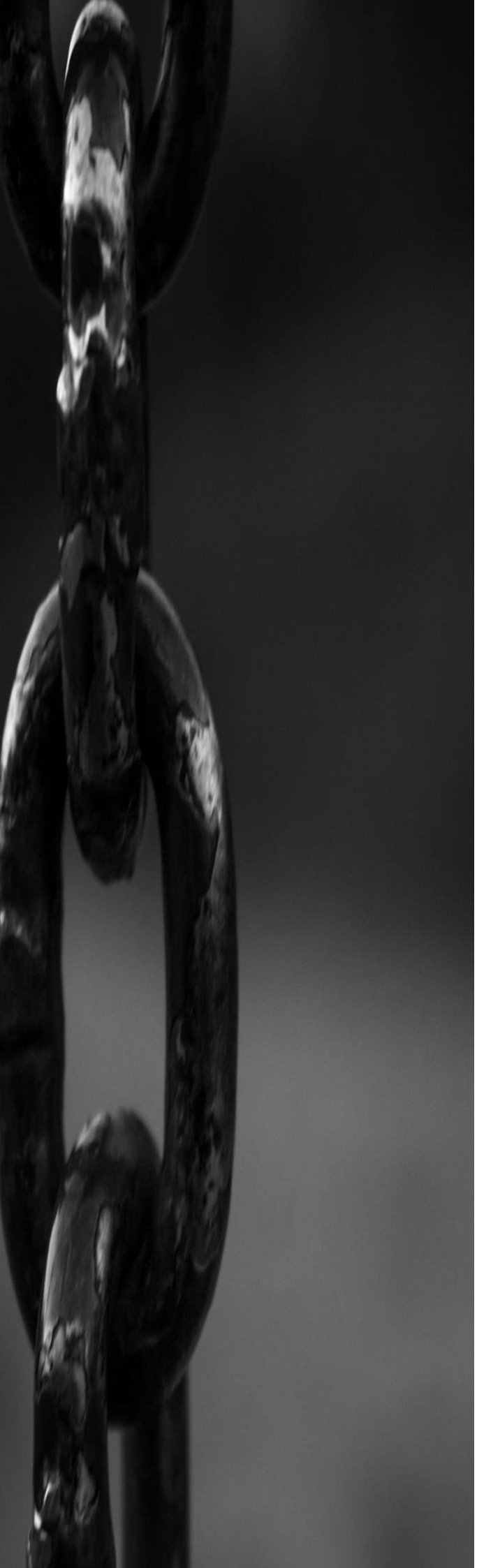


Security Foundation

Securing the Internet of Things Supply Chain

An IoTSF Whitepaper

Release: 1.0.0

Acknowledgements

This paper is an output of the IoT Security Foundation Supply Chain Working Group:

Editor and chair

Amyas Phillips, Ambotec Consulting

Working group members

Amit Rao, Trusted Objects

Anjana Priya, Microchip

Michael Richardson, Sandelman Software Works

Prof. Paul Dorey, CSO Confidential

Rob Brown, Jitsuin

Contributors

Alagar Gandhi, FCA

Alexandru Suditu, OMV Petrom

Andrew Frame, Secure Thingz / IAR Systems

Angela Mison, University of South Wales

Brendan Moran, Arm

Carlos Serratos, Brightsight

Carsten Maple, Warwick University

Chris Torr, MAOSCO Ltd

Christina Skouloudi, ENISA

Colin Lynch, EPS Programming & Logistics

Craig Ormerod, TUV SUD

Dr Gregory Epiphaniou , Warwick University

Hannes Tschofenig, Arm

Haydn Povey, Secure Thingz / IAR Systems

Heather Zhan, Scotiabank

Hemanth Thavarae, TCS

Ivan Reedman, NCC Group

John Moor, IoTSF

Jon Geater, Jitsuin

Khaleel Ahmed, L&T Smart World & Communication

Marc Canel, Imagination Technologies

Mark Davison, Terzo Digital Limited

Mark Zwolinski, University of Southampton

Nirmal Misra, Device Authority

Paul Lockley, Device Authority

Reed Hinkel, Arm Limited

Richard Marshall, Xitex

Ritesh Kumar Kalle, Rakuten Mobile

Rob Wood, NCC Group

Shadi Razak, Angoka Limited

Stanley Berg, Arrow Global Programming Services

Stephan Spitz, Secure Thingz / IAR Systems

Suniel Kumar, Nexiot

Suresh Dantam, Techsys Automation Pvt Ltd

Sylvie Wuidart, STMicroelectronics

Venkatasubramanian Ramakrishnan, L&T Smart World & Communication

Wayne Goodwin, EPS Programming

Table of Contents

About the IoT Security Foundation

The Internet of Things Security Foundation (IoTSF) is a non-profit, global membership organisation. Established in 2015, we are an international response to the complex challenges posed by cybersecurity in the expansive hyper-connected IoT world.

Working collaboratively, IoTSF is the natural destination for IoT technology producers and users which includes cybersecurity professionals, IoT hardware and software product vendors, network operators, system specifiers, integrators, distributors, retailers, insurers, local authorities, and government agencies.

Thank you for downloading this paper – we hope it is useful and helps you with your security journey. We also invite you to look at our website for more informative guidelines, reports, conference talks, blogs and more.

We would also like to invite you to join our growing membership base – whilst our publications are a good source of knowledge, being a member offers superior value for knowledge exchange and brand status.

See our website for more:

IOTSF.ORG or IoTSecurityFoundation.org

BUILD SECURE – BUY SECURE – BE SECURE

1 Purpose of this document

The ICT security community has been giving increasing attention to the problem of protecting software and hardware assets in supply chains, developing a variety of recommendations. Many of these recommendations are applicable to IoT devices but doing so requires a detailed understanding of the IoT supply chain. There is also a need for IoT-specific security recommendations to accommodate IoT device supply chains' unique characteristics.

An IoTSF working group was formed in April 2020 to supply both these needs with an expanded and updated set of security requirements concerning smart devices' supply chains. The group received contributions from 43 experts representing 34 organisations. Drawing on these inputs the working group developed a general characterisation of IoT device supply chains before proceeding to a threat analysis of provisioning operations. In parallel, the group surveyed a range of standards and literature for known attacks and existing advice (see Appendix A: References). From this work, a set of supply chain security principles specifically for IoT was compiled, mapped into Release 3.0 of the IoT Security Assurance Framework [IoTSF-SAFv3] [1] as numerous pre-existing and 29 new requirements, along with an appendix summarising the working group's characterisation of IoT supply chains.

This document expands on those recommendations and characterisation as a stand-alone commentary. The specific aims of this document are to:

1. Provide IoT-reliant organisations with knowledge and tools to identify and manage cyber security risks introduced via IoT device supply chains.
2. Enable vendors in the IoT ecosystem to better protect IoT assets against attack on their way through the supply chain.

[1] https://www.iotsecurityfoundation.org/best-practice-guidelines

2 Who should read this document?

This white paper is of interest to anyone building connected embedded products:

- CEOs interested in understanding risks to their brand, revenue, and IP
- CISOs responsible for big-picture security and compliance
- Product development managers looking to fulfil security requirements
- Engineers designing new connected products and their production processes
- Production managers evaluating manufacturing service providers

See also Appendix D: for suggestions on how to use this paper.

3 Introduction

3.1 Why IoT users should care about supply chains

Imagine that new connected equipment has turned up at your facility. You've selected it very carefully through a rigorous procurement process, no doubt considering its excellent security certifications. You are happy. You are looking forward to putting it to work. It is natural to accept it at face value.

If you do that, you should be aware that you are making a very big assumption. You are assuming that what has been delivered to you is what you were promised. You would *like* to believe that what you have in front of you is a transformational production system, but you don't know. What you *really* have is a black box. The asset sitting in front of you now has travelled a long supply chain to arrive there, which you know little or nothing about, and the reality is that for all you know it could have been compromised at any point.

When you decide to let that asset into your system you are deciding to trust large parts of its supply chain as well, and you won't even know which parts because the supply chain is opaque to you.

3.2 The big picture

This is the reality for the many organisations now operating connected assets as part of operational technology applications, exploiting the real-world monitoring and control capabilities of Internet of Things (IoT) technology. As confidence grows in their ability to deliver innovation, efficiency, and usability such applications are being deployed into ever-more critical operations.

The dark side of this progress is that as everything turns into an Internet device, cyber security turns into everything security [Schneier-IoT-security]. Connected devices are exposed to

cyber-attack from potentially anywhere on the planet and it is all too easy to imagine scenarios where compromise of even a single IoT device could have catastrophic consequences for valuable assets, the environment, human lives, or all the above.

Quantifying such risks is all but impossible, so the usual approach in cost-benefit calculations is to assume they are zero, assuming adherence to design and procurement requirements designed to raise the cost of a successful attack far beyond the likely advantages to any attacker.

The problem with this is that attackers are fully aware that neither requirements nor implementations are ever perfect. They know they only need to identify one vulnerability that has been left undefended. If they can win that battle of wits, the cost-benefit defence collapses. Whilst over the last decade defenders have benefited from development of a wide security literature [ETSI-EN303654] [ISO/IEC27001:2013] [NIST-CSF], as well as embedded hardware and software stacks' remarkable new security features, there is a new front line in this battle, and it is supply chains [ENISA-supply-threats] [NIST-IR8276].

3.3 IoT supply chains are vulnerable

IT systems, including IoT systems, can be compromised by cyber-attacks in their supply chain. Components compromised in the supply chain and then deployed into operational environments open the way for a variety of exploits.

