Uso de Técnicas de Proteção de Software em Sistemas Embarcados

Lucila Bento e Raphael Machado[1]

[1]Instituto Nacional de Metrologia, Qualidade e Tecnologia

10 de dezembro de 2018

Motivação

1. Estima-se que as perdas de empresas norte-americanas em 2017 com crimes cibernéticos tenha sido de US$ 21 bilhões
2. 48% das empresas americanas pesquisadas sofreram uma violação de dados
3. 34% já sabiam da vulnerabilidade antes de serem atacados
4. 62% das empresas pesquisadas não sabem dizer se as vulnerabilidades de software estão sendo corrigidas de maneira oportuna
5. 57% dizem que seus esforços de correção falharam porque suas equipes ainda não detém conhecimento ou não usam os recursos adequados

Ponemom Institute, Today's State of Vulnerability Response: Patch Work Demands Attention, 2018.

Segurança em Sistemas Embarcados

1. Dispositivos imersos em ambientes não supervisionados
   - Serviços convencionais tem sido migrados para nuvem
2. Requisitos de segurança no projeto dos dispositivos
3. Limitação de hardware
4. Competitividade entre concorrentes

Soluções

1. Requisitos não funcionais (de segurança)
2. Treinamento e capacitação de desenvolvedores ainda não resolve
   - Desenvolvedores ainda não tem maturidade em segurança (somente requisitos funcionais)
3. Falhas de segurança são identificadas em ambiente de produção
   - A segurança tem sido contemplada em compiladores e SOs

Segurança em Sistemas Embarcados

1. Uma vez que tenhamos os requisitos de segurança, como avalia-los?
   - Avaliação por terceiros
2. Especificação de RTMs para evolução da segurança
   - Aprimoramento de como testar
3. Como avaliar a segurança (design, código e testes) em um ambiente tão dinâmico e heterogêneo?
4. Até agora tivemos questões processuais, mas como podemos empregar técnicas/tecnologias?

Proteção de Software: Objetivos

1. Resistir a engenharia reversa estática e dinâmica
2. Resistir a modificações não autorizadas
3. Resistir a clonagem de software
4. Resistir a spoofing
5. Esconder segredos estáticos e dinâmicos (criação, transmissão, utilização)
6. Impedir a distribuição de programas "crackeados"

1. **Ofuscação** — proteção contra engenharia reversa
2. **Incorruptibilidade** — proteção contra modificação/monitoramento
3. **Marca d'água** — identificação de autoria e rastreamento de propriedade

Proteção de Software

1. Como proteção de software pode auxiliar na segurança de sistemas não supervisionados
2. Dificultar o entendimento de um código dificultará um agente externo (ameaça) a fraudar o sistema
    - Um sistema fácil de ser entendido também é fácil de ter alguma fraqueza encontrada...
3. Ofuscação
4. Incorruptibilidade
    - Garantir o fluxo lógico e de dados incrementa a segurança
    - Muitos ataques acontecem devido a falhas de segurança em que um atacante pode desviar o fluxo de execução lógico e de dados do programa
5. Garantir a não distribuição do código
    - Marca d'água

Proteção de Software: Referências

1. Área militar dos EUA investe pesado em técnicas anti engenharia reversa

   Michael Peebles and Patrick Jungwirth, Software-Based Anti-Tamper Technique Research and Development, 2014.

2. Skype vem investindo em técnicas de incorruptibilidade e ofuscação ao longo dos anos

   Documentação Microsoft, URL: `https://docs.microsoft.com/en-us/skype-sdk/sdn/articles/subscriber-obfuscation`. Acessado em 07 de dezembro de 2018.

Ofuscação

O que significa ofuscar um programa?

- Transformá-lo em um outro programa de mesma semântica, contudo mais complicado de ser entendido ou alterado por um adversário em relação ao programa original

Dificuldade

- Maior tempo de análise
- Mais dinheiro
- Mais poder computacional

Ofuscação: Exemplo

```
public class C {
   static int gcd(int x, int y) {
      int t;
      while (true) {
         boolean b = x % y == 0;
         if (b) return y;
         t = x % y; x = y; y = t;
      }
   }
   public static void main(String[] a){
       System.out.print("Answer: ");
       System.out.println(gcd(100,10));
   }
}
```

```
public class C {
 static int gcd(int i, int j) {
  int t8, t7, k;
  for (;;) {
    if (i%j==0) {t8=1;t7=0;}
    else        {t8=0;t7=0;}
    if ((t7^t8)!=0)
      return j;
    else {
      k=i%j; i=j; j=k;
    }
  }}
 public static void main(String[] Z1) {
  System.out.print("Answer: ");
  System.out.println(gcd(100, 10)); }
}
```

Incorruptibilidade

- Incorruptibilidade verifica se o código foi modificado e toma uma ação diante de modificação
- Incorruptibilidade x Ofuscação

Incorruptibilidade

- Incorruptibilidade verifica se o código foi modificado e toma uma ação diante de modificação
- Incorruptibilidade x Ofuscação

