From random block corruption to privilege escalation

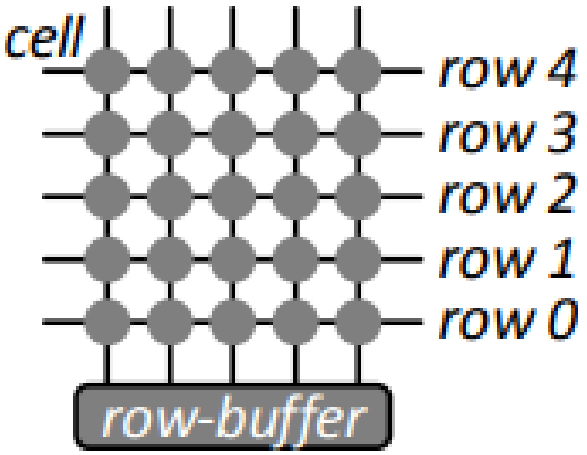
Alunos: Guilherme Calin
Marcio Antonio Coltro Sczepanski

- Quando projetando software, aspectos do hardware são geralmente abstraidos.

-Ataques com acesso ao hardware : cold boot

-Ataques sem acesso ao hardware: mais problematicos

-RowHammer :A expansão contínua da tecnologia de processo DRAM tem permitiu que células menores fossem colocadas mais próximas umas das outras.



-Não é apenas DRAM que contêm dados sensiveis.

-Memória Flash apreseta problemas de confiabilidade.

-Repeated program and erase.

-Cell to Cell interfearence (CCI).

-Instabilidade de tensão após repetidos ciclos de Read-Only.

-Leitura antes de finalizar a programação.

-Medidas adotadas para melhorar a confiabilidade.

-Scrambling.

-Error Correcting Codes (ECC).

-Para aplicar o ataque de ganho de privilégio existem alguns desafios.

-Aplicar CCI

-Driblar o EEC + scrambler

-Controlador SSD

-OS

-USER

-Tendo o ECC, o melhor que o atacante pode esperar é uma incontrolável modificação completamente aleatória da página/bloco da memória.

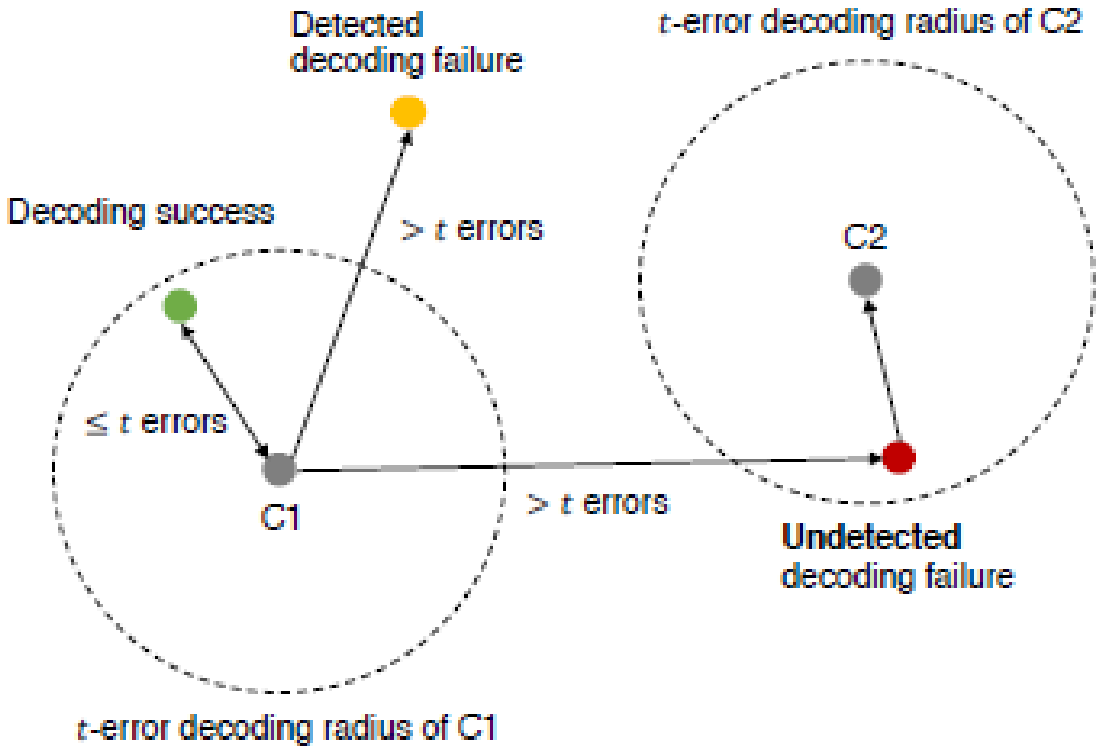
-ECCs geralmente corrigem uma grande quantidade de erros(típico t=50) corrigindo ele para a “palavra-chave” esperada.