Supply chain attacks are extremely cost effective from attackers' points of view. IT assets coming from development, manufacturing and distribution environments are often trusted implicitly by downstream users, despite weak or unknown security controls in those environments. Furthermore, a successful compromise of a single well-chosen IT vendor can fan out to the vendor's entire customer base, as products and software updates are deployed. It is no coincidence that many of the most notorious cyberattacks have been supply chain attacks.

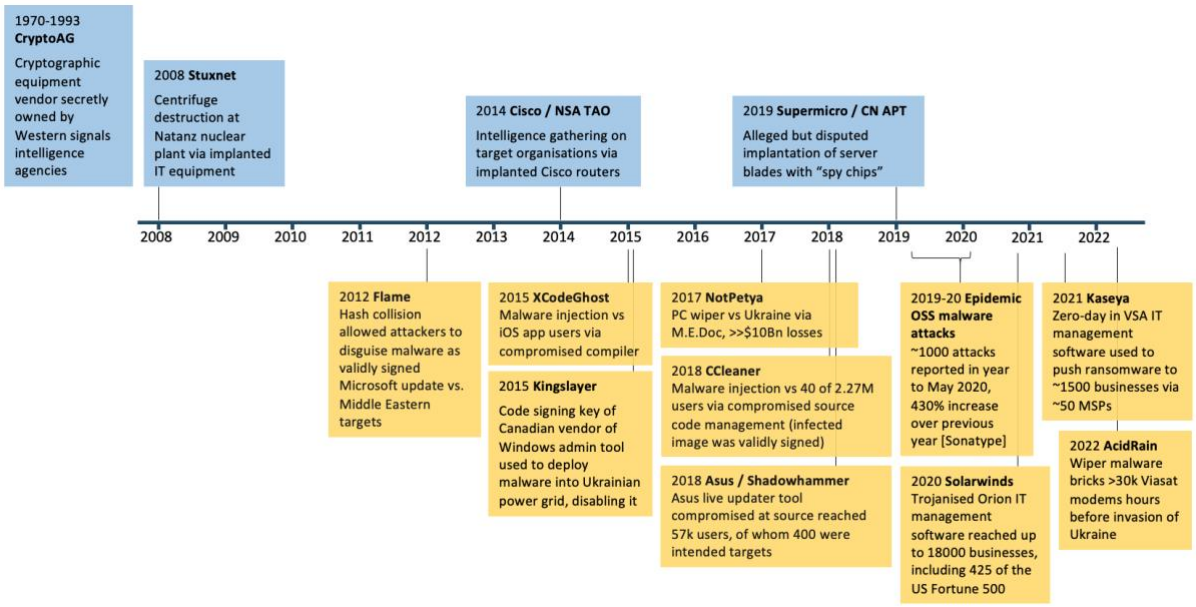


Figure 1 Timeline of well-known hardware and software supply chain attacks

Attackers have recognised that supply chain is the soft underbelly of ICT security. In some domains, the importance of securing supply chains is already well recognised, and steps have been taken accordingly [TIA-SCS9001] [FS-ISAC-third-party] [NERC-CIP-013-2] [MITRE-deliver-uncompromised]. For example, inventory and provenance tracking is an important part of modern software DevOps pipelines, using automated inspection and code signing tools. Another example comes from mobile telephony, where the GSMA has long required UICC provisioning processes to be certified as rigorously as is UICC hardware and software [GSMA-SAS].

The IoT industry must take steps of its own or see supply chains become a damaging new source of vulnerabilities.

4 Anatomy of an IoT device

To understand how to secure the components of an IoT device through its supply chain we need to start by taking a closer look at what they are.

Component	Examples	Basis of trust
Hardware	PCBs Passives Processor ICs Other ICs Enclosures Connectors Cables Wireless modules Sensors Motors	OEMs are expected to design devices suitable to their task and manufacture them into known physical condition, verified by QA inspections, and protected by anti-tamper features. The main threats to devices' physical integrity in the supply chain are poor process control, poor QA, and the use of grey market parts [NCC-device-manufacturing] [ENISA-hardware-threats] [Huang-Counterfeit-ICs]. In deployed environments device hardware is vulnerable to sabotage unless physically secured.
Processor IC Special case of **hardware**	Microcontroller Unit (MCU) System-on-Chip (SoC)	In hardware terms by far the biggest determinants of devices' behaviour are their processor ICs. The integrity of IC hardware can theoretically be compromised in design, fabrication, and distribution [Areno-IC-supply-chain-threats] [Huang-IC-implants]. No such attacks have ever been conclusively and publicly detected, but against operational ICs "glitching" attacks are both cheap and effective.
Software	Bootloaders Supervisors Operating systems Libraries Applications	Software determines much of the actual behaviour of an IoT device. Protecting its quality and integrity is therefore a large part of device security. Confidentiality of device software is often also desired by device developers, to prevent IP theft, vulnerability reconnaissance and counterfeiting. OEMs are expected to specify software sufficient to its intended application, design and develop it to that specification, maintain the integrity of specifications, designs, and source code in their own environments and of built software artefacts throughout the downstream supply chain. IoT device software typically has numerous upstream dependencies. Upstream maintainers are relied on for quality control, maintenance, and integrity of these libraries. Assurance of this requires that OEMs inventory third-party software components and assure themselves of the maintenance status of each, for the deployed life of the device.

Component	Examples	Basis of trust
		Because software is an informational asset which can be signed by its builders for later validation by provisioning sites or target devices, it is possible to eliminate the need to trust intervening supply chain actors. However at least the first software installed will be exposed in a manufacturing environment, which therefore must be trusted to protect that software's integrity and confidentiality. Beyond the point of installation, software integrity and confidentiality rely on the activation of IC features designed to prevent unauthorised modification and disclosure.
Root of Trust (RoT) firmware Special case of **software**	Initial bootloaders Cryptographic module firmware	Root of trust firmware is any device software on which remote parties must depend implicitly because there is no way of remotely verifying it or detecting its corruption[2]. It is invariably made immutable on devices to protect against alteration via software attacks. A device may have one or several RoTs. They can have any functionality but usually it is kept as small as possible to minimise both the attack surface and the amount of code that cannot be updated. IoT MCU RoTs commonly include secure initial boot, identity, and in many cases a full suite of cryptographic functions. RoT software images are typically generated per device model and installed onto any number of individual devices. As with all software the integrity of RoT software must be protected through design and development, before and during provisioning, as well as on target devices. If RoT images contain secret keys their confidentiality must also be protected.

[2] Precise definitions vary. The term is sometimes used to denote any more-trusted software component, not specifically the most-trusted.

Component	Examples	Basis of trust
Initial bootloader Special case of **RoT firmware**	ROM bootloaders	When a microprocessor turns on it starts executing instructions from memory. IoT devices with a trusted boot process always start execution in the same place: the secure initial bootloader. The trusted initial bootloader makes sure the main application is authentic and has not been tampered with, before starting it. This check can't be bypassed by software attacks because the trusted initial bootloader is protected by silicon features. Trusted initial bootloaders are also an important part of the remote software update process, detecting, validating, decrypting, and installing new application software images. Initial bootloaders are installed directly to hardware, via a hardware interface, very early in a provisioning process, often by IC vendors or their subcontractors, sometimes by device OEMs.
Cryptographic Root of Trust (RoT) firmware Special case of **RoT firmware**	Cryptographic accelerator firmware	It is critical that attackers can't guess, intercept or extract devices' secret identity keys, so IoT MCUs often feature special hardware for on-board key generation, plus hardware-protected storage locations which prevent physical extraction of keys. To block software attacks these keys are isolated so application software, which is more complex and more easily hacked, cannot access them directly. Instead, it calls on specially sandboxed cryptographic RoT functions when it needs to use them. Cryptographic RoT firmware is always installed directly to hardware, via a hardware interface, often as the very first step in a provisioning process, by IC vendors or their subcontractors.
Production data	Serial number Serial number certificate Batch number Production date Hardware version	Data describing device production and hardware is often provisioned directly onto devices at time of manufacture for subsequent inventory and fleet management, detection of hardware variants by common software, or in some cases anti-counterfeiting checks by on-board software. Its integrity depends on that of the production process installing it, and its subsequent protection on target device.

Component	Examples	Basis of trust
Symmetric software decryption key	AES256 key	Some IoT projects encrypt over-the-air software updates using symmetric keys distributed to whole classes of devices at manufacture. This may be necessary in highly constrained environments, but such keys are highly vulnerable to disclosure, either via ill-secured production environments or by extraction from any deployed device. Reliance on them is widely discouraged.
Boot software validation key	ECC public key	A public key belonging to whichever organisation is maintaining the software loaded by the initial bootloader - usually the device OEM. This key is sometimes installed along with the secure bootloader, sometimes afterwards via a programming interface or internal API. It is usually the same for all the devices of a particular model. IC vendors are expected to provide hardware features preventing anyone but the software maintainer from changing this key. Device OEMs are expected to ensure the correct key is installed and that it is secured on the device. Software maintainers are expected to strictly control use of the corresponding private key, used for signing software images.
Server certificates	X.509 certificate	Public keys provisioned onto each device for the authentication of remote services, typically in the form of X.509 certificates for use in TLS protocol handshakes. Usually, the same for all devices of a particular model or production order, they are often included in production software loads. IC vendors are expected to provide hardware features preventing anyone but the software maintainer from changing this key. Device OEMs are expected to ensure that the production process installs the correct keys and secures them on devices. The operators of the servers are expected to strictly control use of the corresponding private key, used to authenticate their services to devices.