- Exemplo ingênuo:
  - `if (tampering-detected()) abort();`

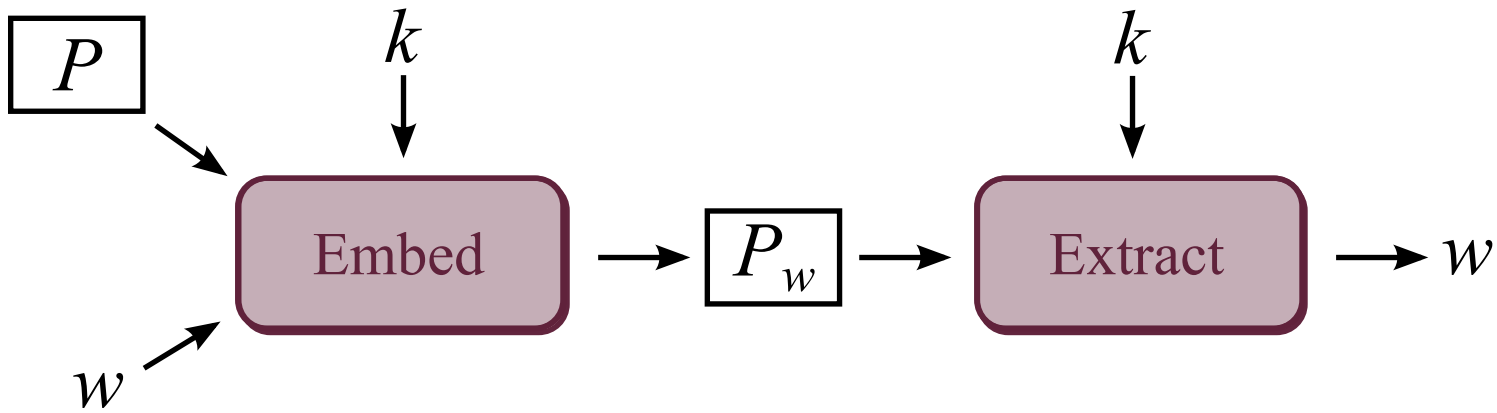
Marca d'água de Software

Embarca um identificador no código a fim de tornar possível rastre-lo posteriormente

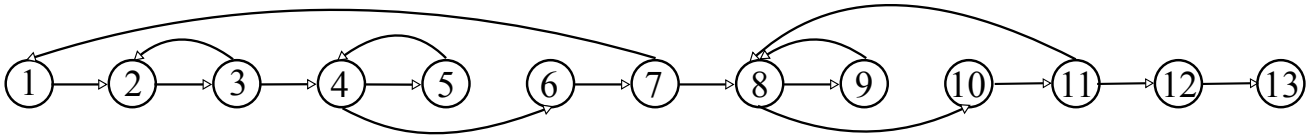
- Autoria, propriedade e/ou usuário

Esquema consiste em

- $encode(w) \rightarrow G$
- $decode(G) \rightarrow w$
- $embed(P, w, k) \rightarrow P'$
- $extract(P', k) \rightarrow w$



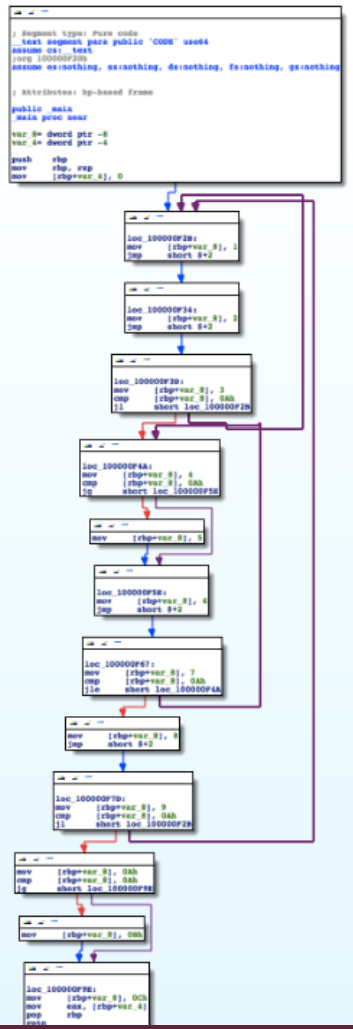
Marca d'água de Software baseada em grafos: Exemplo




```
int main(){

    int x;
    do { // L9 -> L1
        do { //L3 -> L1
            x=1;
        __asm__ ( "jmp L2;" "L2:;");
        x=2;
        __asm__ ( "jmp L3;" "L3:;");
        x=3;
        } while (x<10);
        do {
            x=4;
            if (x <=10)
                x=5;
            x=6;
            __asm__ ( "jmp L7;" "L7:;");
            x=7;
        } while (x<=10);
        __asm__ ( "jmp L8;" "L8:;");
        x=8;
        __asm__ ( "jmp L9;" "L9:;");
        x=9;
        }while (x<10);
        x=10;
        if (x <=10)
            x=11;
        x=12;
}
```

$\Longrightarrow$



Concluindo...

- Quais técnicas são apropriadas para uso em sistemas embarcados?
- Como os fabricantes podem adotar o uso de tais técnicas no desenvolvimento de seus dispositivos?
- Como os RTMs estão relacionados com as técnicas de proteção de software?
  - Necessário adicionar novos requisitos aos RTMs existentes?
- Como testar as técnicas utilizadas?

Obrigada!

Uso de Técnicas de Proteção de Software em Sistemas Embarcados

Lucila Bento e Raphael Machado[1]

[1]Instituto Nacional de Metrologia, Qualidade e Tecnologia

10 de dezembro de 2018