do log e de uma tela do Software de Votação. Esse bug permitiu também que a equipe tentasse modificar o software com vistas à adulteração de votos, ainda que o grupo não tenha obtido sucesso devido às validações de consistência das estruturas de dados pelo Software de Votação.

Existência de bibliotecas de link dinâmico sem assinatura complementar

Além da assinatura embarcada no binário e verificada pelo kernel, todos os arquivos executáveis da urna possuem assinatura digital complementar. Esse mecanismo de assinatura digital garante a integridade e autenticidade de qualquer arquivo na urna, independente da sua execução ou não. Trata-se de assinatura digital baseada em curvas elípticas de 256 bits desenvolvida pelo Cepesc/Abin. Essas assinaturas são geradas ao final do processo de lacração e somente a chave pública é incluída na urna (como um arquivo numa das partições da flash de carga - FC e da flash interna - FI). Cada assinatura é incluída num arquivo que contém uma lista de assinaturas de arquivos de um diretório. A verificação dessas assinaturas é feita por um *daemon*, de acordo com o solicitado pelo Software de Carga - SCUE[2] ou pelo Gerenciador de Aplicativos - GAP[3]. Caso seja encontrada uma assinatura inválida, o funcionamento da urna é interrompido.

Uma falha no conjunto de *scripts* do processo de lacração do TPS resultou na não inclusão de duas bibliotecas na lista de assinaturas do seu respectivo diretório. Dessa forma, tanto o SCUE quanto o GAP não solicitaram que a assinatura digital dessa biblioteca fosse verificada.

Essa falha, em conjunto com o bug na validação de bibliotecas pelo kernel, permitiu que o Grupo 1 avançasse em seu trabalho.

Uso de teclado externo

O software da urna não deveria permitir o uso de periféricos desconhecidos, o que inclui teclados USB. Contudo, o Grupo 1 foi capaz de utilizar um teclado externo USB após ter obtido sucesso em modificar uma biblioteca de link dinâmico para ecoar a digitação do teclado no console da urna.

O uso de um teclado externo foi possível porque o módulo USB HID Input foi indevidamente incluído na compilação do kernel utilizado durante o TPS 2017.

[2] Aplicativo executado pela urna para realização de seu processo de carga. Executado a partir da FC.

[3] Aplicativo executado pela urna para validações diversas e execução de outros aplicativos. Executado a partir da FI após a carga da urna.

Inicialização em máquina virtual

O software da urna não deveria permitir a sua inicialização fora do hardware da urna. Embora o mecanismo até então existente tenha funcionado, o que impediu a inicialização de qualquer aplicativo, o Grupo 4 foi capaz de iniciar o kernel do Linux numa máquina virtual. Ainda que o kernel tenha interrompido a sua execução antes de iniciar qualquer aplicativo, foi possível aos investigadores analisar a memória e recuperar a chave de criptografia do sistema de arquivos.

Parâmetros de segurança no compilador

Uma das sugestões de melhoria apresentadas pelo investigador Cassio Goldschmidt foi a utilização de um conjunto de parâmetros do compilador GCC associados a proteções de segurança. Uma delas é o suporte ao embaralhamento de endereços do espaço de memória dos executáveis. Caso esse mecanismo estivesse habilitado, os ataques realizados sobre o Software de Votação pelo Grupo 1 teriam sido dificultados substancialmente.

Validação de comentários em arquivos JPEG

Mais uma das sugestões de melhoria apresentadas pelo investigador Cassio Goldschmidt foi a inclusão de validações sobre o campo de comentários do cabeçalho do arquivo JPEG. Embora não tenham sido identificadas vulnerabilidades associadas à carga de arquivos JPEG, a ausência de validações sobre o campo de comentário poderia expor o software a defeitos relacionados à *buffer overflow* na leitura desse campo.

Respostas aos achados do TPS

Faz-se agora uma breve descrição das implementações realizadas pela equipe da Sevin sobre as vulnerabilidades e sugestões de melhoria identificadas durante o TPS 2017.