Component	Examples	Basis of trust
Application configuration data	Server addresses Server certificates Application settings Administrator credentials Media libraries ML models Loadable applications	Many device designs allow application settings to be customised at the point of installation. Some designs do this in the factory, to reduce fieldwork for installers or to eliminate the cost of user configuration interfaces. Device operators must assure themselves that the integrity and, if necessary, confidentiality of this data is protected wherever it is installed, and on devices, and in the case of executables throughout its design and development.
Identifiers	Serial numbers UUIDs IC unique IDs Device public keys	Identifiers are used to assert a device's identity to other devices and services. They cannot be secret because such assertions must be sent in the clear. They must be persistent for at least the lifetime of the relationship between the device and its correspondent because corresponding parties use them as an index to look up and store information about the device, e.g., public keys, permissions, application state, activity logs. Most importantly they must be unique, so they can identify an individual device. Very large namespaces may be used so that identifiers can be generated randomly without compromising uniqueness. Where smaller namespaces are used, generation requires access to a database of available identifiers. Many ICs have unique ID numbers fused into them by their manufacturers. If these do not suffice, device OEMs can create and install identifiers as part of their manufacturing processes. They can, if they choose, use the public part of asymmetric device identity keys, or values derived from symmetric device identity keys.
Identity keys	AES key ECC private key	Devices creating or accessing relationships with correspondents over shared network infrastructure must prove that they are who they say they are. This is accomplished by proving possession of a cryptographic secret. That secret is the device's identity key. Proof of possession procedures begin with the verifying party sending a challenge, usually a random number, to the asserting device. The asserting device responds by returning

Component	Examples	Basis of trust
		the challenge, encrypted using its secret identity key. The verifying party then decrypts the response and checks it matches the original challenge. Note that before a remote service can verify a connecting device someone has to have told it the identifiers of devices which it should trust, and the corresponding identity keys with which to authenticate them. Confidentiality of identity keys is a cornerstone of IoT systems' security. No device can be reliably identified if its identity key is disclosed. The boundary of identity keys' secrecy depends on the type of cryptosystem being used. In public key cryptosystems decryption uses the public part of the device's secret identity key. The secret part need be known to nobody other than the device. In symmetric key cryptosystems decryption and encryption use the same key, which needs to be known to device and verifying party both. Identity keys may be generated externally for subsequent installation onto devices, but the risk of disclosure is lower if they are generated internally by devices. In some cases, they may be already present in IC hardware obtained from suppliers. In all cases the quality of the key depends on correct generation and access to a high-quality source of randomness.
Device certificates	X.509 certificate chains	An X.509 device certificate contains an identifier of the device, its public key, and a signature by the issuing certificate authority (CA) over both. The certificate is an assurance by the CA that the identified device possesses the private key corresponding to the given public key. Third parties accepting this assurance can use the device's public key to authenticate it. Properly run CAs require requesting parties to prove possession of the private key, but issuance of a certificate can be made conditional on any number of additional requirements. The existence and enforcement of such requirements is not encoded in certificates: it is metadata about the CA [CSI-firmware] [Richardson-mfr-key-security] . If relying parties' designers know about the conditions

Component	Examples	Basis of trust
		applied by a CA they can choose to rely on its certificates as assurances that the conditions were met. For example, a CA can be set up at the end of an IoT device production line to issue certificates only to devices which are fully assembled and have passed all quality control processes. Third parties who are aware of this arrangement have strong assurance that devices presenting certificates issued by this CA are not counterfeit. It is common to store certificates on devices themselves, to present as needed. This lifts from manufacturing operations the burden of batch-uploading device identities to remote services. Instead, devices can upload their own details on first connection, e.g., as part of a TLS handshake. Remote services only need to check that the certificate was signed by a CA that they trust. A one-time setup to tell the remote services which CAs to trust is sufficient. A CA's public key can itself be signed into a certificate by another CA, lengthening the certificate chain of trust. Such a chain can extend through multiple intermediate CAs up to a root CA. Instructing remote services to trust any device with a specified intermediate CA in their chain of trust allows production line CAs to be rotated in and out of service. Anyone relying on a device certificate needs to understand that they are relying on i) the CA's signing key being kept secret, ii) the public key which they are using to authenticate its certificates genuinely belonging to the CA on which they want to rely, and iii) the party operating the CA having the necessary access and competence to make the assurances that they offer, and the necessary commitment to providing them accurately. Deciding whose assurances are credible is almost always a design-time decision.

Table 1 Components of IoT devices

5 The IoT supply chain

To understand how to secure the IoT supply chain we need to take a closer look at what it does and in what order.

The job of an IoT device supply chain is to deliver devices into an application in a known, trustworthy, and trusted state. It extends forward up to the last point where anyone might take delivery of a device, which is to say all the way through devices' deployed lifetime up to the point where they are permanently put out of service. It extends back beyond knowing, fanning out into components and subcomponents, pulling in tools employed along the way to decompose in turn. Any analysis is forced to draw a line somewhere. The sensible place for that line is the point where one is prepared to trust incoming components - either because their integrity is obvious or explicitly tested, or because they're coming from a trusted supplier. Even then it is necessary to take a view on what constitutes "trusted". For this one needs a working knowledge of what suppliers actually do, as well as a view on what they should do. Fortunately, this is a task that can be delegated to certifying bodies and other specialists.

Each component of an IoT device is the product of a preceding design and production process. It is more realistic to think of the supply "chain" as a supply "network". Anyone in the supply network with access to unprotected assets becomes part of the trust base of that device. Producers of key components and tooling such as embedded operating systems, cryptographic libraries, compilers and ICs carry a significant burden of trust and must demonstrate that they deserve it. But, as the designer of the production process, it is the device OEM who chooses whom to trust and is responsible for securing it overall.

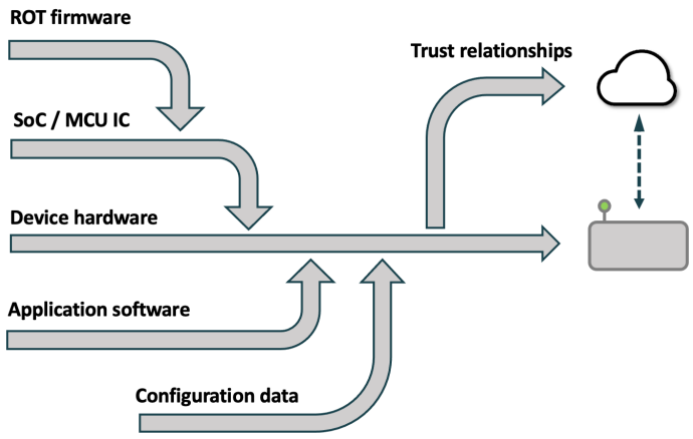


Figure 2 Main branches of a typical IoT device supply network

This supply network comprises of six basic types of operation: *design*, which gathers requirements and specifies a system able to deliver them, *development*, by which functioning production assets are created and put in place, *hardware assembly*, which progressively integrates components and subassemblies into complete devices, *programming*, which

installs logical assets onto physical devices, *personalisation*, which generates a unique identity for each device, and *onboarding*, which places those devices into trust relationships with other systems. Programming, personalisation, and onboarding together comprise the *provisioning* process, by which hardware is put into a functioning state.

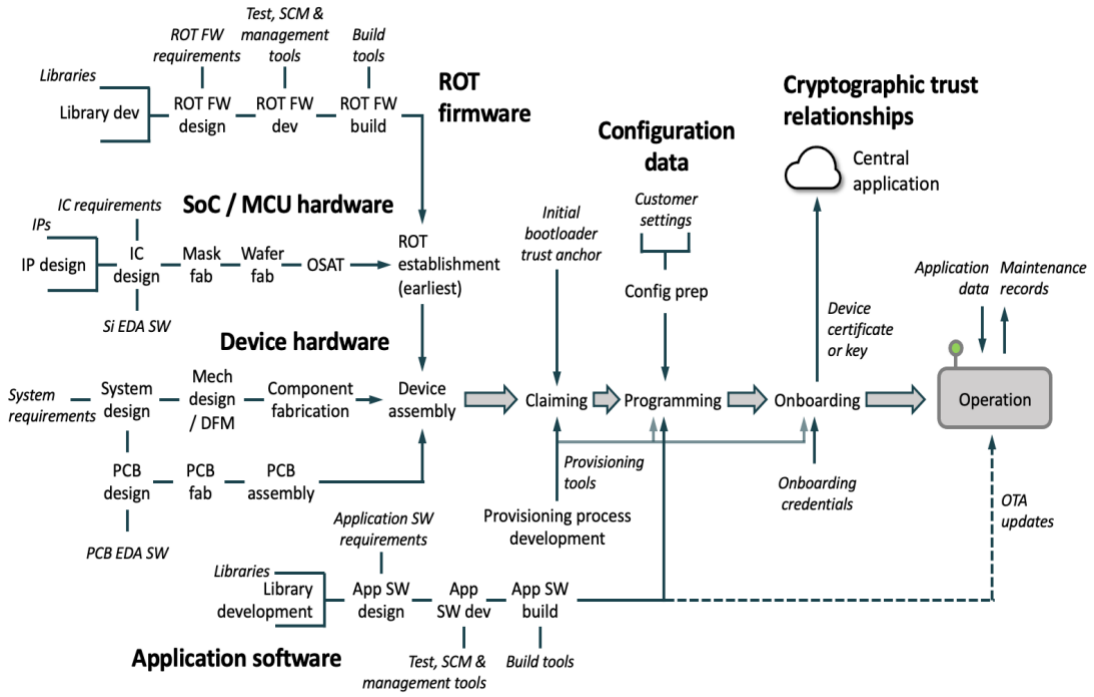


Figure 3 A more detailed look at a generalised IoT device supply network. A similar diagram for a real IoT device would likely feature multiple instances of each kind of branch, multiple hardware integrations and multiple instances of the different provisioning operations. It might also extend through multiple deployed phases.

A cyberattack could potentially be launched at any step in a smart device's supply network.