-Casos possíveis.

-1.A quantidade de erros inseridos é menor ou igual a quantidade definida na correção pelo ECC, corrige e retorna sucesso.

-2.A quantidade de erros inseridos é maior e o vetor binario resultante não cai dentro de outra palavra-chave, ECC retorna erro o qual será tratado pelo SO.

-3.Os erros inseridos são maiores que t e formam um vetor binario que resulta em outra palavra-chave, retorna sucesso!


Detected
decoding failure
t-error decoding radius of C2
Decoding success
> t errors
C2
≤ t errors
C1
> t errors
Undetected
decoding failure
t-error decoding radius of C1

-Então, os possíveis ataques:

1.Mudança de apenas 1 bit de uma forma controlada.(DRAM)

2.Mudança de apenas 1 bit de uma forma não controlada.(DRAM)

3.Modificação não controlada em um bloco inteiro de memória(vários bits).(FLASH!)

4.Corromper um bloco(erro).(FLASH!)

- 1 e 2 não são possíveis devido ao ECC.

- 4 é altamente improvável de ser usada para escalonamento de prioridade(ECC erro).

- Focar no 3.

-Um ataque (3) deve cumprir as seguintes limitações.

-1. A corrupção do bloco atacado deve ter probabilidade baixa ou nula de causar um erro fatal.

-2. O bloco atacado deve ser um bloco que é escrito com frequência.

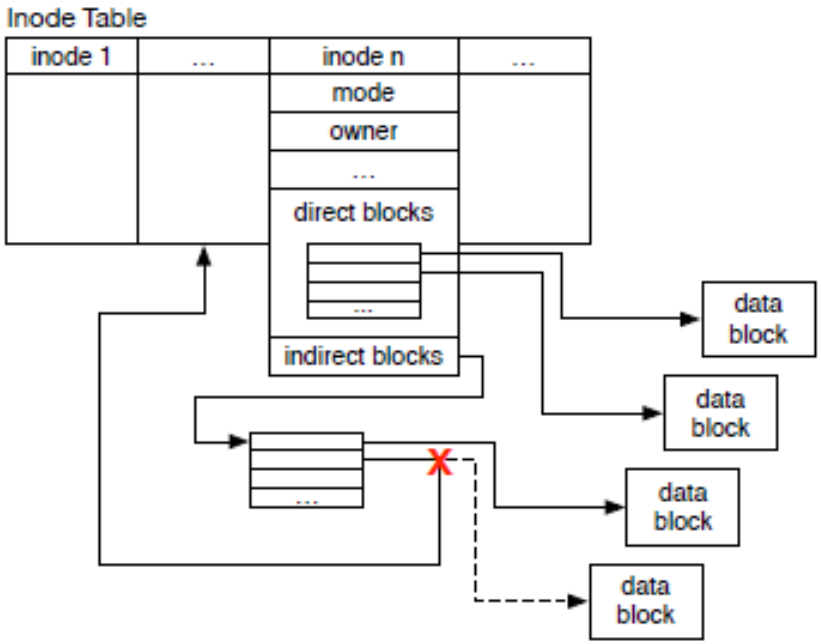
-3. Deve existir a probabilidade do bloco a ser corrompido gerar alguma condição exploravel.

-4. Fazer flush no cache com objetivo de forçar o SO a acessar data corrompida da memória flash.

- Ambiente do ataque – Linux com sistema de arquivos ext3

- Explora o sistema de blocos indiretos que apontam para blocos diretos, os quais contem os dados.

- O ataque começa corrompendo um ponteiro de um bloco de dados dentro de um bloco indireto, procurando algum ponteiro que aponte para um inode.



- Após corromper um bloco indireto com sucesso, para cada um dos 1024 ponteiros de dados continos nele existem as seguintes possibilidades.

- O bloco é interessante (tabela inode, binario imporante).

- O bloco não é interessante (bloco de dados qualquer sem necessidade de privilégios).

- O bloco está “fora” do sistema (aponta para algum lugar que não existe, retorna erro não fatal).

- Caso o atancante consiga corromper um bloco de dados que aponta para inode – modifica SUID bit on – aponta para um bloco de shell – eleva sua prioridade.

- Considerando um sistema de arquivos de 100GB com blocos de 4KB – 9% de chance para ataque de elevar prioridade.

- Ataque tambem possível com blocos indiretos duplos.

- Um nível acima.

- Chance de 99.7% de encontrar um bloco explorável que acabem em escalamento de privilégios.

- Ataque dependente de um sistema de arquivosque usa blocos indiretos (ext2,ext3)

- Acredita-se ser possível em um sistema ext4, porem com probabilídade muito reduzida.

From random block corruption to privilege escalation

Alunos: Guilherme Calin
Marcio Antonio Coltro Sczepanski