Segregação das chaves contidas no código-fonte em *headers* separados

As chaves de criptografia presentes no código-fonte foram segregadas em arquivos *headers* separados. Com isso, foi minimizada a quantidade de código que precisava ser retirada do ambiente de inspeção de código-fonte. A partir daí também foram criados *scripts* para automação do processo de retirada dessas chaves, assim como procedimentos de verificação de retirada de todos os arquivos.

Essa foi uma ação temporária, enquanto a retirada das chaves do código-fonte não era concluída.

Correção do bug na validação de assinatura de binários pelo kernel

Como descrito anteriormente, o bug presente na versão utilizada no TPS 2017 era de simples correção e foi devidamente tratado, com ajustes no tipo da variável e na estrutura condicional que pode determinar a suspensão da execução do kernel.

Além disso, foi feita um revisão dos procedimentos de teste dessa funcionalidade do kernel. Dessa forma, foram reforçados os testes sobre a assinatura digital de módulos, bibliotecas e aplicativos.

Correção dos *scripts* de assinatura usados no processo de lacração

A presença de bibliotecas de link dinâmico sem assinatura digital complementar se devia à sua ausência nos *scripts* responsáveis pela assinatura de arquivos da urna. Todos os scripts foram revisados e agora não há mais arquivos sem esse tipo de assinatura. Também foram criados procedimentos de teste automatizados e manuais, que garantem que todos os arquivos que deveriam estar assinados assim o estão de fato.

Minimização da quantidade de bibliotecas de link dinâmico no Uenux

Até o TPS 2017, o Uenux estava organizado de forma que as porções de software comuns entre os aplicativos eram embarcadas como bibliotecas de link dinâmico, carregadas quando o aplicativo é iniciado. Esse cenário resultava num grande número de bibliotecas gravadas nos cartões de memória.

A partir de agora, a estratégia é a oposta: as porções de software comuns entre os aplicativos são embarcadas como bibliotecas de link estático, fazendo parte do binário de cada aplicativo. Dessa forma, o processo de assinatura digital das bibliotecas é simplificado, reduzindo a possibilidade da existência de arquivos não assinados, e reduz-se também a superfície de ataque na medida em que o número de bibliotecas de link dinâmico é mínimo.

As bibliotecas de log e derivação de chaves, que foram atacadas pelo Grupo 1 no TPS 2017, agora são de link estático e não se encontram mais presentes nos cartões de memória das urnas.

Retirada das chaves contidas no código-fonte

As chaves de criptografia presentes no *bootloader* e no kernel foram removidas. Foi implementado um mecanismo de derivação de chaves a partir de uma informação presente somente na extensão de BIOS da urna eletrônica. A partir de agora o *bootloader* deriva a chave que decifra a imagem do kernel do Linux. O kernel deriva a chave de criptografia do sistema de arquivos e a chave de criptografia que protege as chaves utilizadas pelos aplicativos da urna.

A informação presente na BIOS, e que é utilizada para a derivação de chaves, é chamada de *tabela cripto*. Trata-se de 1024 bytes aleatórios gravados na extensão de BIOS de todas as urnas. Quando o BIOS entra em execução a tabela cripto é carregada para a memória e fica disponível para o *bootloader* e o kernel do Linux.

O processo de derivação de chaves tem início durante a compilação do software (Figura 1). Antes que o *bootloader* e o kernel sejam compilados, é necessário que sejam geradas *tabelas de posições*, que possuem as posições da tabela cripto, cujos respectivos valores devem ser copiados e servirão de entrada para o processo de derivação de cada chave. Essas tabelas de posições são geradas com o auxílio do TRNG da urna. Uma vez geradas as tabelas, são criados arquivos *header* com seus valores para inclusão na compilação do *bootloader* e do kernel.