Design and development are project-type activities which have much in common with software and hardware design and development projects everywhere. During these phases security is identical with quality, because it can be impossible to distinguish between a bug and sabotage [Finite-State-Huawei] [NIST-SP800-161]. Fortunately an extensive literature exists on both secure software development lifecycles (SDLCs) (see Secure software development lifecycles (SDLCs)) and good IT practices to secure development environments [ISO/IEC27001:2013] [NIST-CSF].

Software is particularly vulnerable to development-time attacks (or bugs). Instead of relying on in-house resources OEMs outsource development of the most security-critical components such as RoT software and production tooling needed to personalise devices to trusted vendors, or to trusted OSS projects. Ultimately these components must be configured and integrated into one or more embedded applications and that remains the OEM's responsibility, not just prior to production but for the supported life of the device, during

which its operators should expect and receive timely security updates whenever any component receives a security update. To the extent that such updates are signed or delivered by the OEM/ODM they have a responsibility to ensure that an attacker cannot sign or deliver malware.

Assembly, programming, personalisation, and onboarding are all manufacturing operations, conducted in various manufacturing environments using the production assets and components delivered by preceding design and development and manufacturing phases.

Assembly operations and hardware assets are undoubtedly security-critical; however, hardware components are far less likely than logical ones to be attacked in the supply chain. In any case by far the biggest hardware determinant of devices' behaviour is the processor IC, the design and manufacture of which is outside of device OEMs' control.

Provisioning operations are characteristic of IoT supply chains and necessarily involve handling software assets and private keys, whose integrity and confidentiality must be protected. Most IT security literature concerns itself with software and infrastructure as a service, and the deployment by end users of software onto commodity servers. Securely programming and identifying machines in manufacturing and distribution isn't a common problem, so few standards address it. Those that do are focused on mobile devices. In developing our recommendations for securing IoT supply chains we therefore give provisioning operations special consideration (Table 2).

Operation	Description
Programming	Programming is always performed via a programming interface exposed by the target. Programming operations place software and configuration assets onto devices. These can include assets such as: ● Software images and server certificates, which are the same for every device ● Manufacturing data and customer-specific settings, which change per batch ● Identity secrets and device certificates, which are individually personalised for each device. Device operators rely on the authenticity and integrity of all these assets – and, in the case of identity secrets, also their confidentiality. Device OEMs and ODMs on their part often have an interest in maintaining the confidentiality of their software IP. Programming is rarely as straightforward as writing a standard binary image onto each device passing down a production line. Sometimes binaries are rebuilt per device to check for a specific IC hardware ID, as a defence against cloning. In other cases, configuration data is installed as late as possible in production, or even deferred into distribution. Device identities might be

Operation	Description
	generated externally and programmed individually. A secure bootloader might have been installed by the IC vendor, so that in the CEM it is only necessary to load a signed, encrypted image. It can even be a remote update operation performed in the field. All programmed assets must be protected not just in the programming environment but on the target IC. Because of this, Ics entering a secure programming environment must be authentically what they are believed to be, and they must be configured to prevent unauthorised readout or modification of assets before they leave.
ROT Establishment Special case of **Programming**	Ics fresh off the wafer typically expose a hardware-level programming interface. With no identity or correspondent software already present, this channel is necessarily unencrypted and unauthenticated. The first programming step must therefore take place in a secure, trusted facility (see [CC-site-certification] [GSMA-SAS] for examples). If a suitable ROT is established by that first programming operation, it can expose secure interfaces for provisioning subsequent assets. Examples of this pattern include secure boot loaders which can detect and install new valid software images, and secure programming interfaces. Both are often found as features of ROTs installed by IC vendors.
Claiming Special case of **Programming**	An OEM making use of a secure boot loader established by the IC vendor must **claim** it by programming it with a trust anchor with which to validate the next software in the boot chain. Like ROT establishment, this is a special case of programming. Claiming is a key moment in the life of an IoT device because whoever installs this trust anchor chooses what software runs and thereby takes full control of the behaviour of the device [OCC-firmware-ownership].
Personalisation	Every connected device requires a unique, authenticable identity. Ideally devices should generate asymmetric identity key pairs internally, so the private key need never be exposed externally. Most modern microcontroller ROTs are able to generate high quality key pairs. Older or smaller microcontrollers may lack robust sources of high-quality entropy. Their private keys must be generated externally. Ideally this is done as close to the target device as possible to limit the potential exposure of those keys. The provisioning tool is an ideal place to accomplish this. Personalisation can also include serial numbers and other public identifiers.
Onboarding	IoT devices are useless until they are connected into larger applications. Those applications need to be told which devices to trust and how to authenticate them. There are various ways of doing this, but all involve telling the central application to trust devices which can prove possession of specified secret keys. We call this **onboarding**.

Operation	Description
	The act of onboarding is a major trust decision. When a device operator decides to trust an IoT device they also making a decision to trust the supply chain that delivered it to them, including everyone who has had access to the device and its components. Those components include not just the embedded software developed by the device OEM or ODM, which the operator is expecting to behave exactly according to specification, but all the libraries and hardware elements on which it depends. Device operators unfortunately are not usually in a position to determine for themselves whether an IoT device has been provisioned into a known, trusted, functional initial state. Instead they must rely on someone else's assurances. Someone they trust, often the OEM, needs to assert "this device is in a known trusted state". Where devices are identified using asymmetric (private and public) keys this is accomplished by onboarding the public key to central services. The simplest method is to take a copy of each devices' public key on the production line and upload it to the central service. The copy should be taken when the device is fully provisioned, but before it leaves the trusted manufacturing environment. A more powerful and flexible method is to sign each device's public key into a certificate chain on the production line and load that certificate chain back into the device. The device can later deliver its public key to the central service itself, as part of a TLS handshake. Central services can onboard that key on the authority of any Certificate Authority (CA) certificate in the chain. This method allows large volumes of devices to be onboarded in a single operation. In both cases, whether keys are onboarded directly to the central service from the production line or signed into certificate chains of trust, it is essential that only trusted parties perform that operation [RFC-6024-TA-management]. The fewer entities involved the better. In this respect signing devices into chains of trust offers a further advantage because CA keys can be stored in an onsite HSM or secure element, or offsite in a secure facility, where they can be used without ever being exposed in manufacturing environments. It is important to note that the private keys of all the Cas in the chain of trust must be similarly protected, because an attacker gaining the use of any of them gains the ability to onboard any device they choose [Richardson-mfr-key-security]. Note also that the right to onboard devices, which may repose in possession of a CA private key,
Reset	Resetting a device means to return it to an earlier stage in its provisioning sequence so that it can be reprovisioned with new software, data and relationships. Many IoT devices are provided with some kind of reset

Operation	Description
	function. Common scenarios are change of owner, repairs, recovery from compromise, and responsible disposal. A change of user without changing central service only requires a return to the user-association state. This may involve erasing settings and data from the previous user and re-enabling the user association procedure. A change of central service likely requires devices to be removed from their previous service, returned to a pre-onboarding state and onboarded to the new service. The operator of the new service ought to be as well-assured of the devices' integrity as the previous one. Note that removal from the old central service is usually accomplished on the server side, either by removing (or "overboarding") departing devices' identities from services' list of trust anchors, or by explicitly blacklisting them. On the device side the only sure method of removing a relationship with a remote service is to erase the identity key used and generate a new one. This would be inconvenient if that identity were used in other relationships, so designers may find that a good rule of thumb is to have devices use different identities for different relationships. Repair scenarios may require diagnostic access, drop-in replaceability, or remanufacture capabilities. Diagnostic access is a significant security surface but is not a reprovisioning scenario – unless it permits any kind of write operations. It is essential that those are secure exactly as much as the original provisioning operations, if their existence is not to undermine user's confidence in the devices. Similarly, drop-in replaceability should never be accomplished by cloning a faulty device's identity because the mere possibility renders all such devices less trustworthy. A simple rule of thumb for these scenarios is to approach them as remanufacture, where devices are returned to an earlier state and put through the same provisioning process as when originally manufactured, equally well-secured. End of life is often not given much consideration because it is not a feature most customers will pay for. If it is implemented, it is often via physical destruction. Responsible disposal instead aims to allow device hardware to be repurposed for subsequent applications. It must remove trust relationships and confidential data from Ics and unlock them, in effect returning them to a state before OEM provisioning. Implementors should assure themselves that it is impossible for such devices to be reintroduced into their own provisioning processes.

Table 2 Provisioning operations

To deliver a smart device in a known, functional, and trusted initial state, its supply chain must provision it with many software and data assets and into many trust relationships, often in a sequence of provisioning steps that begins with a blank IC and ends with a fully functional and secured device. Each step may be performed by a different actor, using a different tool, each provisioning the device into an intermediate state. The process may begin upstream of the OEM, with IC vendors provisioning bare dies [SAM-L11-provisioning], and it may extend to as late as immediately before devices' live deployment, with installers commissioning devices on site [Interact-Pro-install-guide] [IETF-RFC8995-BRSKI] [FIDO-device-onboard-wp].

IoT OEMs already design provisioning sequences and create or specify provisioning tools for each step of those sequences, as part of their device development. Although they may not have been considered it before, protecting their devices against deliberate attacks in manufacturing is just another design goal. One way of achieving it is to allocate key provisioning steps to more-trusted suppliers. Another is to specify secure Ics and provisioning tools that can keep assets out of harm's way in untrusted supply-chain environments.

Before pursuing either approach it is important to understand the central coordinating role of each provisioning step. Figure 4 shows a generalised version, developed by surveying a wide variety of real-world provisioning operations. Those operations used a variety of tools including production-line PCs, certificate authorities running offsite in data centres, and specialist programming devices. What they all have in common is that they perform some combination of operations from Table 2 on target devices and central IoT applications, on behalf of an OEM/ODM if not under their direct control.

