Analiza performanței sistemului

Viteză de execuție. Consum de memorie. Consum de putere. Securitate

Analiza performanței sistemului

- Timp de execuție
- Consum de memorie
- Consum de putere
- Securitate

Timpul de execuție

Foarte important în sisteme embedded!

Probleme în analiza timpului de execuție:

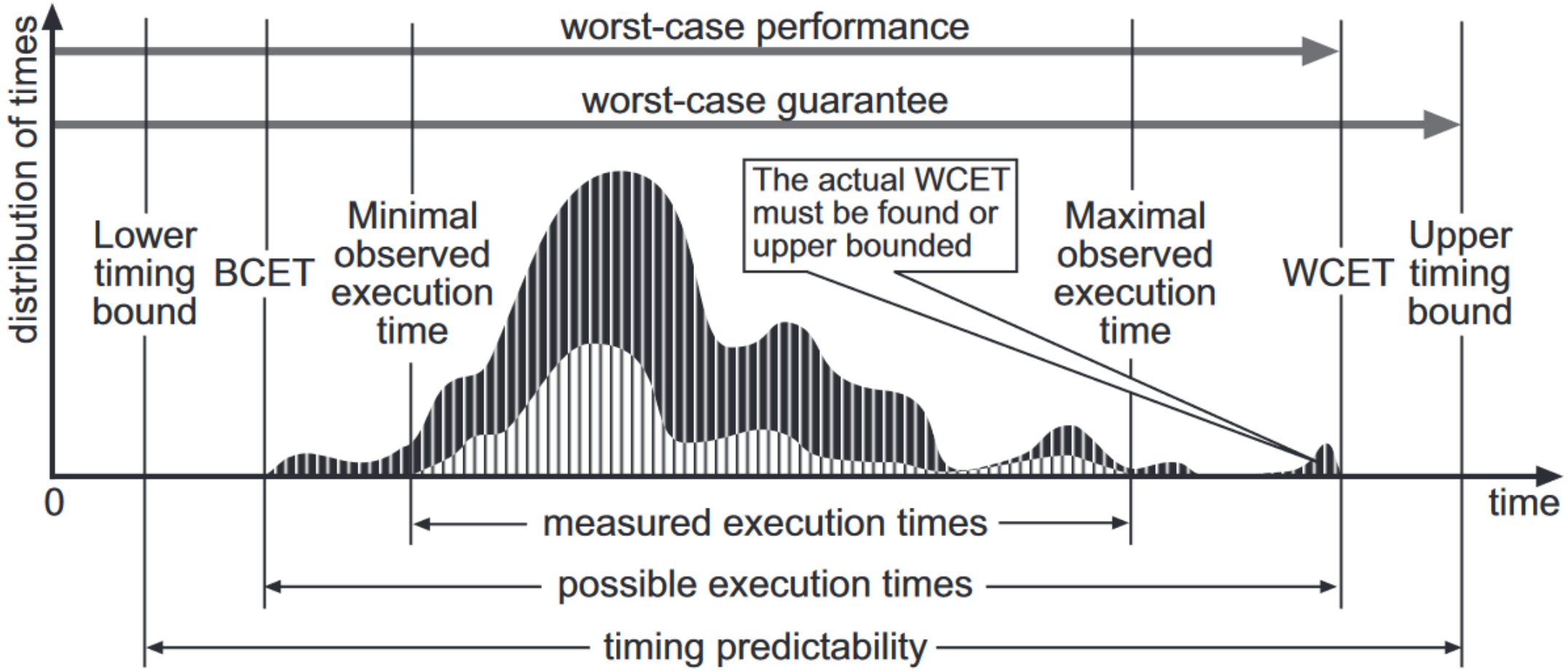
- Estimarea Worst-Case Execution Time (WCET)
- Estimarea distribuției timpilor de execuție
- Cazuri limită
- Simulare software-in-the-loop

Worst-Case Execution Time (WCET)

- WCET = Cel mai lung timp necesar unui task pentru a finaliza execuția
  – dependent de valorile de intrare și de condițiile mediului

- BCET (Best-Case Execution Time) = timpul cel mai scurt necesar pentru execuția task-ului

Fișiere header



R Wilhelm et al., The worst-case execution-time problem—overview of methods and survey of tools, ACM Transactions on Embedded Computing Systems, 2008

Problema determinării WCET

Problema: Având codul pentru un task și cunoscând sistemul pe care se va executa (sistem de operare, HW) să se determine WCET-ul task-ului.

