Ayush Jain (UNI : aj2672)
COMS W4180 Network Security
Written Assignment 3

1 Problem 1

a. The top commands gives us a summary of the current processes running in the system along with their PIDs, %Memory, etc whereas the netstat command tells the port being occupied in your system by different processes and network connections. It also tells you the pid of the local processes occupying the ports.

Since, the system always has a set number of processes, the result of top command and netstat should stay fixed. Also, since their is only one user operating the system, any malicious process has to enter the system from the internet.

So, if use the netstat and top commands to collect data on the various pids, their network usage their memory consumption and their internet usage, we should be able to figure out any suspicious activity happening on the system. The data collection will have to happen every few minutes depending on the expected packet downloads and memory usage. The time frame will have to be enough to catch any suspicious activity in the sample of the time frame.

In the approach above, there could be false positives and false negatives. It is possible that a known process can run an internal sub-process which is not expected leading to a false positive, or processes like system updates which can lead to network usage but are fairly safe. Such processes can lead to false positives. Similarly, a hacker could also hide the malicious process under the mask of a regular process and suspected network usage. Such a process can fool the IDS based on only top and netstat commands and can lead to false negatives.

b. If the developer of the software is given access to the system, he or she can run new proceses that were not usually expected by the system. Because of this the IDS will have to include scope for some errors and unexpected activity to happen. This will reduce the accuracy of the IDS as more scope for errors gives more space for hackers to run suspicious activity without being caught.

2 Problem 2

The following security issues were identified in zigbee.

- If a nonpreconfigured device joins a network, a single key may be sent unprotected and enable encrypted communication. This one-time transmission of the unprotected key results in a short timeframe of exploitability in which the key could be sniffed by an attacker.
- As every ZLL device joining to a ZLL network shall use per definition the ZLL master key to derive the active network key, knowledge of the ZLL master key allows an attacker to intercept the key-exchange and acquire the current active network key. This would then allow the attacker to control all devices in the ZigBee network.
- ZLL devices also support a feature called Touchlink Commissioning that allows devices to be paired with controllers. As the default and publicly known TC link key is used, devices can be stolen. Tests showed that amateur radio hardware using normal dipole (Rasperry Pi extension board) antennas already allowed Touchlink Commission from several meters away whereas for security reasons this should only work in close proximity.
- The tested home automation system is not capable of resetting or changing the applied network key, so even if a user notices unwanted behaviour in the network, there would be absolutely no possibility of locking the intruder out. Also no automatic key rotation could be identified during a timeframe of eleven month.

- The smart lighting solution is also vulnerable to a device takeover from any external party. It was possible to steal light bulbs and join them to a fake network without knowledge of the active secret keys. An attacker just has to send a reset to factory default command to the light bulb and wait for the bulb to search for ZigBee networks to join. The bulb will connect to the first network available without any further interaction of a user.

3 Problem 3

The authors are able to identify the length of underlying plaintexts and the number of plaintexts that are encrypted.

None of the Countermeasures dicussed by the authors are completely able to prevent the identification. The Countermeasures discussed are brodly classified intot three categories:

1. Type-1: SSH/TLS/IPSec-Motivated Countermeasures
   (a) Session Random 255 padding: Performs well against P classifiers but performance against LL and H decreases sharply at high universal size.
   (b) Packet Random 255 padding: Performs better than session random in almost all scenarios. However, the performance trend against classifiers is similar.
2. Type-2: Other Padding-based Countermeasures
   (a) Linear padding: Performs best across all countermeasures.
   (b) Exponential padding: A little worse in performance than linear padding.
   (c) Mice-Elephants padding: Performs very badly against H classifiers at high universal sizes.
   (d) Pad to MTU: Performs badly against LL and H classifiers at high universal sizes, but not against P classifiers. Common intuition about the Pad to MTU countermeasure is that it ought to work well against TA attacks since it ensures that no individual packet length information is leaked. However, as we seen in the second row of Figure 6, we see this intuition is wrong in large part because the number of packets is still leaked.
   (e) Packet Random MTU padding: Similar to Pad to MTU.
3. Type-3: Distribution-based Countermeasures
   (a) Direct target sampling: Performs badly against LL and H but not against P classifiers.
   (b) Traffic morphing: Similar to Direct target sampling.

In general, Type-1 and Type-2 are expected to obfuscate bandwidth usage. They do, but these per-packet paddings only add noise to the low order bits of total bandwidth. Specifically, the change to bandwidth usage is too small relative to what would be needed to make two websites bandwidths to overlap significantly and hence they perform poorly with Bandwidth based classifiers. Type-3 countermeasures perform better against Bandwidth based classifiers.

4 Problem 4

a. TCP

b. arrival time, packet size, direction

c. SMTP-in (25), HTTP (80), HTTP over SSL (443), FTP (20), SSH (22), Telnet (23), as well as outbound SMTP and AOL Instant Messenger traffic.

5 Problem 5

Following are the examples of IOT devices hacked in the previous years.

1. One of such devices were baby monitors. The weaknesses in the monitors made it possible to include monitoring live video feeds, changing camera settings, harvesting video clips stored online, and adding to the list of users who are authorized to remotely view and control a monitor. Following were the list of weaknesses in a Philips baby monitor.

   The Philips In.Sight B120 establishes a direct connection to the camera's backend web application onto the public Internet, unencrypted and unauthenticated. By brute forcing the possible hostname and port number combinations an attacker can locate an exposed camera and is able to watch the live stream, enable remote access (e.g. Telnet), or change the camera settings.

2. Another device to be hacked was the pacemakers. The effects of the security vulnerabilities were that pacemakers could be hacked or switched off and firmware could rewritten on these devices. Following were the list of security loopholes found in the devices.
   (a) The pacemakers contained a secret function which could be used to activate all pacemakers and implantable cardioverter-defibrillators (ICDs) in a 30 foot -plus vicinity.
   (b) The usernames and passwords to the manufacturers development server were written in the clear inside a function in the code.