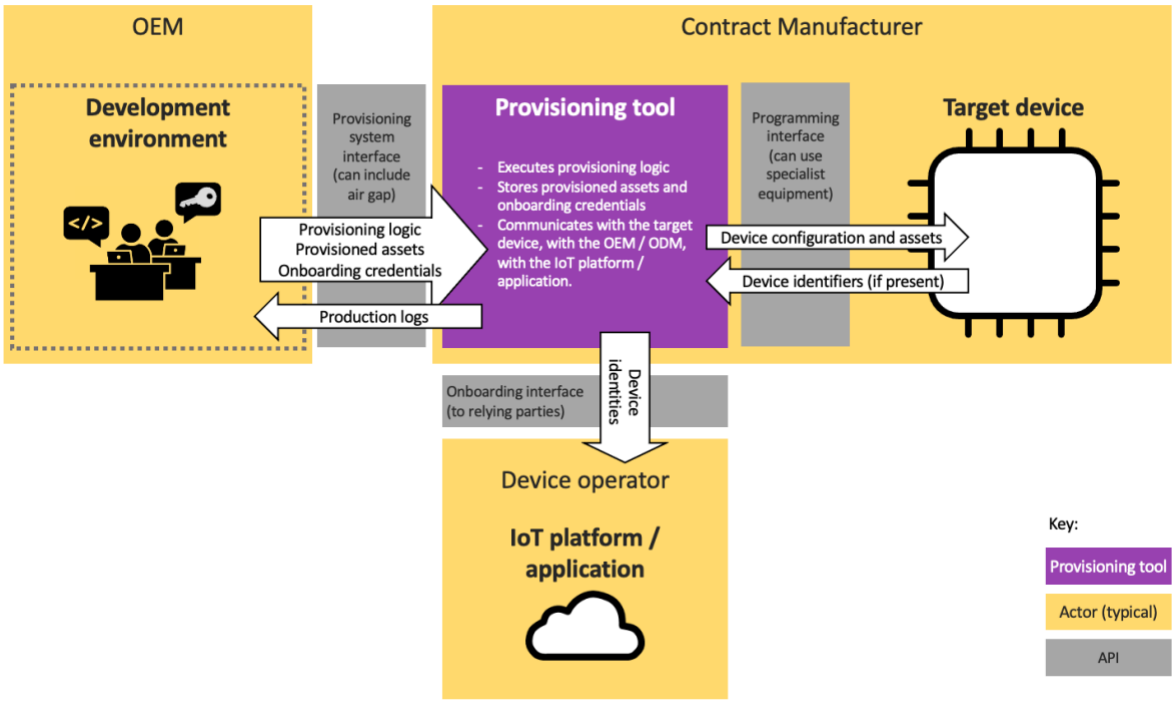


Figure 4 Generic provisioning operation

Exercising strict control over provisioning operations can be challenging when manufacturing is outsourced to contract electronics manufacturing (CEM) facilities. CEM facilities vary greatly but in many cases an OEM/ODM hands over manufacturing specifications, software assets and specialised production tools and helps set up manufacturing processes to the CEM whose primary responsibility is the delivery of a certain amount of product. Although OEMs/ODMs can select a CEM based on quality and security certifications they do not otherwise have much control over how their assets are used and protected. This is unsatisfactory because most CEMs operate on very thin margins and their focus is on cost control, not information security. To address this many IC vendors now offer processors with preinstalled ROTs that enable secure programming in untrusted CEM environments. For those not using such devices there is the option of using specialist CEMs offering secure programming services.

6 Threat model

How can OEMs secure their provisioning processes? An answer to this question must start with a threat model in which potential attackers are identified, and end with a recommendation of countermeasures. Table 3 asks "who has the capability to mount attacks on or via device supply chains, and what do they want to achieve?" The answer to this question gives us our starting point defending against them.

Group of threat actors	Goals and capabilities
Advanced Persistent Threat (APT) groups	APT groups operated by nation states prosecute espionage and sabotage missions against enemy operations and infrastructure by compromising information systems. Civil and business infrastructure, including IoT systems, is very much in scope. Such groups have deep resources of skills and time, are able and motivated to conduct long-term campaigns, and thanks to national security laws requiring the cooperation of local organisations may have physical and logical access to factory facilities in their jurisdiction.
Ransomware groups	Whilst state sponsored attacks attract much of the attention a variety of criminal actors are also developing supply chain attacks. A popular technique is to gain commit rights to popular open source software (OSS) libraries and infest them with malware. In May 2020 Sonatype reported a 430% increase in such attacks over the preceding year [Sonatype-2020]. Meanwhile, organised cybercrime groups have refined a business model known as big game hunting [ITProPortal-big-game], patiently infiltrating critical systems at victim organisations before pressing a kill switch, followed by demands for ransom. Ransomware groups share APT groups' motivation to infiltrate and disrupt and are also sufficiently capable to be a serious threat.
Unscrupulous CEMs	Alongside these new cyber adversaries OEMs face longstanding problems of IP theft, counterfeiting and other forms of revenue diversion. There are rogue actors in many legitimate manufacturing operations who want to make a little extra money and are willing to do so at OEMs' expense, whether by disclosing confidential IP to competitors or counterfeiters, increasing margins by using sub-par components from the grey market, reconditioning manufacturing discards for sale, or simply producing extra devices for sale on their own account [NCC-device-manufacturing]. The scale of these activities is often surprising to those outside the industry and all of them have the potential to benefit cyber attackers as well as eroding OEMs' revenues.

Table 3 Adversaries active against supply chains

Given these adversaries' goals we have used the formal method of attack trees [Schneier-attack-trees], developed in Appendix B:, to systematically identify potential attacks against IoT device manufacturing operations. We have focused on this because provisioning

operations are poorly covered by existing literature, because device OEMs must be able to guarantee the known, good initial condition of their devices to their customers, and because manufacturing is also where device OEMs are often subjected to revenue diversion through IP theft, counterfeiting, overproduction and other means.

The method of attack trees produces a list of potential attacks. Well-known attacks can be identified in our list [ENISA-IOT-supply-chain], as well as numerous theoretical attacks. With the exceptions of the introduction of grey market components and leakage of devices into grey markets, all the attacks involve the disclosure, modification or denial of information assets handled during provisioning. For each attack we have identified an appropriate countering security requirement, and these we have contributed into release 3 of the IoTSF Security Assurance Framework [IoTSF-SAFv3]. They are summarised in Table 4 Twenty-six principles for secure supply chains. IoT Security Assurance Framework requirements in bold are new contributions. below.

We invite readers to apply the formal method specifically to their own systems. The method of attack trees is excellent at identifying specific attacks, but it can only be as specific as the description of the system being analysed. Because we analysed a generalised provisioning system our recommendations are themselves quite general. Readers can however use the concepts developed to describe our generic supply chain to describe their specific supply chain. Anything inside the boundary of that description can specifically analysed for potential vulnerabilities using the attack trees method. Anything outside should be secured via a separate analysis, or by following recommended best practices.

In parallel with our formal analysis, we surveyed the literature on supply chain attacks and recommended controls. We found that industry bodies in some highly-regulated [MITRE-deliver-uncompromised] [TIA-SCS9001] [NERC-CIP-013-2] [FS-ISAC-third-party] and counterfeit-prone [NCC-device-manufacturing] [NCC-provisioning] verticals have issued guidelines based on insightful analyses. A number of ICT-wide bodies have published supply-chain specific analyses [ENISA-supply-integrity] [ENISA-hardware-threats] and recommendations [BSIMMsc] [NIST-IR8276] [NIST-SP800-161] [ISO/IEC-20243-1] [ISO/IEC-27036-1] [NCSC-supply-chain] [Synopsys-procurement-language]. Some have addressed IoT supply chains specifically [ENISA-IOT-supply-chain] [IIC-IoT-security]. Many have produced secure software development lifecycle (SDLC) guidelines [BSA-SDLCv1.1] [BSIMM] [ISO/IEC-27034-1] [NIST-SP800-218] [OWASP-SAMMv2] [PCI-SSFv1.1] [SAFEcode-SSD], including some for IoT specifically [ISA/IEC-62443-4-1] [IIC-software-trustworthiness] [CSI-firmware] [ENISA-IOT-SDLC]. Some have analysed specific provisioning problems in depth [OCC-firmware-ownership] [Richardson-mfr-key-security] [RFC-6024-TA-management] [IETF-RFC8995-BRSKI] [FIDO-device-onboard-wp] [RFC9124-update-info-model] including initial provisioning environments [GSMA-SAS] [CC-site-certification]. Much attention is being given to the potential role of software bills of materials (SBOMs) in enhancing the security of deployed ICT [CycloneDX] [SPDX] [SWID].

From these sources we grouped similar recommendations to obtain a small number of "best practice" principles for securing ICT supply chains, applied to our general IoT supply chain. Their motivation and implementation can be examined in detail in the original sources. To these we have added IoT-specific provisioning principles, developed from our own analysis.

Finally, to each of these principles we have mapped testable implementation recommendations from the IoTSF Security Assurance Framework (Assurance Framework). The goal of the Assurance Framework is to bring security recommendations covering all the disparate parts of IoT systems together in one document for IoT device operators and vendors, so release two of the Assurance Framework already contained many specific recommendations relevant to all parts of the supply chain. Release three adds twenty-nine new recommendations, ensuring all twenty-six supply chain security principles are fully addressed. The table below summarises the principles and how they map into IoTSF-SAFv3 recommendations.