În general WCET nu poate fi determinat!

Abordări în analiza timpului de execuție

- Analiza flow-ului programelor
  - Se determină cea mai lungă cale din program
  - Se determină limitele buclelor
  - Se identifică direcții fezabile din program
  - Se identifică dependițe între fragmente de cod diferite
- Analiza comportamentului procesorului
  - Pentru fragmente mici de cod se generează limite pentru timi de execuție pe platforma studiată
  - Se modelează detalii ale arhitecturii inclusiv comportamentul memoriei cache, al pipelinenurilor, predicția branch-urilor, etc.

Rezultatele fiecărei analize sunt folosite pentru cealaltă analiză.

Abordare clasică

1. Realizarea “manuală” a modelului comportamental pentru procesor
2. Utilizarea modelului pentru a determina cele mai defavorabile stări de pornire pentru fiecare bloc de bază → se măsoară timpul de execuție pentru aceste blocuri pornind din stările determinate
3. Pe baza acestor măsurători se definesc limite superioare pentru fiecare bloc
4. Se formulează un program liniar pentru determinarea sumei maxime a limitelor superioare de-a lungul path-ului analizat

Cum măsurăm timpul de execuție

Diferite tehnici cu grad de acuratețe diferit:

- Utilizarea codului sursă pentru măsurarea timpului de execuție pe baza counterului de cicli al procesorului
- Utilizarea de simulatoare
- Utilizarea de analizoare logice

Probleme în măsurarea timpului de execuție

- Instrumentarea adaugă întârzieri suplimentare
  - Se compensează prim măsurarea unor secvențe de cod suficient de lungi
- Utilizarea multitasking-ului face ca numărătorul de cicli să fie incrementat chiar dacă task-ul este suspendat
  - Se realizează mai multe măsurători și se procesează timpul bazat pe acestea
- Sisteme Multicore\hyperthreading
  - Task-ul evaluat trebuie să fie executat de un singur core
- Managementul consumului de putere
  - Viteza procesorului poate să varieze. Numărătorul poate să fie resetat în low power mode

Consumul de memorie

- Consum de memorie RAM
  - Dimensiunea memoriei RAM disponibile trebuie adaptată la consumul maxim posibil în sistem în cel mai defavorabi caz
- Consum de memorie ROM
  - În general destinat pentru stocarea programului – dimensiunea trebuie să permită eventuale update-uri
  - Stocare dinamică de informații – memoria trebuie să acopere necesarul de stocare

Evaluarea consumului maxim de memorie RAM

Componente ale memoriei RAM:

- Secțiune statică – dimensiunea este cunoscută la compilarea programului
- Stivă – spațiu alocat static, managementul dinamic nu garantează depășirea spațiului alocat
- Heap – spațiu alocat static, la fel ca în cazul stivei acest spațiu poate să fie depășit în funcție de necesități