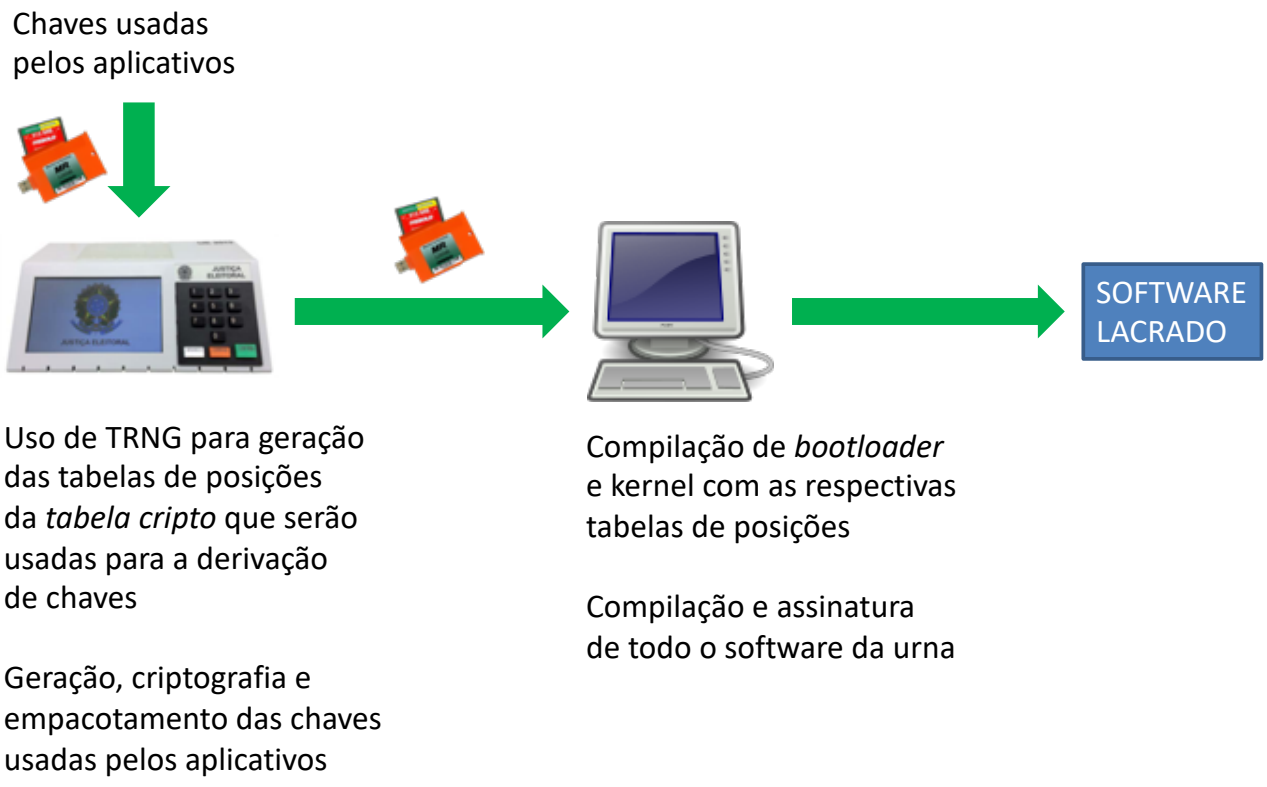

Figura 1 - Derivação e empacotamento de chaves.

Além da geração das tabelas de posições, a urna eletrônica também recebe, gera e criptografa chaves que são usadas pelos aplicativos (assinatura de arquivos de saída da urna, QR Codes e relatórios impressos, por exemplo) e as cifra a partir da chave derivada com a respectiva tabela de posições. Dessa forma, as chaves utilizadas pelas aplicações já saem cifradas pela urna e somente nela podem ser abertas.

No *bootloader*, o processo de derivação da chave de criptografia do kernel tem início no cálculo do hash SHA 512 do *bootstrap* da MBR (Figura 2). A partir desse hash é derivado o endereço de memória onde encontra-se a tabela cripto. Usa-se então a tabela de posições embarcada no *bootloader* para a seleção de valores da tabela cripto - cada valor da tabela de posições é usado como índice para a seleção de um valor na tabela cripto. Esses valores são então submetidos a um HMAC, que deriva a chave de criptografia AES CounterMode 256 utilizada para decifrar a imagem do kernel.

