Iron Man - Feature #504
IoT security

28 Feb 2016 11:38 - Hammel

Status:	New	**Start date:**	28 Feb 2016
Priority:	Low	**Due date:**	
Assignee:	Hammel	**% Done:**	50%
Category:	Research	**Estimated time:**	0.00 hour
Target version:	002 - MVP		
Severity:	01 - Critical		

Description

I've been thinking about security with IoT devices. Any data we store about IoT devices is on the PiBox server. So we need to secure that data. We can encrypt it. We need a key for that. The key cannot be kept on the PiBox server. If we put it on our phone then someone could hack the phone to get to the server and enable access to the IoT data.

The trick may be to have the key on the phone but have the server required user input to enable it. So at power up the phone is required to connect to the PiBox server to enable IoT device use. Without user input, the request from the phone is denied and IoT devices are not enabled.

Seems pretty straight forward and is known as two-factor authentication.

The problem is: what happens on power failure? The rebooted device will come up waiting on a connection from the phone (which could be automated) but requires the user to be at the PiBox server to enable it. That doesn't work if you're on vacation.

So we need some variation of this. It's not required for proof-of-concept. But it's needed in the MVP.

History

#1 - 28 Feb 2016 11:39 - Hammel

- Description updated

#2 - 27 Jul 2017 21:37 - Hammel

- % Done changed from 0 to 10

This needs to be done using AES. See https://redmine.graphics-muse.org/projects/ironman/wiki/Software_Links

#3 - 13 Jul 2018 14:54 - Hammel

- Priority changed from Normal to Low

- % Done changed from 10 to 50

- Severity changed from 03 - Medium to 01 - Critical

On wire security will be handled with AES, which is now implemented in Jarvis <-> monitor communications and will be implemented shortly for monitor<->sensor communications.

Sensor security is hardware based, meaning you can't modify it's configuration without manually changing switches.

Monitor security will require a security audit to see how we can prevent access to the machine. I know there are some features that need to be outright disabled (like telnet) and some that should be disabled in production (like ssh). The web interfaces probably need auditing for holes too though due to their very simplistic nature and API I don't see many ways to get through it other than maybe URL overflows (which nodejs should be handling,

I think.

Lowering this because most security related issues are being tracked already, sans security testing of the web APIs.