Determinarea dimensiunii necesare pentru stivă

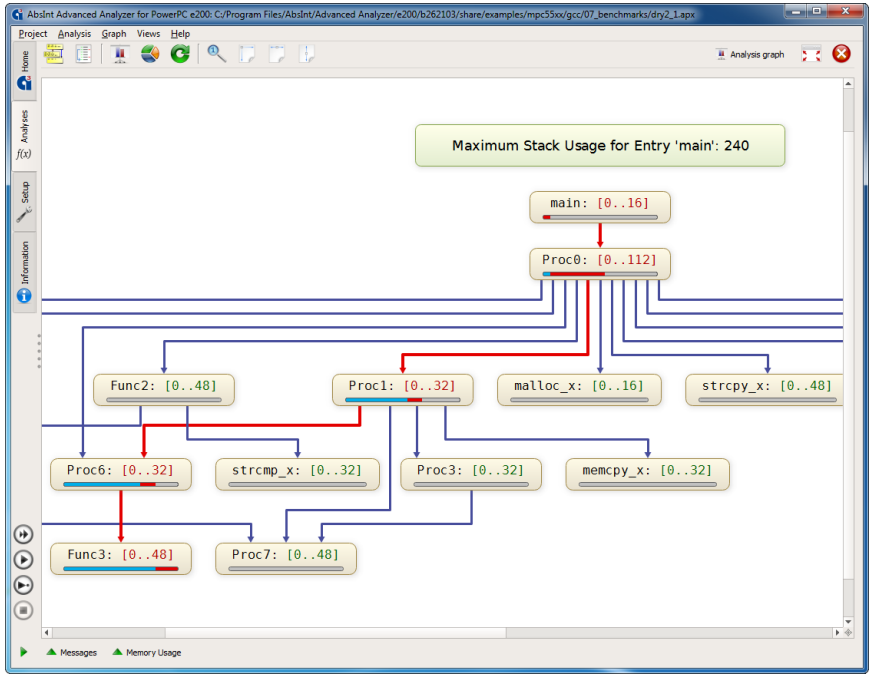
Abordare frecventă:

1. Se alege o dimensiune inițială mare pentru stivă
2. Prin codul de inițializare (startup code) se umple stiva cu un pattern predefinit (de ex., 0x55, 0xCD)
3. Se efectuează teste definite pentru a trece sistemul prin toate stările posibile
4. Se analizează stiva după testare pentru a identifica dimensiunea zone în care patternul a fost suprascris

Determinarea dimensiunii necesare pentru stivă

Analiză statică:

- Se utilizează pachete software sau module capabile să determine automat cel mai mare consum de memorie pentru stivă la care se poate ajunge.
- Exemplu: AbsInt StackAnalyzer



Determinarea consumului de memorie ROM

- Dimensiunea alocată pentru memoria program și cea dedicată constantelor se poate afla în urma procesului de link

- În cazul utilizării dinamice a memoriei ROM se poate calcula consumul în funcție de regimul de funcționare

Consum de putere

Surse pentru consumul de putere:

- Procesor
- Frecvența de lucru
- Module interne
- Periferice externe

Managementul consumului de putere

- Consumul poate fi imbunătățit prin
  - Optimizarea codului sursă
  - Oprirea perifericelor neutilizate
  - Adaptarea dinamică a frecvenței și a tensiunii de lucru
  - Intrarea în starea de low power mode când nu este necesar ca dispozitivul să-și execute funcțiunea principală

Analiza securității

Motivație

- Până de curând nu s-a luat în considerare prae serios subiectul securității sistemelor embedded
- Sunt raportate din ce în ce mai multe atacuri asupra sistemelor embedded din diverse domenii
- Atacurile sunt făcute posibile datorită lipsei mecanismelor de securitate