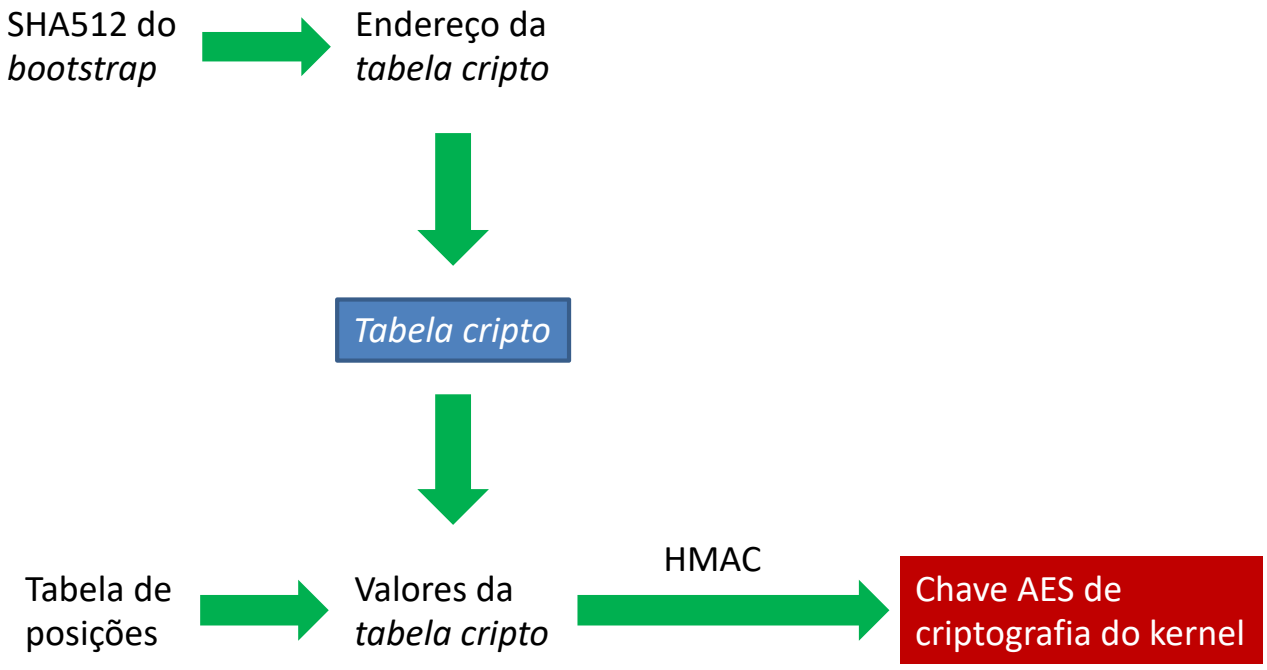

***Figura 2 - Derivação de chave no* bootloader.**

No kernel, o processo de derivação de chaves é semelhante (Figura 3). O kernel recebe como parâmetro do *bootloader* o hash SHA 512 de setores do próprio *bootloader*. A partir desse hash é derivado o endereço de memória onde encontra-se a tabela cripto. Usa-se então a tabela de posições embarcada no kernel para a seleção de valores da tabela cripto. Esses valores são então submetidos a um HKDF, que deriva a chave de criptografia AES XTS 256 utilizada no sistema de arquivos da urna. O mesmo processo é utilizada para derivar a chave AES CBC 256 utilizada para a criptografia das chaves usadas pelos aplicativos, mas utilizando outra tabela de posições.