	Principle	Motivation	IoT Security Assurance Framework Recommendations
1	Authenticate targets before performing provisioning operations on them	Attack D1.4.1, D1.4.2, A2, B1.1.2.1	**2.4.14.10**, **2.4.14.11**, **2.4.14.12**, 2.4.14.5, **2.4.14.17**, **2.4.4.11**, **2.4.14.18**
2	Disable hardware programming and debug interfaces after ROT establishment	Attack D3.1, B3.2.3	2.4.14.1, 2.4.4.9, 2.4.4.13, 2.4.4.14
3	Ensure reliable access to production assets	Attack A4	**2.4.14.20**
4	Limit factory provisioning operations in time, number or location	Attack A1.1.2, A1.2.1	**2.4.14.22**
5	Protect confidentiality and integrity of device software as it is programmed onto devices	Attack D1.3, B2.2.4	2.4.14.6, **2.4.14.21**
6	Protect confidentiality and integrity of device software *en route* to provisioning tools	Attack D1.1, B2.2.2	**2.4.14.15**
7	Protect confidentiality and integrity of device software stored in provisioning tools	Attack D1.2, A3, B2.2.3	2.4.5.32, 2.4.14.6, **2.4.14.16**

	Principle	Motivation	IoT Security Assurance Framework Recommendations
8	Protect confidentiality of device identity secrets	Attack C2.1, C2.2	2.4.9.4, **2.4.9.9**, 2.4.14.6, **2.4.4.16**, **2.4.4.17**, 2.4.9.2, 2.4.14.9
9	Protect confidentiality of onboarding credentials *en route* to provisioning tools	Attack B1.1.1.1, B1.1.3.2	2.4.9.3, **2.4.14.15**
10	Protect confidentiality of onboarding credentials stored in provisioning tools	Attack B1.1.1.2, B1.1.3.3	2.4.14.6, **2.4.14.19**
11	Verify destruction of manufacturing discards	Attack A1.2.2, A1.3	2.4.14.3, 2.4.14.4, **2.4.14.14**
12	Verify the authenticity and integrity of production logs used for onboarding devices	Attack B1.1.2.2	2.4.16.5
13	Consider security of device assets all through development and production, as well as in deployment	Best practices	2.4.3.1, 2.4.3.4, **2.4.3.6**, 2.4.3.10, 2.4.3.19
14	Devices must be uniquely and reliably identifiable, both physically and logically	Best practices	2.4.8.1, **2.4.8.18**, **2.4.14.10**, **2.4.14.11**
15	Devices should ship with, and key components should have, an explicit security maintenance life	Best practices	2.4.5.36, 2.4.5.35, 2.4.3.9.1, 2.4.12.12
16	Ensure security of supply	Best practices	2.4.14.2
17	Evaluate quality and security of components when they are designed-in	Best practices	2.4.3.24, **2.4.3.27**, 2.4.3.28, 2.4.5.37, 2.4.6.14, 2.4.5.37
18	Evaluate quality and security of received software updates	Best practices	2.4.6.14, 2.4.3.22, 2.4.3.29, 2.4.5.38, 2.4.5.37
19	Implement factory reset, user reset and other resets as reversions to an earlier provisioning step	Best practices	2.4.12.11, 2.4.16.1, 2.4.16.2, 2.4.16.4, 2.4.16.7, **2.4.14.24**, **2.4.14.23**, 2.4.8.16

	Principle	Motivation	IoT Security Assurance Framework Recommendations
20	High-quality software is more-secure software	Best practices	2.4.5.16, 2.4.5.23, 2.4.5.13, 2.4.5.14
21	Know what you've got: inventory third-party components and devices out to at least the first tier of suppliers	Best practices	2.4.14.25, **2.4.3.26**, 2.4.14.8
22	Monitor your inventory for security vulnerabilities / compromises / updates	Best practices	2.4.3.9, 2.4.3.21, 2.4.7.20, **2.4.14.13**
23	Protect assets on deployed devices	Best practices	All of section 2.4 including 2.4.4.1, 2.4.4.2, 2.4.4.3, 2.4.4.4, 2.4.4.11, 2.4.5.7, 2.4.4.6, 2.4.5.10, 2.4.5.11, 2.4.5.12, 2.4.5.15, 2.4.9.7
24	Protect integrity of software updates - and of the update signing process	Best practices	2.4.5.2, 2.4.5.3, 2.4.5.4, 2.4.5.8, 2.4.5.9, 2.4.5.19, 2.4.5.20, 2.4.9.8
25	Secure your development environments	Best practices	2.4.5.17, 2.4.5.18, **2.4.5.40**
26	Software other than roots of trust should be updateable.	Best practices	2.4.3.25, **2.4.5.39**, **2.4.5.41**, 2.4.3.20

Table 4 Twenty-six principles for secure supply chains. IoT Security Assurance Framework requirements in bold are new contributions.

7 Conclusion

7.1 IoT supply chains are vulnerable

The words “supply chain” may mean the entire provenance of an IoT device, its subassemblies and components, the operations that have been performed on them and the environments in which they have been performed, up until any given moment. The “supply chain” thus extends through the whole deployed life of the device all the way back to specification and design of individual components, and all the way forward to final decommissioning. It is a superset of the more narrowly defined environments considered in most security literature.

Because of this, supply chain security recommendations are a superset of other security recommendations, for example those created for deployed devices. Such recommendations must identify all the confidential and trusted assets comprising IoT devices and provide for their security at every point in their journey from design to decommissioning.

Add to this the heterogeneity of IoT devices, each model of device having a unique set of assets, each asset following a unique path of design, production, delivery, and integration, and creating or following a set of generally useful recommendations begins to look like a real challenge.

It is exactly this broad scope that makes supply chains very hard to defend. Coupled with the opportunity to fan out an attack to downstream customers, it is also what makes supply chains so attractive to attackers.

7.2 Understand and secure your own supply chain

The first contribution of this paper is to simplify and help solve this problem, by developing from original contributions a general description of IoT device supply networks. This model can be easily adapted to accurately describe any specific IoT device supply network, and it is sufficiently general that useful security analysis can performed on it in its general form.

Amongst these general observations, one is that upstream branches of this IoT device supply network, in particular IC supply chains, are opaque or at least hard to influence for device OEMs. Device OEMs’ main control over these branches is via careful selection of suppliers.

A second observation is that several branches describe activities for which existing literature already provides excellent security recommendations. Several secure software development lifecycle (SDLC) publications cover software design and development activities. Numerous IoT device security publications cover the protection of deployed devices.

The one activity which is not well addressed by existing literature, but which is characteristic of IoT device supply chains, is device provisioning. The second contribution of this paper is an

original security analysis of these device provisioning operations, characterising them and systematically identifying potential attacks against them.

The third contribution of this paper is a set of twenty-six principles for securing IoT device supply chains, eleven derived from the provisioning analysis and fifteen gathered via contributions from IoTSF members and a review of existing literature. To each of these principles this paper has mapped concrete, testable security recommendations from the IoTSF's Security Assurance Framework (Assurance Framework). Twenty-nine new recommendations have been developed and contributed into the third edition of the Assurance Framework to ensure full coverage of the principles. Most recommendations necessary to implement the supply chain security principles were already present in the Assurance Framework's second edition, as expected given its role consolidating recommendations from across the IoT security community, and how "supply chain" denotes a superset of other IoT security scopes.

7.3 Be a responsible supply-chain citizen

Any organisation operating IoT devices, or relying on services provided by such organisations, should understand that they could be targeted by criminal organisations or state actors via those devices' supply chains. The more they rely on those devices, the more of a target they become. "We're not a target" is in any case never a wholly safe assumption given the way in which many supply chain attacks fan out with no concern for collateral damage.

Such organisations should assure themselves that their devices are sufficiently resistant to cyberattack both in deployed environments and in upstream supply chain environments. They should look on this security assessment as a necessary part of procurement decisions, to be considered alongside function and cost. They do not themselves have to analyse and assess the trustworthiness of the supply network lying behind their IoT devices, but they should expect their suppliers - device OEMs - to do so and to be able to say how they implement the principles.

Nobody is better placed to assess, monitor and maintain the security of IoT device supply networks through devices' deployed lifetimes than device OEMs. They decide what components to use and can do so based on the transparency and security of those components' own supply chains. They in effect design the supply chain for their own devices, through design of production processes and selection of production service providers as well as through component selection.

Securing device provisioning operations boils down to ensuring unprotected device and production assets are only ever exposed in trusted environments. This can be easily achieved by using a secure programming facility to establish a ROT, which in turn exposes secure provisioning interfaces. Used with secure provisioning tools, subsequent provisioning operations can then safely be performed even in untrusted CEM environments. Some IC

vendors now ship IoT microcontrollers with such a ROT preinstalled, lifting from OEMs the burden of creating a trusted provisioning environment themselves.

As the principal victims of overproduction, IP theft, counterfeiting, and leakage of reject parts to grey markets, OEMs may find that as in addition to protecting their customers from cyberattacks, secure provisioning provides significant revenue protection benefits.

Ultimately every supplier in the IoT device supply chain bears some responsibility, to ensure the quality and integrity of IoT device and production assets to which they have access, and on which IoT users depend.

Appendix A: **References**

Supply chain trends

[Finite-State-Huawei] *Finite State Supply Chain Assessment, Huawei Technologies Co. Ltd*; Finite State (2019); accessed at https://finitestate.io/finite-state-supply-chain-assessment/ Fascinating analysis which found no obviously malicious back doors but lots of accidental vulnerabilities, highlighting the fact that bugs and sabotage may be indistinguishable. Higher-quality code is therefore more-secure code.