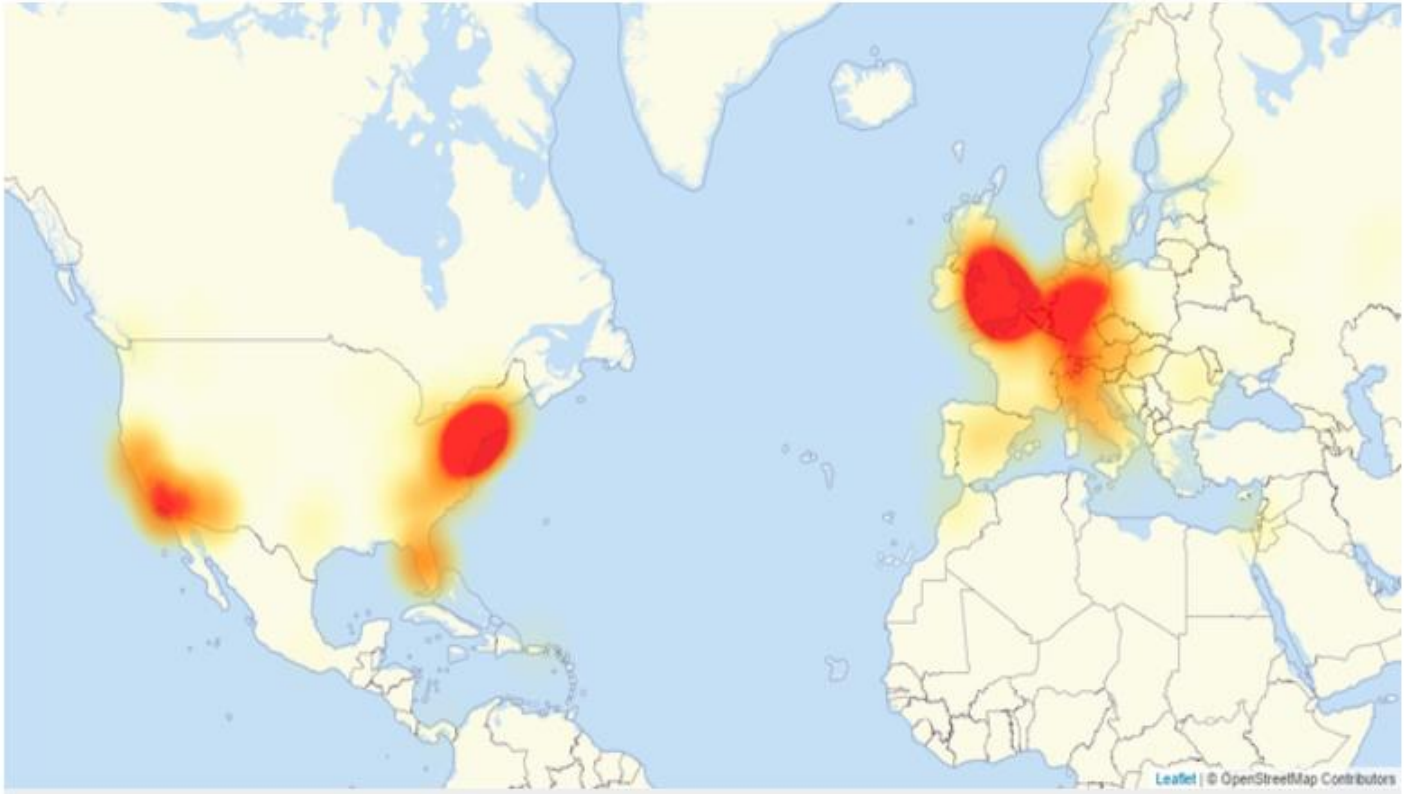
Atacuri:
Dyn attack: Oct. 21, 2016

Botnet realizat din dispozitive IoT compromise (imprimante, camere IP, baby monitor)

Infectate cu malware-ul Mirai

Puterea atacului: 1,2 terabit/s

Major internet services including Spotify, Twitter, Paypal, Reddit, the PlayStation Network, Netflix, SoundCloud and a number of media websites were difficult or impossible to reach early Friday.



A live outage map for the payments website PayPal shows widespread disruptions as of 12:45 p.m. ET on Friday, Oct. 21, 2016, following an attack on Internet infrastructure company Dyn.