Dessa forma, as chaves não encontram-se mais embarcadas no código-fonte e nem nos binários. As chaves são derivadas a partir de uma informação aleatória gerada durante o

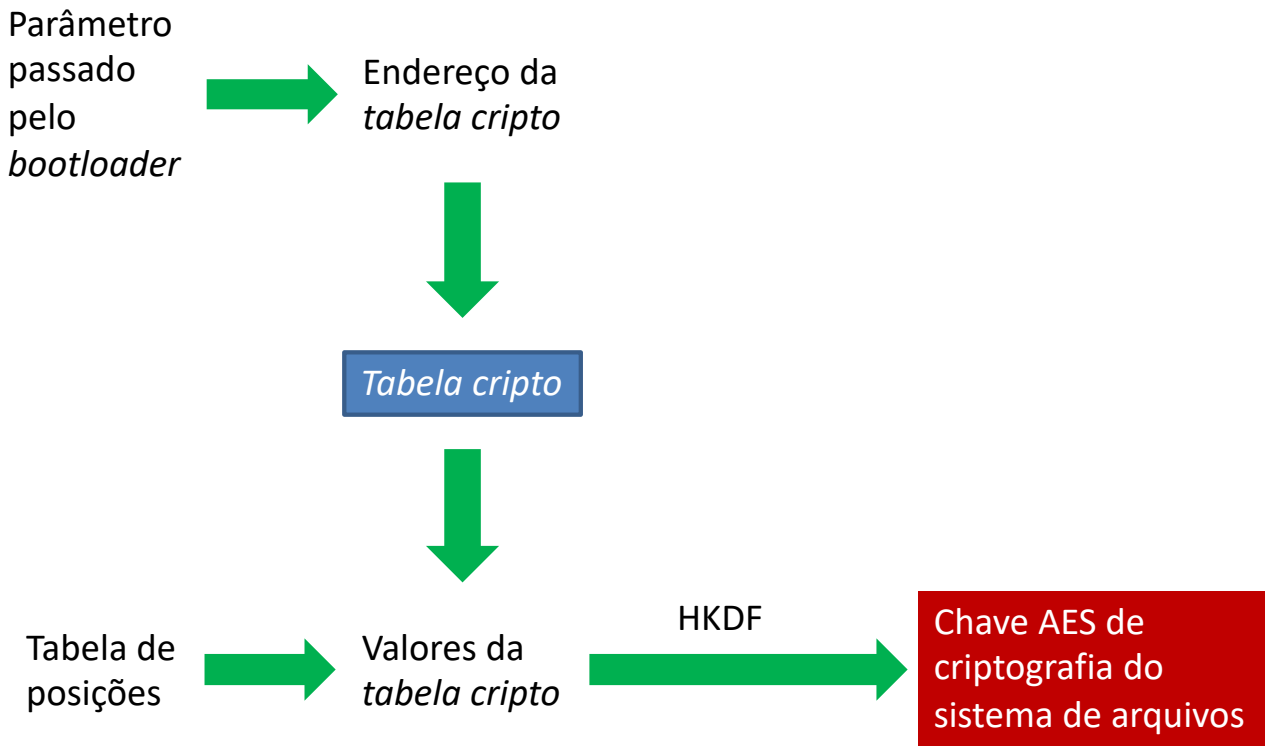

Figura 3 - Derivação de chaves no kernel. O mesmo processo é utilizada para a chave que decifra as chaves usadas pelos aplicativos.

processo de lacração (tabela de posições) em conjunção com uma informação presente somente no hardware das urnas (tabela cripto).

Validação do hardware da urna pelo kernel

Foi incluída nas rotinas iniciais do kernel do Linux uma função que valida o hash SHA 512 de parte das informações do /proc/cpuinfo. Dessa forma, o kernel valida se está sendo executado numa máquina com um dos modelos de processador usados pelas urnas eletrônicas. Caso o kernel não esteja sendo executado no hardware esperado, a sua execução é interrompida. Hoje as urnas utilizam duas alternativas de processador: um modelo de AMD Geode e outro modelo de Intel Atom. Essa validação dificulta bastante a execução do kernel num ambiente virtualizado.

