DOS attack mitigation simplified ahah

Mikhail Lyamets @fenchelfen BS17-7

Actors

Alice – innocent Android device, desperately wants to read Macbeth

Bill – Linux machine, stores complete Shakespeare's works

Dozer – malicious junky who's gonna take Bill down

Scenario

Dozer sends a manifold of SYN requests to **Bill**; he mitigates this by adding **Dozer** to the blacklist after the third SYN request and dropping all its threads after the timeout expires. During the attack **Alice** asks **Bill** to give her some **Macbeth**.

Goal: let **Bill** send **Alice** the file despite the attack set up by **Dozer**.

Result: SUCCESS. **Dozer**'s attack doesn't cause any denial of service, **Alice** (or any other peer) can acquire any file that **Bill** possesses almost instantly.

Implementation details

To track the number of connections by a host I use the hashmap. I adopted it for storing integers as values. Blacklist is a hashmap as well.

Explanation and screenshots

IMGUR LINK with a TIMESTAMP

The tiniest window runs SSH and is called **ff@stranberg**; it's connected (through SSH) to my Android device running **Termux** – terminal emulator. Two others are called **ff@Bill** and **ff@Dozer** from left to right.

Pictures are placed in a chronological order. Initially, **Bill** is waiting for incoming connections (pic. 1). After that, **Dozer** comes to play and starts his attacks on **Bill**.

Bill, in turn, puts **Dozer** into the blacklist and drops the first 3 threads allocated for **Dozer** after they timeout (pic. 2).

Alice asks **Bill** for the file – success. **Alice** then checks if it's actually downloaded by executing **cat macbeth** in her shell.

P.S **Dozer** sends SYN messages to **Bill** with an interval of 0.1 sec. The delay can be made shorter but it makes **Dozer** crash too soon due to the limit on the number of file descriptors imposed by the system.