Supply Chain Security and Compliance for Embedded Devices & IoT

Written by Martin Callinan – Director, Source Code Control Limited and,
David Phillips - BSI Panel Chair – IST15/09 – International Standards for IT Asset Management

Revise April 2017

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Introduction

The use of software powered embedded devices is poised for massive growth over the next few years as medical devices, automotive and consumer electronics industries all embrace the Internet of Things (IoT). Not only will the fields embracing IoT be diverse, but also the devices themselves will be diverse, from everyday computers and tablets to sensors, light switches, thermostats and the infrastructure supporting them.

The IoT industry will rely on software to run a small army of embedded devices. In order for technology companies to meet the demand and pace of development much of the software used will rely on open source technologies, with the final software assembled from an even deeper universe of IoT code libraries and web-based protocols accessing a mesh of fast-evolving resources.

The subject of this article: the scale of our dependency on buried computing assets – both the devices and the code we take for granted – will soon reach new dimensions. Can we keep control?

New Scope of Security Controls for New IoT Device Capabilities

The ever-upgrading platforms we know today will manifest in IoT tomorrow as an explosion of software and hardware, many with convoluted intellectual property sources. Each week chip builders announce smarter ways to architect ARM® and other cores, achieving another step change in embedded processor power available to IoT designs. Today’s tiny computer platforms can hold richer and more capable functions than we believe possible with negligible demands on power – they are so small, cheap and reliable, we can forget these assets are even there. Furthermore, just as micro-code in Intel® PC processors updates without our knowledge, the IoT code and functions I use expand silently over my lifetime, evolving as I renew domestic appliances and personal property.

This poses developers with colossal questions how to protect people, IP rights and data rights for us under in a chaotic ecosystem. More than ever, we need intelligent policies and good practices, so assets are identified and controlled throughout these dynamic infrastructures now proliferating in our homes and lives.

Developing from a Secure Foundation

When designing a device, it is important to engineer in security and quality controls from the ground up. For technology organisations there will be third party suppliers of hardware and software that will become part of the device shipping to end customers. It is not uncommon for organisations to trust manufacturers to deliver secure technology. However, with the fast pace of developments and the pressure to deliver solutions to market many organisations outsource parts of the manufacturing process to unverified partners and in doing so lose control of product and component assembly.
Agile Design with Multiple Interfaces Is More Vulnerable

Traditionally computing platforms have been engineered either for relatively closed systems for specific uses, or else for more general use but having network interfaces founded on closely specified hardware and software, tight protocols and with segmented networks enabling security services to operate effectively. We also tend to use them with well-known classes of software applications. Although problems with security vulnerabilities arise, the conditions under which they arise can be readily described (even by relative novices) and clear warnings issued, for example by:-

- Warning users of a specific kind of platform;
- Naming familiar applications or executable files installed, that may contain the vulnerability;
-Specifying a network, URL or open port by which threats attack and ultimately compromise the system.

The fact these are familiar scenarios, or one of a few major classes has worked in our favour to highlight and contain vulnerabilities of classical traditional client computer platform environments.

However, in the new era of agile device development, IoT devices are generally much more fragmented platform-types that don’t conform to labels we are familiar with. Their executables and applications are nowhere near as visible and often contain multiple communications interfaces – and these can easily be non-conformant, for example, the growing variety of pseudo-implementations of Bluetooth-like wireless data communications.

At the outset then, we should recognise that due to the diversity of IoT, without action we will all find describing and sharing the conditions that can lead to a vulnerability to be very much harder.

Grey Market and Counterfeit Embedded Devices

Taking embedded devices as an example there is a significant grey market for components or complete platforms such as a network device which would have an embedded processor with an operating system most likely based on Linux which can be booted and application’s code then loaded to control the device.

There are numerous risks associated with using grey market components and devices:

2. Traceability – markets such as defence and aerospace require traceability of components
3. Unable to meet requirements for standards such as ISO9001 SAE9120
4. Malware – what is in embedded code on the devices? What is the provenance of the code?

Governance Framework

Existing standards do offer many of the elements of governance to assist agile IoT developments. Some examples ready to come to the rescue are the policies, procedures and repeatable process (many of which can be automated) for identification of software found in the ISO/IEC 19770-series of standards.