Validação do BIOS no *bootloader* e no kernel

Com o uso da tabela cripto, presente somente no BIOS utilizado nas urnas eletrônicas, uma consequência é bloqueio do uso do software fora da urna. Na medida em que chaves criptográficas são derivadas a partir de informação presente no BIOS, caso essa informação não esteja presente ou esteja incorreta, deriva-se a chave errada, a decifração não é válida e o software não funciona.

Validação do *bootstrap* do MBR pelo *bootloader*

O ataque realizado pelo Grupo 4 durante o TPS teve início com a adulteração do *bootstrap* da MBR da flash de carga para que carregasse o *bootloader* presente na flash (o código do *bootstrap* normalmente não carrega nada e apresenta mensagem de erro se executado fora da urna), para em seguida decifrar o kernel e carregá-lo. A partir de agora, o *booloader* valida uma porção do hash SHA 512 do *boostrap*. Caso o hash esteja inválido o boot é interrompido. Em seguida, o hash também é usado para a derivação do endereço de memória da tabela cripto. Caso o *bootstrap* seja modificado, o *bootloader* apontará para o endereço errado e acessará valores inválidos para a tabela cripto. Com isso, a chave usada para decifrar o kernel será derivada incorretamente e o kernel decifrado será inválido e não executável.

Outra validação embutida no *bootloader* é o hash SHA 512 de alguns dos seus setores. Caso essa porção do *bootloader* seja modificada, o boot é interrompido. Esse mesmo hash será usado pelo kernel mais adiante para derivação do endereço da tabela cripto. Dessa forma, cria-se uma dependência entre o *bootloader* o o kernel. Caso esse hash esteja incorreto, a chave de criptografia do sistema de arquivos derivada é incorreta e o kernel não consegue iniciar qualquer coisa.

Retirada do suporte no kernel a dispositivos não utilizados

O módulo USB HID Input foi retirado da compilação do kernel. Além disso, foi feita uma revisão geral das configurações do kernel e todo o suporte a hardware não presente na urna foi removido. Finalmente, foram criados procedimentos de testes que certificam que dispositivos desconhecidos de fato não são reconhecidos pelo kernel quando conectados à urna.

Desconexão de dispositivos USB em portas não utilizadas

A urna eletrônica possui portas USB adicionais (duas na traseira do terminal do eleitor e uma na lateral do terminal do mesário), previstas para a conexão de novos periféricos, tais como o módulo impressor de votos. Foi incluído um mecanismo que faz a desconexão lógica de dispositivos ligados às portas que não devem ser utilizadas. Esse tratamento é feito em espaço de usuário, pelo *daemon* udev, que desconecta qualquer dispositivo inesperado, tanto durante quanto após a inicialização da urna.

Atualização frequente do kernel do Linux

Já no TPS 2017 era utilizada uma versão do kernel do Linux com suporte de longo prazo (versão 3.18.48). Dentro desse mesmo ramo de versões, o kernel foi atualizado para a versão 3.18.102 para o Teste de Confirmação e deve receber novas atualizações até a lacração das Eleições 2018.

Essas atualizações mitigam diversas vulnerabilidades, incluindo inúmeras relacionadas ao suporte de dispositivos USB.

Utilização de todos os parâmetros de segurança do GCC sugeridos

Todos os parâmetros de segurança sugeridos para uso com o GCC (compilador utilizado no Ecossistema da Urna) foram adotados. Entre eles, destaca-se o parâmetro -fPIE, que gera aplicativos compatíveis com o suporte ao embaralhamento de endereços do espaço de memória dos executáveis.

Validação de comentários em arquivos JPEG

Os arquivos JPEG passaram a ter o cabeçalho validado quanto ao campo de comentários. O aplicativo Gedai-UE, responsável pela geração das mídias que preparam as urnas para as eleições, agora exige que o campo de comentários do JPEG esteja vazio. Dessa forma, impede-se qualquer tentativa de ataques de *buffer overflow* sobre esse campo.

Teste de Confirmação do TPS 2017

Durante os dias 7 e 8 de maio de 2018, as equipes que obtiveram algum sucesso durante o TPS 2017 puderam verificar as correções implementadas pela equipe técnica do TSE e validar a sua efetividade.

