Intrusion Detection System

Introduction

- IDS monitors network traffic or system logs to detect suspicious activities.
- It alerts system administrator when potential hostile traffic is detected.
- Components of an IDS
  - Sensor
  - Console
  - Detection Engine

HIDS

- A HIDS monitors
  - the state of a system
  - stored information
  - Process tree
  - Size of certain vital processes
  - System Logs

and check that the contents of these appear as expected.

Example → OSSEC

NIDS

- Detects malicious activity such as Denial Of Service attacks, port-scans by monitoring network traffic.
- Reads incoming packets and tries to find suspicious patterns.
- Passive system → Doesn't try to mitigate the attack
- NIDS →Network Traffic
- HIDS →System logs

Example →Snort

IDS vs. IPS

- IDS - Does not try to mitigate the attack, just alerts the administrator.
- IPS - takes preventive measures.
  - E.g. break the connection, reprogram the firewall
- Snort running in inline mode - IPS.

IDS Techniques

- Anomaly based Detection
  - Designed to detect abnormal behavior in the system
    - Baselines the normal usage pattern
    - Alerts when usage deviates from the normal behavior
    - Example if a user logs on and off 20 times a day while the normal behavior is 1-2 times

IDS Techniques

- Signature based Detection
  - Uses specifically known patterns to detect malicious code
  - These specific patterns are called signatures.
  - Identifying the worms in the network is an example of NIDS based signature detection

Topics Covered

- Signature Based Detection v/s Anomaly Based Detection
- Signature Based Detection
  - Polymorphic Worm Detection
    - Control Flow Graph Approach
    - Enhancing CFG approach by using graph coloring
- Anomaly Based Detection
  - Data Mining Based Approaches
    - Supervised Learning (Association Rule Mining)
    - Unsupervised Learning (Clustering)
- Real Time Data Mining Based IDS
  - Cost Sensitive Modeling
- Snort Architecture
- Port Scanning
  - Various techniques of Port Scanning
  - Detecting Port Scanning – A probabilistic approach

Agenda

- Signature Based Detection
  - Polymorphic Worms
    - Detection using CFG
    - Enhancing using Graph colouring
- Anomaly Based Detection
  - Data mining approach
    - Supervised Learning

Detection of Polymorphic Worms

Virus vs. Worm

- Virus
  - Cannot replicate
  - Needs an agent to propagate

- Worms
  - Can replicate on its own
  - Works in three phases: Scan → Compromise → Replicate

Worms Vs. Polymorphic worms

- Worms
  - A unique representation
    - Same bit string across all instances.
  - Can be detected using signature based techniques.

- Polymorphic Worms(PW)
  - Change their representation before spreading.
  - Can’t be detected using signature based techniques.

Achieving Polymorphism

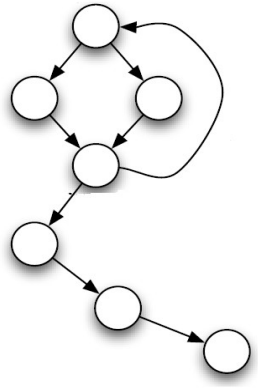
- Encryption
  - Encrypt the code of the worm with a random key before spreading
- Code substitution
  - Substitute the instructions with other semantically equivalent instructions
    - E..g. Multiplication → N times addition

Parts of a Polymorphic Worm(PW)

- Body/Code of the worm
- Polymorphic Engine (PE)
- Polymorphic Decryptor (PD)

Observation: A worm always has some executable part.

PW Detection – CFG construction



- Move A 10
- Move B 10
- ADD B
- JMP BLOCK2
- MOV A 15
- MOV B 20
- MUL B

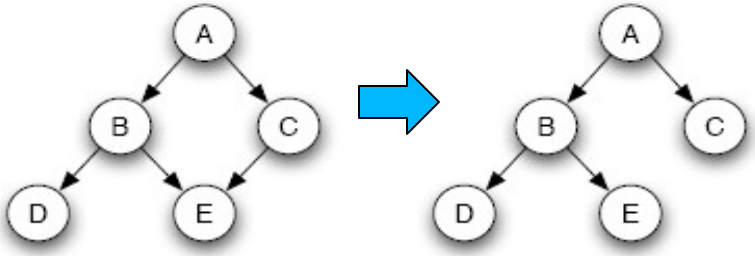
- linear disassembly of the byte stream
- Nodes → Describes the sequence of instruction without any jumps.
- Edges → jump instruction making transition from one node to another.