[ITProPortal-big-game] Into the cyber wilderness: The rise of big game hunting; Ippolito Forni for ITProPortal (2020); accessed at https://www.itproportal.com/features/into-the-cyber-wilderness-the-rise-of-big-game-hunting/

[Schneier-IoT-security] *The Internet of Things Will Upend Our Industry*; IEEE Security & Privacy (2017); accessed at https://www.schneier.com/essays/archives/2017/03/the-internet-of-things-will-upend-our-industry.html

[Sonatype-2020] State of the Software Software Supply Chain; Sonatype (2020); accessed at https://www.sonatype.com/resources/white-paper-state-of-the-software-supply-chain-2020

Cybersecurity frameworks

General ICT

[ETSI-EN303654] *EN 303 645 V2.1.0 Cyber Security for Consumer Internet of Things: Baseline Requirements*; ETSI (2020); accessed at https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.00_30/en_303645v020100v.pdf Many of the provisions of this leading standard for securing connected consumer devices, e.g. concerning software integrity, vulnerability disclosure policy, software maintenance, require cooperation of upstream suppliers to adequately fulfil.

[ISO/IEC27001:2013] ISO/IEC 27001:2013 Information technology — Security techniques — Information security management systems — Requirements; ISO/IEC (2013), accessed at https://www.iso.org/standard/54534.html Part of the ISO/IEC 27000 family of standards which provide best-practice recommendations on information security management for IT-using organisations. The focus is on sound management of information security, much as a company should manage quality. 27001 is the core requirements document in the series. Annex A provides some suggested controls which are expanded upon in ISO/IEC 27002:2022, including supplier relationships.

[NIST-CSF] *NIST Cybersecurity Framework 1.1;* NIST (2018); accessed at https://www.nist.gov/cyberframework One of the leading manuals for securing IT operations, it has been widely adopted internationally. A number of its risk identification goals across business environment, governance, supply chain risk management, and threat protection goals across data protection and maintenance, are explicitly directed at upstream suppliers. Many or all the others could be considered implicitly so.

IoT-specific

[IoTSF-SAFv3] *IoT Security Assurance Framework, Release 3*; IoTSF (2021); accessed at https://www.iotsecurityfoundation.org/wp-content/uploads/2021/11/IoTSF-IoT-Security-Assurance-Framework-Release-3.0-Nov-2021-1.pdf Collection of practical security controls for IoT collected from the IoTSF's broad community of security professionals and IoT users. Their scope is the whole lifecycle of endpoint devices and their supply chain, back-end services, web and mobile interfaces. The controls are organised under six security goals - including secure production processes and supply chain. Five compliance classes are defined. Higher compliance classes need to be achieved in more severe threat environments. It is intended that a device should be easily assignable into an appropriate class, and that this class should then select appropriate controls from the list. IoTSF also provides a spreadsheet of the controls to aid collection of evidence, and a set of Best Practice Guides containing concise descriptions of 14 recommended practices.

Supply chain guidelines from ICT bodies

Analysis

[ENISA-supply-integrity] *Supply Chain Integrity: An overview of the ICT supply chain risks and challenges, and vision for the way forward*; ENISA (2015); accessed at https://www.enisa.europa.eu/publications/sci-2015 An early (non-defense) overview of ICT supply chain security with a focus on identifying gaps in knowledge, standards and regulation. Notes earlier (defense) work on ensuring authenticity and integrity of semiconductors.

[ENISA-supply-threats] *Threat Landscape for Supply Chain Attacks*; ENISA (2021); accessed at https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks Interesting analysis of 24 known attacks. Notes attackers increasingly shifted their attention to suppliers, who are often less well-defended than their customers. An unwelcome but unavoidable conclusion is that third party components should be both validated and tested before use.

Guidelines

[BSIMMsc] *BSIMMsc - Applying the BSIMM to the software supply chain;* Synopsys (2019); accessed at https://www.bsimm.com/about/bsimm-for-vendors.html Based on BSIMM9, identifies 22 out of BSIMM9's 116 activities on which to assess software suppliers development processes, putting into practice the idea that suppliers should be held to the same security standards as your internal development.

[ISO/IEC-20243-1] *ISO/IEC 20243-1:2018 Information technology — Open Trusted Technology ProviderTM Standard (O-TTPS) — Mitigating maliciously tainted and counterfeit products — Part 1: Requirements and recommendations*; ISO/IEC (2018); accessed at https://standards.iso.org/ittf/PubliclyAvailableStandards/index.html A sophisticated and certifiable ICT supply chain security standard for suppliers of ICT products. It also takes a whole-life view of "supply chain" from design to disposal. It attempts to generalise so that its provisions work for any (and every) supplier-acquirer link in a supply chain. Give particular attention to counterfeiting as a serious risk to end users, counterfeit products' integrity being unverifiable and, being unsupported by the original provider, liable to expose their users to significant financial and productivity losses.

[ISO/IEC-27036-1] *ISO/IEC 27036-1: Cybersecurity — Supplier relationships — Part 1: Overview and concepts*; International Standards Organisation (ISO) (2014); accessed at

https://www.iso.org/standard/82905.html The introductory part of a four-part standard covering the management of information risks involved in acquiring goods and services from suppliers. This standard does not limit itself to ICT but acquirers of ICT can require vendors to certify to ISO/IEC 27001 including additional requirements from ISO/IEC 27036-2C. Essentially it extends acquirer's security posture onto their suppliers using three groups of contractually binding requirements: information security, quality and audit. A very sound approach in principle but not always feasible in practice. Part two contains the requirements. Part three contains guidelines for specifically ICT supply chain security.

[NCSC-supply-chain] *Supply chain security guidance*; UK National Cyber Security Centre (2018); accessed at https://www.ncsc.gov.uk/collection/supply-chain-security Twelve practical and accessible principles for general use. Not intended for critical infrastructure or defence operators.

[NIST-IR8276] *NISTIR 8276 Key Practices in Cyber Supply Chain Risk Management: Observations from Industry (2021)*; NIST (2021); accessed at https://csrc.nist.gov/publications/detail/nistir/8276/final "Organizations can no longer protect themselves by simply securing their own infrastructures since their electronic perimeter is no longer meaningful; threat actors intentionally target the suppliers of more cyber-mature organizations to take advantage of the weakest link. … This document provides the ever-increasing community of digital businesses a set of Key Practices that any organization can use to manage cybersecurity risks associated with their supply chains." Based on 24 industry case studies [https://csrc.nist.gov/projects/cyber-supply-chain-risk-management/key-practices]

[NIST-SP800-161] *SP 800-161 Rev.1 Supply Chain Risk Management Practices for Federal Information Systems and Organizations*; NIST (2022); accessed at https://csrc.nist.gov/publications/detail/sp/800-161/rev-1/final Provides guidance for identifying and mitigating risks including insertion of counterfeits, unauthorized production, tampering, theft, insertion of malicious software and hardware, as well as poor manufacturing and development practices. Takes the view that an IT/OT product or service might be compromised at any point in its life via supply chain vulnerabilities, including in design, development, distribution, deployment, acquisition, maintenance, and destruction. Notes the equivalence of quality and security. Interestingly, observes that even for general IT systems "There is no gap between physical and cybersecurity."

[Synopsys-procurement-language] *Procurement Language for Supply Chain Cyber Assurance*; Synopsys (2016); accessed at http://globalforum.items-int.com/gf/gf-content/uploads/2016/10/Jarzombek-Procurement-Language-SCM.pdf See also https://www.synopsys.com/blogs/software-security/software-supply-chain-risk-management/ and https://www.linkedin.com/pulse/podcast-securing-software-supply-chain-through-part-2-van-elderen/

Supply chain guidelines from IoT bodies

[IIC-IoT-security] *Industrial Internet of Things - Volume G4: Security Framework IIC:PUB:G4:V1.0:PB:20160926*; Industrial Internet Consortium (IIC) (2016); accessed at https://www.iiconsortium.org/IISF.htm An educational document accompanying s the IIC's Industrial Internet Reference Architecture v 1.9 section 6 on the *Permeation of Trust in the IIoT System Lifecycle* is an excellent discussion of how multiple actors create and handle multiple assets through chains to custody to create IIoT devices, concluding that "The IIoT system owner/operator must trust that each prior step in the process has been implemented correctly to support the trust assumptions in the layers above him."

[ENISA-IOT-supply-chain] *Guidelines for Securing the Internet of Things*; ENISA (2020); accessed at https://www.enisa.europa.eu/publications/guidelines-for-securing-the-internet-of-things Building on [ENISA-IOT-SDLC] provides a description of the IoT supply chain, lists 22 threats (attack vectors) and 30 high-level good practices (controls).

Supply chain guidelines from industry bodies in other verticals

[TIA-SCS9001] *SCS 9001: Supply chain security standard*; Telecoms Industry Association (TIA) (2022); accessed at https://tiaonline.org/what-we-do/scs-9001-supply-chain-security-standard/ Produced after wide consultation, this is the US telecoms industry's response to emerging supply chain attacks. It was strongly motivated by national defence and regulatory interest in the issue. It is certifiable. The handbook defining 55 controls is paid-for and relatively costly, but an accompanying whitepaper is available gratis.

[MITRE-deliver-uncompromised] *Deliver Uncompromised*; MITRE (2016); Chris Nissen, John Gronager, Robert Metzger, Harvey Rishikof; accessed at https://www.mitre.org/sites/default/files/publications/pr-18-2417-deliver-uncompromised-MITRE-study-26AUG2019.pdf Develops a 15 point action plan for the US DoD, a major acquirer of ICT systems, to better protect its assets in their supply chains. Much of the action plan is relevant for other acquirers, including requirements to incorporate security measures, preference for suppliers demonstrating superior security, imposition of contractual security obligations.

