### CMPSC 443, Homework 5

1. (12 pts) Recall that the anomaly-based IDS example presented in the slides is based on file-use statistics. The expected file use percentages (the Hi values in the Table are periodically updated, which can be viewed as a moving average)
2. Why is it necessary to update the expected file use percentages? (3pts)
3. When we update the expected file use percentages, it creates a potential avenue of attack for Trudy. How and why is this the case? (3 pts)
4. Suppose that at the time interval following the results in the second update of the table in the slides, Alice’s file use statistics are given by A0=0.05, A1=0.25, A2=0.25, and A3=0.45. Is this normal for Alice? (3pts) Compute the updated values of H0 through H3. (3 pts)

>>>

a. User changes over time, so if these values do not change, you will soon get many

false alarms.

b. Trudy can simply “go slow" and eventually convince the IDS that her actions are

normal.

c. In this example, S =0:118, which exceeds the threshold of 0:1 given in the book.

Therefore, this would be considered abnormal.

Please refer to the slides to compute the updated values for H0 through H3.

1. (8pts) Explain the two types of IDS systems by approaches and list two advantages of one against the other.

>>> signature detection and anomaly detection. See slides for a comparison.

1. (9 pts) Explain the same origin policy (3pts) and how it is exploited by cross-site scripting (XSS) attacks (6pts).

>>> The **same origin policy** permits scripts running on pages originating from the same site to access each other's methods and properties with no specific restrictions, but prevents access to most methods and properties across pages on different sites.

>>> Problem usually occurs when sites don’t sanitize user input to strip HTML

* Example: chat room (or MySpace or blog sites) that let users enter comments
* The “comments” can include JavaScript code
* This JavaScript code can transmit the user’s authentication cookies to some other site

based on the same origin policy, the javascript code injected by an attacker to a website can access the cookie placed by that website

1. (8 pts) Explain what is SQL Injection attack and why it can happen? Give an example.

* SQL injection is a code injection technique where an attacker places malicious code in SQL statements, via web page input. The consequence is that one’s database could be destroyed. SQL injection is one of the most common web hacking techniques.
* The root cause is lack of input sanitization.
* An example:
  + normal SQL query code:
    - statement = "SELECT \* FROM users WHERE name ='" + userName + "';“
  + If user name is input as ' or '1'='1
  + The SQL query executed by the database server becomes

SELECT \* FROM users WHERE name = '' OR '1'='1';

Which returns all records.

1. (10pts) Firewall concepts.
2. List the tree types of firewalls and describe their differences (e.g., locations in network stack, content they look at)
3. Explain why stateful packet filter can prevent TCP ACK scan attack whereas packet filter cannot.

>>> (1) packet filter – network layer, inspect IP address, port number

stateful packet filter – transport layer, remember TCP connections and flag bits

application proxy - application layer, check application content (i.e., data payload)

>>> (2)

TCP ACK scan attack scans for open ports thru firewall. It works by sending TCP ACK packets to destinations with different port numbers to find which port is open. Without prior 3-way handshake, it violates TCP/IP protocol, so the recipient responds with a RST packet if the port is open. Receiving a RST packet is an indication of the open port.

packet filter – do not keep track of connections, so RST packet will be returned to the attacker. But stateful packet filter keeps track of connections, so it will drop such RST packets; as a result, the attacker won’t be able to receive RST packets and cannot know if the port is open.

1. (15pts) Understand firewall rules. SMTP (Simple Mail Transfer Protocol) is the standard protocol for transferring mail between hosts over TCP. A TCP connection is set up between a user agent and a server program. The server listens on TCP port 25 for incoming connection requests. The user end of the connection is on a TCP port number above 1023. Suppose you wish to build a packet filter rule set allowing inbound and outbound SMTP traffic. You generate the following rule set:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rule** | **Direction** | **Src Addr** | **Dest Addr** | **Protocol** | **Dest Port** | **Action** |
| **A** | In | External | Internal | TCP | 25 | Permit |
| **B** | Out | Internal | External | TCP | >1023 | Permit |
| **C** | Out | Internal | External | TCP | 25 | Permit |
| **D** | In | External | Internal | TCP | >1023 | Permit |
| **E** | Either | Any | Any | Any | Any | Deny |

1. Describe the effect of each rule.

**>>>a.** Rules A and B allow inbound SMTP connections (incoming email)

Rules C and D allow outbound SMTP connections (outgoing email)

Rule E is the default rule that applies if the other rules do not apply.