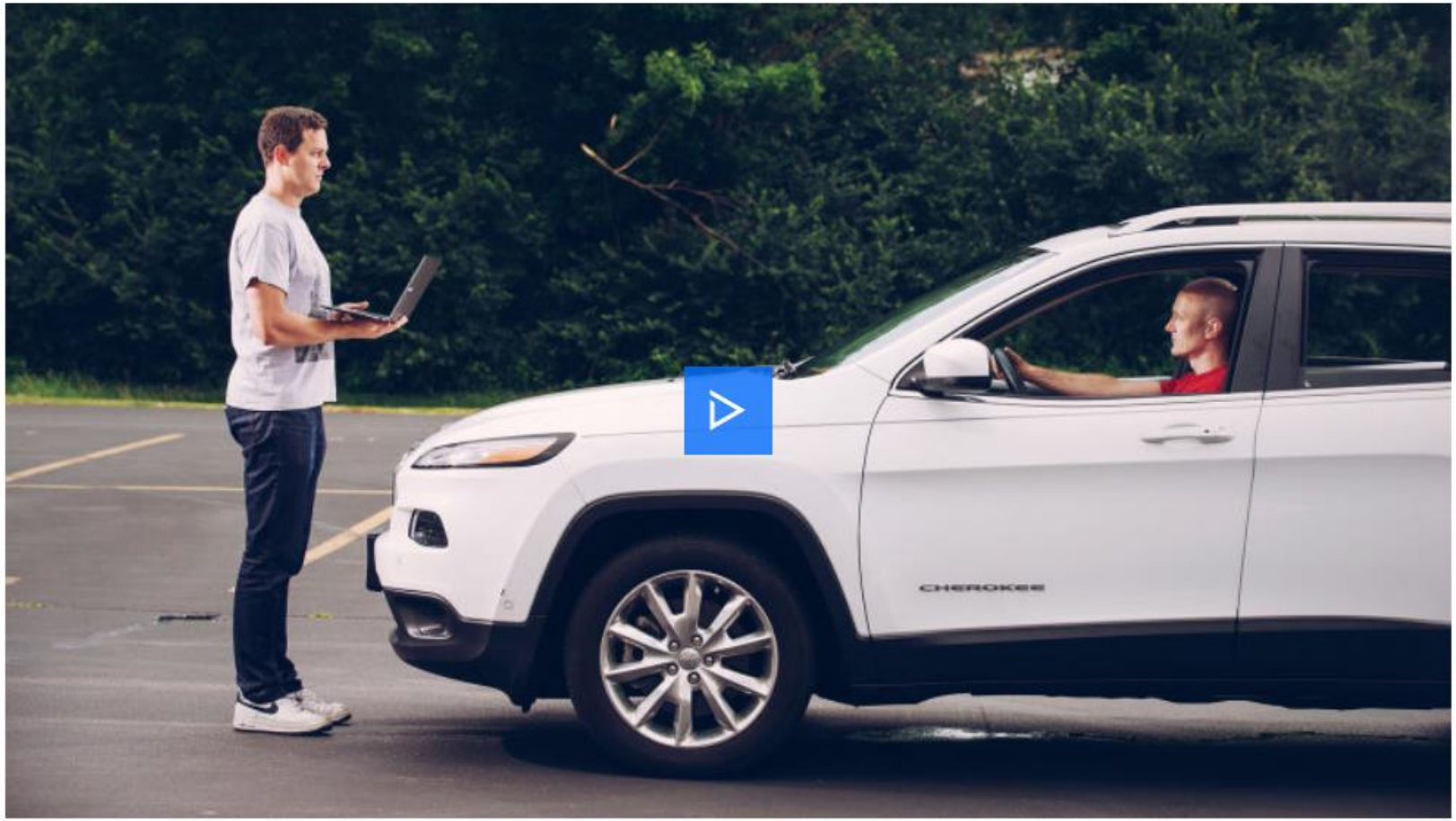
Photo credit: DownDetector.com

Atacuri: Car hacking

Automobilele nu prezintă mecanisme de securitate pentru majoritatea funcționalităților pentru că au fost considerate sisteme izolate ferite de atacuri

ANDY GREENBERG SECURITY 07.21.15 8:00 AM

HACKERS REMOTELY KILL A JEEP ON THE HIGHWAY—WITH ME IN IT



Atacuri: Stuxnet 2010

Worm identificat în 2010 care țintea sisteme industriale, mai precis un anumit tip de dispozitive PLC care sunt folosite printre altele la controlul centrifugelor pentru separarea materialelor nucleare radioactive.

