Some Fit and Finish items for Eve for smooth operation of Edge Devices

Considering Fit-n-Finish is the priority as compared to new features, the following are some of the things that need to be improved:

1. Logging / Debuggability
   a. Linux state (Ip Table Rules, Disk contents, ifconfig -a, ip route etc)
      i. Need Kernel Information out to debug Broken hardware
   b. PubSub state
   c. Ability to run some pre-defined diagnostic commands
   d. Better visibility into each agent state
      i. Internal Agent state
      ii. Internal Agent counters
      iii. e.t.c
   a. App Instance Create / Delete / Refresh / Purge / Restart etc. For Example:
      i. Verifying?
      i. Auto Retriable Error
      ii. User intervention needed.
      i. App Instance Create RCVD (Name, time etc)
      ii. Downloading / Verifying Images
      iii. Download / Verifying images Done
      iv. Error
      v. Copying For RW image
      vi. Reserving Resources
      vii. Starting Instance
      viii. Instance Create Done
   b. Have a structured format to log such events as INFO events - which user can then look to know the actual details from the device
      i. These are also very useful from the developer perspective.
   a. Kernel state / Disk State / Temp etc..
   a. The main thrust is to improve the content of logging, to be able to debug 95% of the issues without reproducing the issue or accessing the device using console / ssh.
   b. Ability to prioritize messages based on severity level - CRIT / ERROR / INFO / DEBUG - in that order and drop the messages starting from lowest priority ones
   c. Proper Device Events - In INFO - Provide various events, AS SEEN FROM DEVICE, for each trigger:
   d. Kernel Coredump in case of kernel crashes..
   e. Information to debug broken hardware
2. Device Reboot reason
   a. First boot after install
   b. User Triggered Reboot - Time
   c. Upgrade - Time
   d. Upgrade Failure - Rollback
   e. Unexpected Reboot
      i. Agent Crash - Details
      ii. Agent Watchdog Timeout - Details
      iii. Hardware Watchdog - Details
      iv. Kernel Crash
      v. Power Failure
      vi. In these cases - log.Errorf() message details so that it goes into Kibana
      vii. UI - should just put this from user perspective - and hide the details of the crashes
   a. Current reboot reason is more for the developer.
   b. For the user - Reboot reason should be seen as follows:
3. Device Events
   a. These don't always correlate with what is going through the system, especially after reboot
   b. We see events toggling between Init / unknown / init / downloading etc.
   c. These fixes are more of Bug Fixes
4. More visibility into state of Bigger triggers
   a. Include more granular information, like Reboot Started etc.
      i. Currently, there is no visibility on when the device received the reboot, when it is done shutting down all app instances and when it is actually rebooting. This is all useful information to an admin waiting for systems to come up.
      ii. Some states are:
      iii. Shutdown applications
      iv. Starting Reboot (Still from Older image)
      v. Bootup (As soon as possible)
      vi. Or may be the existing msg is good enough
   b. Upgrade - We can provide more visibility to the user:
      i. Image Downloaded
      ii. Verification
      iii. Install
      iv. Shutting down applications
      v. Reboot
      vi. Booted up as part of Upgrade
      vii. Testing in progress (Update time remaining)
      viii. Testing Done / Upgrade Successful
   c. Upgrade FAILURE Fallback - cases:
      i. Shutdown applications
      ii. Reboot to Fallback image Started
      iii. Bootup of Fallback image

           iv. .. This part is common to bootup
       a. Reboot Device
5. Advertise capabilities to Cloud to enable smooth upgrades, to easily allow cloud deal with multiple versions of devices.
       a. For example - Cloud can send Encrypted secrets Vs. Clean text - Doesn't need to send both
       a. Are these used by Cloud to change behavior dynamically? Or just for Inventory analysis?
6. Eve images on Docker hub.
7. Fallback Interface configuration (Lower priority)