It is now much easier to conquer one of the first barriers to effective data for managing complex software assets categories, thanks to public access to an internationally agreed vocabulary, key principles and rich definitions specifically created to help achieve this improvement in control.
Developer toolkits – even pro-suites – can be weak at managing software and libraries as proper assets, lacking well-formed definitions of key asset properties required for their proper management (standardised licence classifications, entitlements, usage metrics, platform/version IDs). However, many can be configured properly – if we apply these new definitions. It is then possible to achieve good baseline reports for controlling secure asset use, cut risk and spot unexpected non-compliance. Stakeholders using such reports also talk the same language: so where are the foundation principles for confidently managing our increasingly complex software and platform assets?

Help is now at hand in Part 5 of the ISO/IEC 19770-series - an internationally agreed industry standard providing clear and common vocabulary and a structured approach for effectively managing IT assets (both software and hardware) and for building trustworthy data for control and showing good governance. (Outside the embedded world this series is already popular with managers, auditors and practitioners seeking to improve CMDBs, CI records and reporting.)

- The new publically available Part 5 vocab is readily available here (by accepting ISO copyright): http://standards.iso.org/ittf/PubliclyAvailableStandards/c068291_ISOIEC_19770-5_2015.zip

This accessible standard offers vocabulary and more to assist in building trustworthy asset management for security, risk reduction, confident licensing or service negotiations and for cutting waste and cost.

As well as applying these improved definitions and processes, we also need responsible managers making up the chain of governance in IoT. Managers must require controls and conduct verifications of these practices within their part of the IoT supply chain. Organisations with such a Management System can benefit from stating their allegiance to a good Standard and by driving good governance from the top down, citing their conformance — starting with regular self-assessment.

In time, potentially more industry infrastructure may help these good practices to interwork effectively across the many different components making up our exploding IoT universe.

Supply Chain Governance
Organisations should look to adopt supply chain security. Every supplier of components (hardware, software or firmware) should have quality assurance over their source and credentials of suppliers.

A striking driver in the current market is the torrent of ‘Maker’ Kit variants. These offer anyone prototyping embedded controllers – including many quite suited to IoT applications - an increasing choice, ever cheaper as each new feature emerges (Bluetooth LE for example). Kick-starter teams now offer capable sub $9 computer platforms. These initiatives are wonderful for prototyping and a side effect is the aggressive cutting of supply costs of very powerful chips and interface components.

Yet there are a number of areas that can be increasingly overlooked when sourcing components:

1) At ground-level, smart choices begin with the Embedded/Electronics Hardware Designer. Careful choices of components build-in safe electronic components. Confidence in a platform begins at the lowest level - in the support chips – many of which themselves contain intelligent processors and bare-metal code. Increasingly, embedded designers rely on modules which can include
- Network modules
- Wireless modules
- Single-wire / Near Field Communications modules
- Co-Processors and FPGAs
2) **Firmware engineers** also choose the embedded processor architecture for IoT devices, and so firmware professionals also have a key role in controlling sourcing of:

- Main processor
- Security feature sets (such as secure boot capability)
- Embedded Architectures and their associated reference designs.

The latter often come with technology guarantees which can be a good sign of assurance. For example, more and more processor architectures experience end of life each year, whereas others such as Freescale (now NXP) offer certain architectures with come with many years of guarantee and designers can use this to judge whether they will offer greater confidence of support as reliable IoT solutions.

3) **Supply Chain Officers** – purchasing staff will need to aware of the risks of these new classes of module, system-on-chip and powerful component so they can apply rules and corresponding procedures for appropriate sourcing controls and batch identifications.

4) **Production/Inventory Controllers** – will need to implement controlled MRP systems that make use of batch data through manufacturing, support traceability and retain identities.

5) **Inspection and Testing Offices** – will need to apply centralised data to their test procedures.

6) **Embedded Firmware Programmers** – will need to have checks and balances on versions of code, components of code, libraries and software and ensure similar traceability of version, recognising these may not correspond with manufactured PCB revisions.

Supply chain risk across the manufacture of devices

Responsible suppliers of embedded into professional/industrial IoT are now offering a *Production Readiness service* – this can help convert such fast prototype kit designs into reliable platforms that are proven for volume deployment.

For a handy checklist, try the *9 success factors for reliable production of embedded*, as suggested by UK’s BitBox Ltd, whose platforms cover many M2M and private-cloud IoT, some protecting >100,000 nodes: [http://www.bitbox.co.uk/design-services/arduino-volume-electronics-production](http://www.bitbox.co.uk/design-services/arduino-volume-electronics-production)
Open Source Software Policy

Most companies Open Source Software in their in-house or third party sourced do not have a fully defined open source software policy or a cohesive view of what should be in an open source software policy.

