Behavioral Malware Detection Approaches for Android

Mehedee Zaman, Mohammad Rakib Amin, Tazrian Siddiqui

DETECTION METHODS

STATIC

- Constructs a **Control Flow Graph** (CFG) of the malicious binary executable file.
- Searches for malware signature in the CFG.
- Might need deobsfucation of executable.
- Can be executed without running the executable under analysis.

SIGNATURE
- Portion of the CFG, identical to that of a known malware.

BEHAVIORAL

- Lets the malicious application run in a sandboxed environment.
- Traces various behaviors of the application at runtime.
- Behavior is matched against a database of known malware behavioral signature.

SIGNATURE
- Network traffic with a known malicious domain.
- Part of **Systrace** log, similar to that of a known malware., etc.

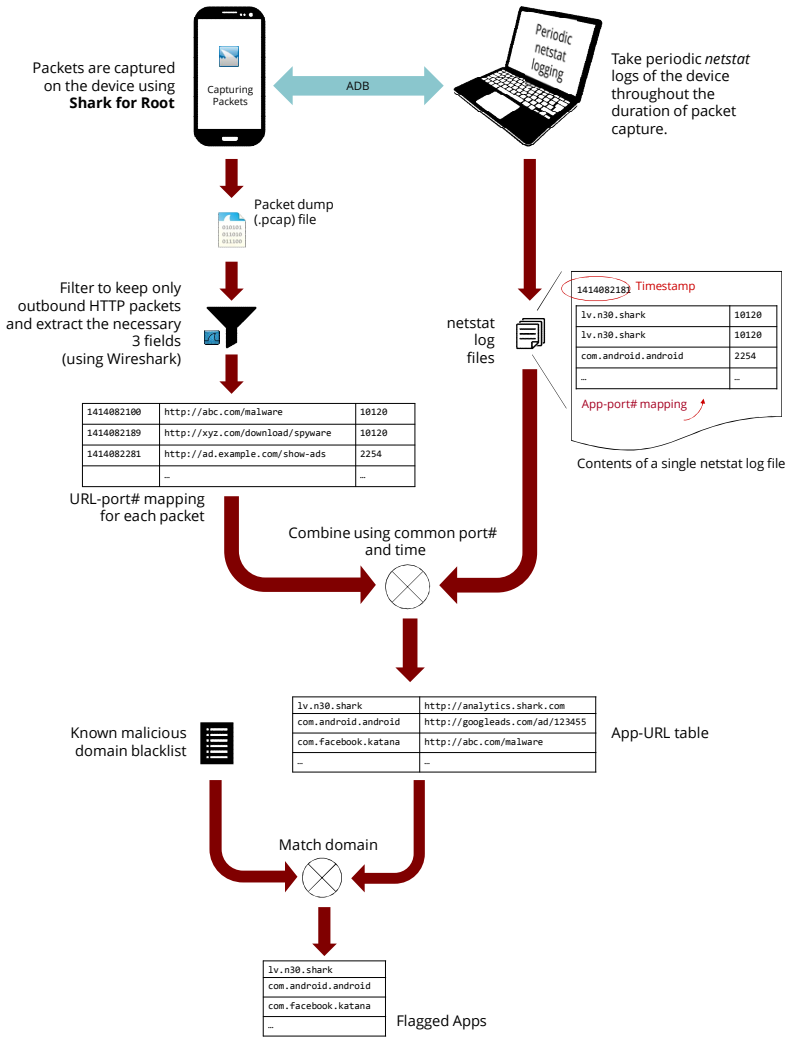
OUR ASSUMPTIONS

- Malware will communicate over HTTP only.
- Malware C&C servers use domains which are already listed as malicious domains in our blacklist.

FUTURE PLANS

- Develop a single application to all the work on the device.
- Collect system call logs and extract known malware signatures.
- Use machine learning to train a classifier which could detect malicious system call sequences.
- Use these approaches together, which will enhance our domain blacklist as well as the syscall based classifier.

DETECTION STRATEGY



ALGORITHM FOR MAPPING APPLICATIONS TO PACKETS USING PORT# AND TIMESTAMP

Let t be the timestamp of a packet. Let $t_1, t_2, t_3, ..., t_{100}$ are the timestamps of the *netstat* outputs (they are stored in corresponding netstat log files).

Of course $t_1 < t_2 < t_3 < ... < t_{100}$.

If $t < t_1$ or $t > t_{100}$, we discard the packet. We only consider packets with t such that
$t_1 \leq t \leq t_{100}$.

Now for each of these packets, there is an i such that $t_i \leq t$ and $t_{i+1} > t$.

We assign a packet to an application using the rules shown in the table.

Case 5 indicates that after t_i, some application opened the port, sent some packet(s) and then released the port before t_{i+1}. So this packet has gone untraced. We can lessen the frequency of such occurrences by decreasing the interval between t_i and t_{i+1}.

Case	Application using the port at t_i	Application using the port at t_{i+1}	Decision
1	A	A	A sent the packet
2	A	none	A sent the packet
3	none	A	A sent the packet
4	A	B	If t is closer to t_i → A sent the packet If t is closer to t_{i+1} → B sent the packet
5	none	none	Discard the packet

Department of Computer Science and Engineering (CSE), BUET