1. Your host in this example has IP address 172.16.1.1. Someone tries to send e-mail from a remote host with IP address 192.168.3.4. If successful, this generates an SMTP dialogue between the remote user and the SMTP server on your host consisting of SMTP commands and mail. Additionally, assume that a user on your host tries to send e-mail to the SMTP server on the remote system. Four typical packets for this scenario are shown:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Packet** | **Direction** | **Src Addr** | **Dest Addr** | **Protocol** | **Dest Port** | **Action** |
| **1** | In | 192.168.3.4 | 172.16.1.1 | TCP | 25 | ? |
| **2** | Out | 172.16.1.1 | 192.168.3.4 | TCP | 1234 | ? |
| **3** | Out | 172.16.1.1 | 192.168.3.4 | TCP | 25 | ? |
| **4** | In | 192.168.3.4 | 172.16.1.1 | TCP | 1357 | ? |

Indicate which packets are permitted or denied and which rule is used in each case.

**>>> b.** Packet 1: Permit (A); Packet 2: Permit (B): Packet 3: Permit (C)

Packet 4: Permit (D)

1. Someone from the outside world (10.1.2.3) attempts to open a connection from port 5150 on a remote host to the Web proxy server on port 8080 on one of your local hosts (172.16.3.4) in order to carry out an attack. Typical packets are as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Packet** | **Direction** | **Src Addr** | **Dest Addr** | **Protocol** | **Dest Port** | **Action** |
| **5** | In | 10.1.2.3 | 172.16.3.4 | TCP | 8080 | ? |
| **6** | Out | 172.16.3.4 | 10.1.2.3 | TCP | 5150 | ? |

Will the attack succeed? Give details.

**>>>c.** The attack could succeed because in the original filter set, rules B and D allow all connections where both ends are using ports above 1023.

1. (6pts) 10.36

a. It's a very bad idea. In WEP, the keystream is repeated each time the IV repeats

(assuming that the long-term key K doesn't change). However, this modi\_ed

version is even worse, since each packet is encrypted with the same keystream|

this is at least as bad as using a one-time pad over and over and over and over

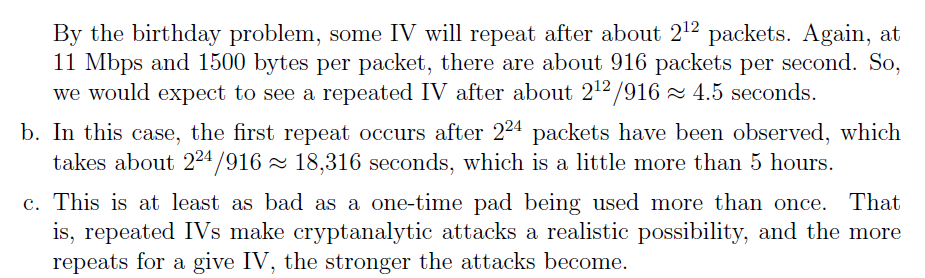
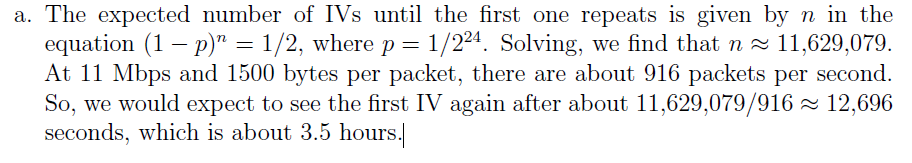
and …

b. It would be worse. While the way that the key K and IV are combined in WEP

leads to a fairly simple cryptanalytic attack, this attack would be even easier.

1. (6pts) 10.37 (a, b, c)

This is similar to the birthday problem in Chapter 5.3.



1. (8pts) Suppose an attacker wants to secretly send out the bits “11000111” through a file lock covert channel. Explain how it works.

>>> Let lock and unlock stand for 1 and 0, respectively. The Trojan program needs to lock, lock, lock, unlock, unlock, unlock, lock, lock a file that is known to the colluding, time-synchronized, spy program, which checks the locking status of the same file. (if you start with the leftmost bit, that is also fine).

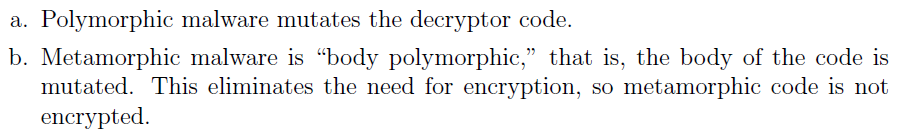
1. (6pts) What is a rootkit? Please list at least four types of rootkits .

>>>It is software used after system compromise to:

* + Hide the attacker’s presence
  + Provide backdoors for easy reentry

>>> Application-level, library-level, kernel-level, under-kernel rootkits.

1. (6pts) 11.22.



1. (6pts) The following code fragments show a sequence of virus instructions and a metamorphic version of the virus. Describe the effect produced by the metamorphic code.

|  |  |
| --- | --- |
| Original code | Metamorphic Code |
| Mov eax, 5  Add eax, ebx  Call [eax] | Mov eax, 5  Push ecx  Pop ecx  Add eax, ebx  Swap eax, ebx  Swap ebx, eax  Call [eax]  nop |

>>> This metamorphic version could destroy the signature derived from the original code by adding useless instructions.