Without a clearly defined policy companies will leave themselves exposed to risks such as security, licence compliance or operational risk.

It is not a straightforward task creating a policy and it is not a one off task but rather that something that is always evolving to reflect developments in the open source software industry.

The high level steps to creating a policy are:

- Identify key stakeholders
  - Developers
  - DevOps
  - Legal
  - Human resources
  - Management: CTO, CIO, CEO,…
  - Software architects
  - Security CISO, CSO..
- Elect and executive sponsor
- Secure stakeholders buy in
- Define the company’s strategy
  - Reduce IT costs
  - Leverage open source communities for skills and faster development
  - Contribute back to open source communities
- Risk management
  - Security
  - Licence compliance
  - Operational risk
- Scope
  - What is covered?
  - Who is covered?
- Open source software approval process
- Audits of source code and processes
- Source code maintenance and related service level agreements
- Create a draft policy
- Get widespread review and acceptance, starting with your stakeholders
- Validate
- Communication plan
- Maintain and evolve

The open source software policy should be a positive contribution to the organisation and employees should want to be engaged with the policy rather than the process being viewed as
placing an unnecessary burden on an individual’s job role due to lack of understanding behind the project.

Open Source Software Development

Open source software is now broadly used in the development of software applications. The ability to re-use components of code already created allows development teams to create more code, with more functionality, faster. It also promotes the adoption of standards and makes applications more interoperable.

Although open source software components typically require no licensing fee, it does come at a cost. This cost is uncertainty – or perceived uncertainty in many cases. That is, uncertainty of the ownership structure, of the licensing terms, of the stability of the code. Most software developers will be meticulous about what components they use from the perspective of functionality as they want to build code that works.

However, those open source software components could have inherent business risks associated with them which should not be solely down to individual developers to be responsible for. Those risks are:

- Legal risk/licence IP compliance – Open source software components licence analysis discovers legal obligations as well as potential intellectual property (IP) risks.
- Security vulnerabilities - Uncovers security vulnerabilities contained within Open Source components.
- Operational risk - Ensuring Open source software components meet required technical and architectural standards.
- Community support – Is there a sustainable community support open source components

Organisations in house developing should have open source software policies that govern how developers use open source software components. One way to address the code risk is a source code review or audit prior to releasing an application. However, there is an increased difficulty and cost applying fixes to deployed embedded devices as are found in IoT devices.

It is imperative risk monitoring of components is undertaken all the way through the software development cycle. The earlier issues and vulnerabilities are located, the less impact it will have on development and the cost overhead of managing risks as a whole and ultimately on meeting business deadlines. Equate finding licensing irregularities, problematic IP, or potential security vulnerabilities in a software application to finding a bug in a software application. The earlier it is discovered the less expensive and impactful it is to correct.

An efficient process would include pro-active source code monitoring. This will lead to a more continuous compliance model. In this model there is monitoring of open source software components throughout the development cycle. The first stage would be to implement software component package pre-approval which if implemented well should head off issues from a risky component being integrated in an application. This is where a developer must have approval from a designated manager to use a third party open source component in their code.

As stated earlier there would need to be a policy guiding the manager in their decision to accept or reject the request. Typical information that would enable a decision to be made would be

- Project & package information
Conclusion

IoT is about seamless experience of the benefits of connecting the information in our daily lives. This necessarily leads to an IoT comprised varied and powerful hardware and code that should flow in our homes and lives without the end-user worrying about software discovery or identification or control... Yet these controls will necessarily become more challenging due to the diverse nature of IoT elements and the very nature of IoT data interactions.

Also, the very promise of IoT is that we enjoy these benefits without users needing to stop to inspect this maze of technology, register sources or credentials. Indeed, it will not be as easy for us to spot or intervene against vulnerabilities. So we will be much more reliant on others and on new kinds of controls and the good standards, policies, process, procedure and proven methods covered above.

We must conclude, with IoT bringing fresh asset management challenges there is a pressing need to apply these good controls from the ground up and a tangible value add for both industry and users. Fortunately, we do have at the ready extensive toolkits, resources and valuable proven methods as we’ve shown. Many are quite suited to be reapplied to the new world of IoT and achieve for us all a high level of confidence in end-to-end security and compliance.

If we draw on these lessons our IoT, even with all its fluid new embedded ecosystems, can bring within it the effective controls needed for well-protected interactions and trustworthy operation.