Estiveram presentes os integrantes do Grupo 1 (liderado pelo Prof. Dr. Diego Aranha) e do Grupo 4 (liderado pelo perito Ivo Peixinho). As equipes tiveram acesso a todo o código-fonte do software corrigido, assim como do software usado em novembro de 2017 - o que permitiu aos investigadores avaliar as diferenças entre cada versão. A pedido dos investigadores, também foi disponibilizado o código-fonte da extensão do BIOS da urna (sem incluir a tabela cripto) e do kernel do Linux em sua versão original, sem as extensões desenvolvidas pela Sevin (o que permitiu aos investigadores avaliar com mais propriedade quais foram as intervenções feitas pela equipe técnica do TSE).

O Grupo 1 foi capaz de inspecionar as correções diretamente relacionadas a assinatura digital e entenderam que as ações foram efetivas e que o problema foi adequadamente tratado. Com isso, não é mais possível modificar bibliotecas de link dinâmico e carregá-las na urna, com o intuito de afetar o comportamento dos aplicativos.

O Grupo 4 conseguiu modificar o *bootloader*, de modo a ignorar as verificações diretas sobre o hash do *bootstrap* da MBR e do hash de setores do próprio *bootloader*. Contudo, não obtiveram qualquer sucesso nas tentativas de iniciar o kernel numa máquina virtual. Em todas as tentativas, a derivação da chave de criptografia da imagem do kernel produziu uma chave inválida e, consequentemente, tentou-se colocar em execução um kernel inválido e não executável. A avaliação do Grupo 4 foi de que as contramedidas implementadas no software foram efetivas para impedir a execução do software fora da urna.

Conclusões e trabalhos futuros

A edição do TPS de 2017 contou com a presença de pesquisadores e profissionais altamente qualificados em técnicas de criptografia, desenvolvimento de software seguro e engenharia reversa. Os achados dos investigadores provocaram correções e melhorias fundamentais para que o conjunto de software do Ecossistema da Urna atingisse um patamar ainda mais elevado de segurança e robustez para as Eleições 2018.

Durante o Teste de Confirmação os investigadores tiveram a oportunidade de avaliar todas as modificações realizadas no software. E todas as correções e melhorias se mostraram efetivas e impediram a reprodução, ou mesmo derivação, dos ataques executados em novembro de 2017.

O trabalho de aprimoramento da segurança do conjunto de software do Ecossistema da Urna não se encerra aqui. Seguindo o plano traçado no relatório publicado em dezembro de 2017, novas melhorias serão implementadas até a lacração do software para as Eleições 2018. E há trabalho também para depois das eleições deste ano, quando está prevista a substituição do mecanismo de derivação de chaves apresentado neste documento pela utilização do MSD[4]. A utilização das plenas capacidades do MSD só é possível quando os modelos mais antigos de urna eletrônica, que não contam com esse dispositivo, forem retiradas de operação, o que deve ocorrer para as Eleições 2020.

Além do Teste Público de Segurança, a comunidade técnico-científica pode contribuir para o aprimoramento da segurança do conjunto de software do Ecossistema da Urna no período previsto para acompanhamento do desenvolvimento do software[5]. Trata-se de um período de seis meses que antecede à lacração do software, no qual é possível auditar todo o código-fonte.

Apesar da comprovada efetividade das ações apresentadas aqui para a mitigação das falhas encontradas durante o TPS 2017, a equipe da Sevin está aberta a sugestões dos próprios investigadores e da comunidade técnico-científica em geral. Críticas e sugestões podem ser enviadas para sevin@tse.jus.br.

---

[4] *Master Secure Device* - dispositivo de segurança embarcado na urna, responsável pelas verificações da cadeia de segurança do boot, e que possui a capacidade de geração e guarda segura de chaves criptográficas.

[5] Resolução TSE nº 23.550/2017 - http://www.tse.jus.br/legislacao-tse/res/2017/RES235502017.html