The Stuxnet Attack On Iran's Nuclear Plant Was 'Far More Dangerous' Than Previously Thought

Michael B Kelley
Nov. 20, 2013, 12:58 PM 112,209

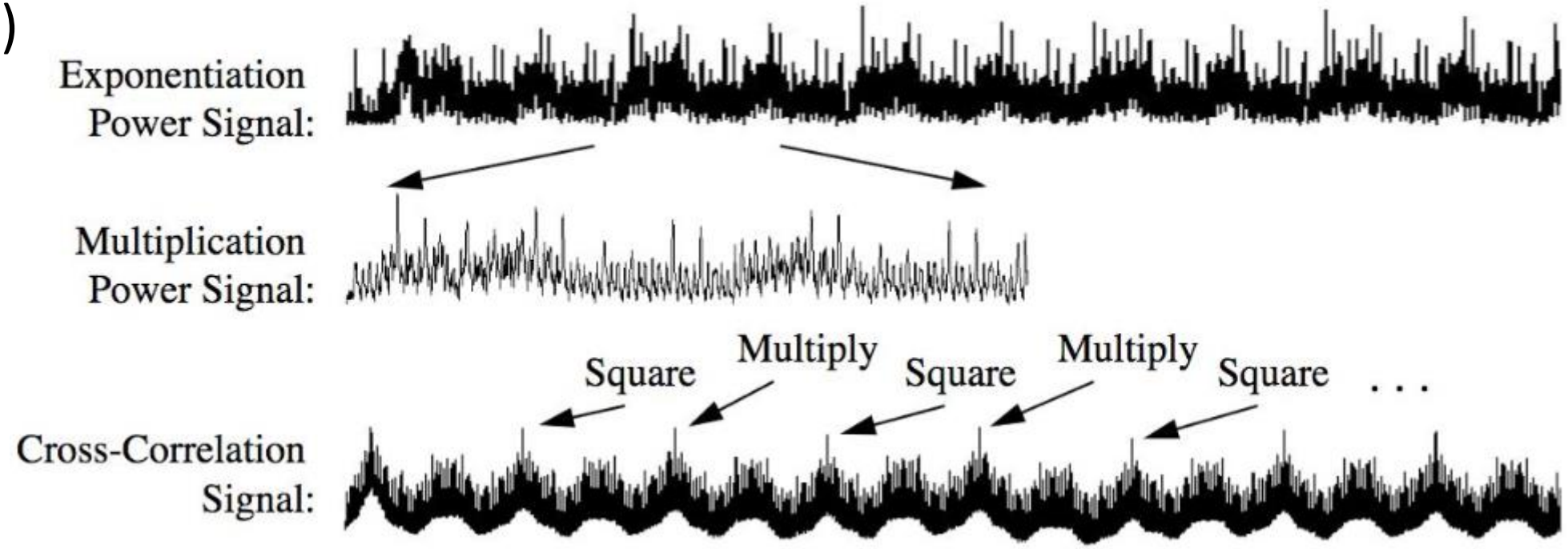
FACEBOOK LINKEDIN TWITTER EMAIL PRINT

The Stuxnet virus that ravaged Iran's Natanz nuclear facility "was far more dangerous than the cyberweapon that is now lodged in the public's imagination," cyber security expert Ralph Langer writes in Foreign Policy.

Stuxnet, a joint U.S.-Israel project, is known for reportedly destroying roughly a fifth of Iran's nuclear centrifuges by causing

Atacuri side-channel

Extragerea de informații private prin utilizarea caracteristicilor fizice ale unui sistem (timp de execuție, consum de putere, camp electromagnetic, sunet)



[T. Messerges et al. CHES, 1999.]

Măsuri de protecție

- **Comunicare autentificată** – asigură identitatea participanților la comunicare
- **Criptarea informațiilor** – informațiile sunt păstrate sau transmise în format criptat
- **Prevenirea scurgerilor de informații side-channel**

Materie pentru examen

- Slide-uri curs
- Edward A. Lee, Sanjit A. Seshia, INTRODUCTION TO EMBEDDED SYSTEMS A CYBER-PHYSICAL SYSTEMS APPROACH, 2017
  - Capitol 1 Introduction
  - Capitol 2 Continuous Dynamics
  - Capitol 3 Discrete Dynamics
  - Capitol 8 Embedded Processors
  - Capitol 9 Memory Architectures
  - Capitol 10.1 Input and Output - I/O Hardware