CFG of a binary code → cluster of closely connected nodes

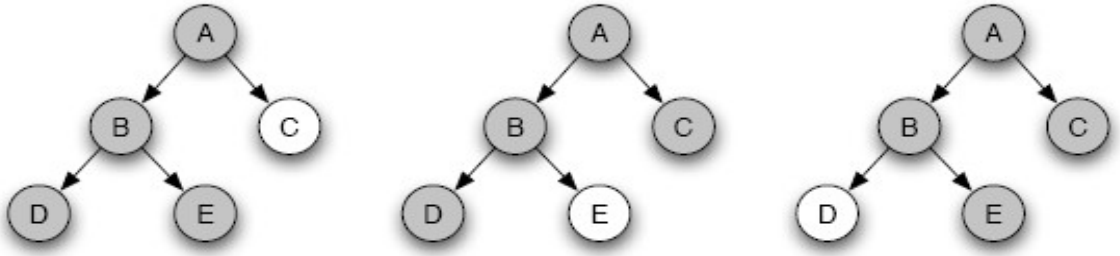
CFG of random sequence → isolated nodes

PW Detection – Sub Graph Generation

Generate k node connected sub graphs



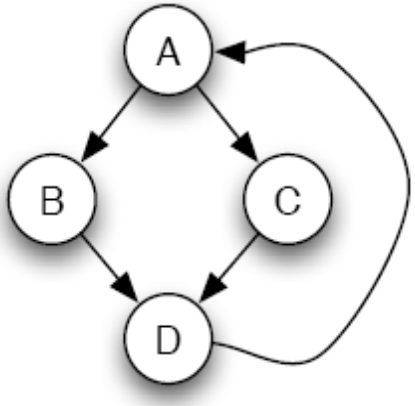
Control flow graph Spanning tree



4-node subtrees

PW Detection – Signature Extraction

Covert the k-node sub-graph into its canonical form



4-node subgraph

	A	B	C	D
A	0	1	1	0
B	0	0	0	1
C	0	0	0	1
D	1	0	0	0

Adjacency matrix

→ 0110 0001 0001 1000

4^2-bit fingerprint

PW Detection

- If same signature is observed in many packets flowing across different sources and destinations → Worm

Graph Colouring

- Classify Instructions → 14 sets

Class	Description	Class	Description
Data Transfer	`mov` instructions	String	x86 string operations
Arithmetic	incl. shift and rotate	Flags	access of x86 flag register
Logic	incl. bit/byte operations	LEA	load effective address
Test	test and compare	Float	floating point operations
Stack	push and pop	Syscall	interrupt and system call
Branch	conditional control flow	Jump	unconditional control flow
Call	function invocation	Halt	stop instruction execution

Graph Colouring

- A 14 bit colour value → associated with each node (1 bit corresponding to 1 class)
- When one or more instructions of certain class appears in the basic block , the corresponding bit of the basic block colour value is set to 1.
  - E.g. MOV A, B 00000000000010
    MUL A,10 00000000000001
    PUSH A 00000000010000

    Node Colour : 00000000010011

Continued…

- Append 14 bit colour value to each node in the adjacency matrix of the sub graph

- Concatenate the rows as before and get the new fingerprint

Data Mining Approach for IDS

Need for Data Mining Techniques

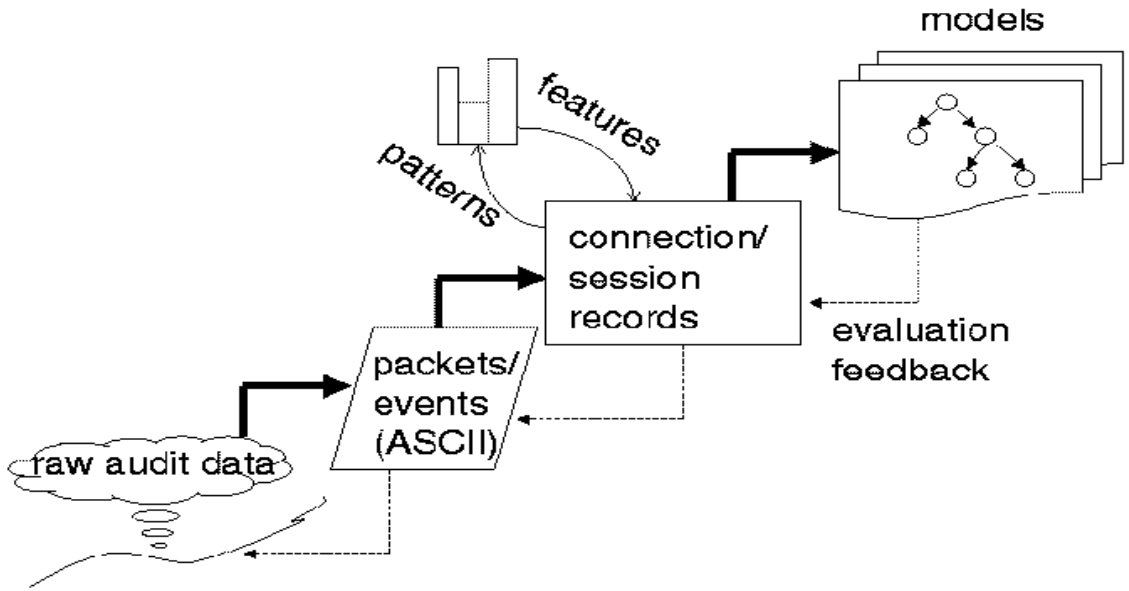
- Building an effective IDS is an enormous knowledge engineering task
- Rely on intuition and experience
- Hand-coded rules
- Limited extensibility and scalability
- A more systematic and automated approach is needed

Central Theme

- Apply Data Mining techniques to extensively gathered audit data
- Accurately capture behavior of intrusions and normal activities
- More effective as models are compute and validated using large amount of audit data
- Extensible

MADAM ID framework

- **M**ining **A**udit **D**ata for **A**utomated **M**odels for **I**ntrusion **D**etection



References

- C. Kruegel, E. Kirda, D. Mutz, W. Robertson and G.Vigna. Polymorphic Worm Detection Using Structural Information of Executables
- S. Singh, C. Estan, G. Varghese and S. Savage. Automated worm fingerprinting
- http://www.berkeley.edu/news/media/releases/2003/02/04_worms.html

End