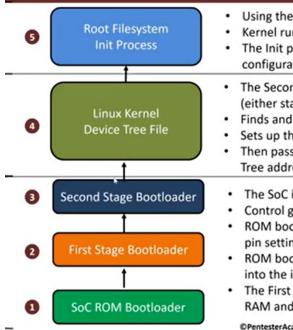
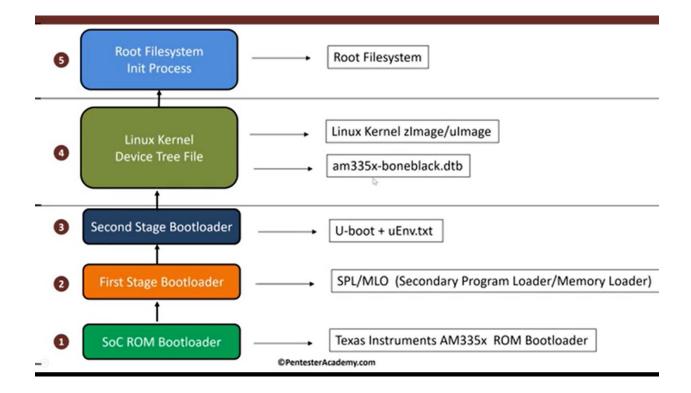
Embedded Linux Boot Process

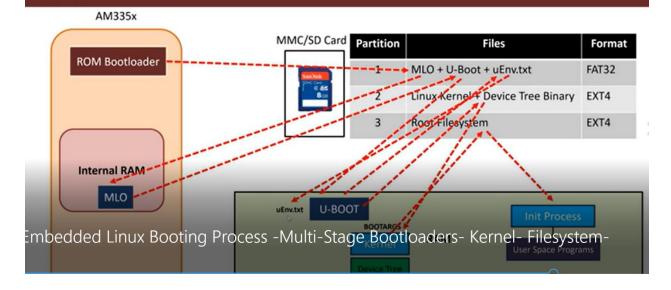


- Using the Boot Args, the Kernel locates and mounts the root filesystem
- Kernel runs the Init process (PID 0) to start userspace
- The Init process now spawns other userspace processes based on its configuration files
- The Second Stage Bootloader reads its configuration settings (either statically embedded or external file)
- · Finds and loads the Linux Kernel and the Device Tree Binary into RAM
- Sets up the Kernel Boot Arguments
- Then passes control to the Kernel which uses the Boot Args and Device
 Tree address to initializes itself and hardware devices
- The SoC is powered and begins execution at the reset vector
- · Control given to the ROM bootloader
- ROM bootloader decides boot device order based on hardware pin settings
- ROM bootloader loads the First Stage Bootloader from the boot device into the internal SoC memory and passes control to it
- The First Stage Bootloader copies the Second Stage Bootloader into RAM and passes control to it

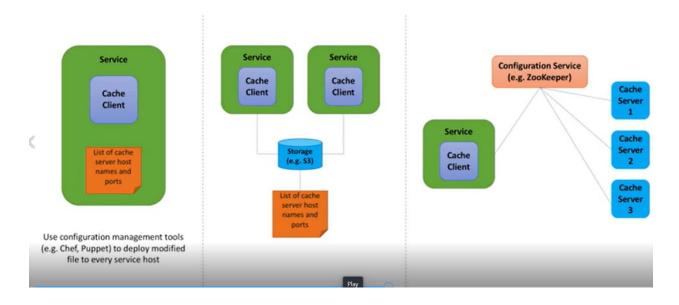
©PentesterAcademy.com



BeagleBone Black Boot Process



Maintaining a list of cache servers



```
Solution {
public boolean canFinish(int numCourses, int[][] prerequisites) 🞚
   int[] indegree = mew int[numCourses];
   for (int[] pre : prerequisites) {
       indegree[pre[0]]++;
   Set<Integer> zeroDegree = new HashSet();
       (int i = 0; i < numCourses; i++) {
          (indegree[i] == 0) {
            zeroDegree.add(i);
      (zeroDegree.isEmpty()) {
   }
while (!zeroDegree.isEmpty()) {
       Iterator<Integer> it = zeroDegree.iterator();
       int course = it.next();
       zeroDegree.remove(course);
        for (int[] pre : prerequisites) {
               (course == pre[1]) {
                indegree[pre[0]]--;
if (indegree[pre[0]] == 0) {
                    zeroDegree.add(pre[0]);
            }
   for (int num : indegree) {
       if (num != 0) {
```