[NCC-device-manufacturing] *Secure Device Manufacturing: Supply Chain Security Resilience*; NCC Group (2015); Rob Wood; accessed at https://research.nccgroup.com/wp-content/uploads/2020/07/secure-device-manufacturing-supply-chain-security-resilience-whitepaper.pdf Insightful analysis of counterfeiting in mobile phone production.

[NERC-CIP-013-2] *Cyber Security - Supply Chain Risk Management CIP-013-2*; North American Electric Reliability Corporation (NERC) (2021); accessed at https://www.nerc.com/pa/Stand/Reliability%20Standards/CIP-013-2.pdf The second generation of this regulatorily-enforceable standard for US grid operators, requiring security controls for supply chain risk management. Requires acceptance testing of third-party components.

[NCC-provisioning] *Secure Device Provisioning Best Practices: Heavy Truck Edition*; NCC Group (2019); Rob Wood; accessed at https://www.nccgroup.trust/globalassets/us-web-images/nick/secure-device-provisioning.pdf Fascinating IoT security case study, highly notable for its attention to manufacturing and maintenance processes.

[FS-ISAC-third-party] *Appropriate Software Security Control Types for Third-Party Service and Product Providers, Version 2.3*; Financial Services Information Sharing and Analysis Center (FS-ISAC) (2015), accessed at https://www.fsisac.com/resources/thirdpartysecuritycontroltypes-whitepaper "Third-party software is the new perimeter for every financial institution." Suggests that vendors should undergo the BSIMMsc maturity assessment, an audit of application security testing, an audit of OSS management, deliver an SBOM, and be made responsible for software security in contract language. Includes sample contract terms and sample vBSIMM questionnaire.

IoT provisioning problems

Keys and trust anchors

[RFC-6024-TA-management] *Trust Anchor Management Requirements*; R. Reddy (NSA), C. Wallace (Cygnacom) (2010); accessed at https://datatracker.ietf.org/doc/html/rfc6024 Protecting the confidentiality of devices' private keys is pointless if an attacker can simply tell a targeted service to trust devices of their choice. Credentials used to manage trust anchors on central services are even more valuable than individual devices' private keys.

[Richardson-mfr-key-security] *A Taxonomy of operational security considerations for manufacturer installed keys and Trust Anchors*, Michael Richardson (Sandelman Software Works) (2021); accessed at https://www.ietf.org/archive/id/draft-richardson-t2trg-idevid-considerations-06.html#name-table-of-contents Highly insightful and thorough survey of methods used by manufacturers of silicon and devices to secure i) installation of trust anchors and private keys into devices, and ii) their own private certificate-signing keys. This equivalence, often overlooked, is critical to the security of many IoT device supply chains. Describes different types of common trust anchors and identities. The work is intended to inform development of protocols concerning automated onboarding of IoT devices to local services, e.g. BRSKI, but is of general interest even for less complex scenarios.

[SAM-L11-provisioning] *Entrust provisions root identity for Microchip's IoT-ready SAM L11 Microcontrollers*; Entrust (2020); accessed at https://www.entrust.com/-/media/documentation/casestudies/microchip-entrust-hsm-cs.pdf

Initial bootloaders

[OCC-firmware-ownership] *Ownership and Control of Firmware in Open Compute Project Devices*; Open Compute Project (2018); Elaine Palmer (IBM), Tamas Visegrady (IBM), Michael Osborne (IBM), accessed at https://www.opencompute.org/documents/ibm-white-paper-ownership-and-control-of-firmware-in-open-compute-project-devices The integrity of a server in a data centre, like an IoT device, depends in large part on its initial secure bootloader. This paper defines "ownership" of such devices as the right to update their software, which is to say, the right to set the bootloader's validation key. Interesting discussion of how that right can be established, represented and in particular transferred – an interesting reprovisioning scenario.

Firmware update

[RFC9124-update-info-model] *RFC 9124 A Manifest Information Model for Firmware Updates in Internet of Things (IoT) Devices*; Brendan Moran, Hannes Tschofenig, Henk Birkholz (2022); accessed at https://datatracker.ietf.org/wg/suit/documents/ Fine development of a threat model around IoT device software updates, used to motivate RFC 9019 A Firmware Update Architecture for Internet of Things.

Securing provisioning facilities

[CC-site-certification] *Common Criteria Supporting Document Guidance, Site Certification, v1.0 rev.1 CCDB-2007-11-001*; Common Criteria (2007); accessed at https://www.commoncriteriaportal.org/files/supdocs/CCDB-2007-11-001-SiteCertificationProcessv1-0.pdf

[GSMA-SAS] *GSMA Security Accreditation Scheme*; accessed at https://www.gsma.com/security/security-accreditation-scheme/ GSMA's scheme for certifying UICC (SIM) card manufacturing sites, including their security-critical certificate installation. Covers both physical and logical security. The requirements document is available freely, the implementation guidelines document is not.

Late onboarding

[FIDO-device-onboard-wp] *FIDO Device Onboard: A specification for automated, secure IoT provisioning technology*; FIDO Alliance (2021); accessed at https://fidoalliance.org/intro-to-fido-device-onboard/ Solves a similar use case to RFC8995 except that it is aimed more at late-onboarding devices to instances of central management services running in the cloud, e.g. where a manufacturer ships a standard SKU of streetlight which needs to onboard to the appropriate city's light management system during installation. Very similar to Intel's earlier Secure Device Onboarding (SDO), from which it is developed.

[IETF-RFC8995-BRSKI] *Bootstrapping Remote Secure Key Infrastructure (BRSKI) RFC 8995*; Max Pritikin, Michael Richardson, Toerless Eckert, Michael H. Behringer, Kent Watsen, (2021); accessed at https://datatracker.ietf.org/doc/rfc8995/ Proposed standard Internet protocol for automatically installing the certificates of local control servers onto newly-connecting IoT devices, e.g. where luminaires need to connect to an in-building lighting control system. Amongst other things, provides an excellent illustration of how onboarding is a two-way operation in which devices and central services exchange certificates.

[Interact-Pro-install-guide] *Installation Guide - Interact Pro - Release 1.3*; Interact Lighting (2020); accessed at https://sme.interact-lighting.com/web/help/interact-pro/1.3/_attachments/IA-Pro_1-3_IG_R03.pdf This connected product from Signify lighting makes extensive use of installers to configure the system and application.

Method of attack trees

[Schneier-attack-trees] *Attack trees: Modeling security threats*; Bruce Schneier, Dr. Dobbs' Journal (1999); accessed at https://www.schneier.com/academic/archives/1999/12/attack_trees.html

Secure software development lifecycles (SDLCs)

Also known as **application security**.

[BSA-SDLCv1.1] *The BSA Framework for Secure Software – A New Approach to Securing the Software Lifecycle v1.1*; BSA (2020); accessed at https://www.bsa.org/reports/updated-bsa-framework-for-secure-software Excellent and progressive document which takes a risk-based approach, meaning that one of the first things it recommends is conduct of a threat analysis, and is outcomes-focused, meaning that by stating the desired outcomes without stating how to achieve them the requirements are technology-agnostic. Offers a set of requirements for developing secure software. Under Secure Development > Supply Chain there are recommendations to track third party components, that the software is easily identifiable, that is it protected from tampering. Draws on SAFECode but is different in its organisation, its broader scope and in how it references many other standards. Fully implements NIST's Secure Software Development Framework, with mapping.

[BSIMM] *Building Security In Maturity Model (BSIMM) version 11*; Synopsys (2021); accessed at https://www.bsimm.com Completely empirical annual report documenting current secure software development practices across a wide membership.

[ISO/IEC-27034-1] *ISO/IEC 27034-1:2011 Information technology — Security techniques — Application security — Part 1: Overview and concepts;* ISO/IEC (2011); accessed at

https://www.iso.org/standard/44378.html Closely aligned with ISO/IEC 27005 for information security risk management, this is part one of a seven-part standard describing how an organisation can set up effective, efficient processes for building security into applications and application development. It explicitly is not a standard for secure applications or application development, only of how those can be achieved. It is aimed at those specifying, designing, developing or procuring software applications and is notable in this context for how it makes no distinction between in-house and 3rd-party developments.

[NIST-SP800-218] *SP 800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities*; NIST (2022); Murugiah Souppaya (NIST), Karen Scarfone (Scarfone Cybersecurity), Donna Dodson; accessed at https://csrc.nist.gov/publications/detail/sp/800-218/final

[OWASP-SAMMv2] *Software Assurance Maturity Model (SAMM) version 2*; Open Web Application Security Project (OWASP) Foundation (2020); accessed at https://owaspsamm.org Intended to give organisations developing web applications a to analyse and improve their web application security. Covers governance, design, implementation, verification and operations. Presents 15 security practices each with three levels of maturity. Supplier security is considered explicitly under the Security Requirements and Secure Build practices.

[PCI-SSFv1.1] *Software Security Framework version 1.1*; Payment Card Industry (PCI) (2021); accessed at https://www.pcisecuritystandards.org/document_library?category=sware_sec#results Security requirements for card payment software, and its development, aimed at protecting the integrity and confidentiality of payment transactions and data. All vendors of card payment software must adhere to this standard. Like BSA it used an objective-based approach to requirements, instead of requiring a checklist of controls. Draws on SAFECode's Fundamental Practices for Secure Software Development [SAFEcode-SSD] and NIST's Cybersecurity Framework [NIST-CSF]. Certification requires that third-party components are inventoried, properly used, correctly functional, monitored for vulnerabilities.

[SAFEcode-SSD] *Fundamental Practices for Secure Software Development - Essential Elements of a Secure Development Lifecycle Program, Third Edition*; SAFEcode (2018); accessed at https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf Presents a collection of used practices, similar to BSIMM's approach, relating to secure design, development and testing of software. This version added management of vulnerabilities and third-party components, building on SAFEcode's 2017 paper *Managing Security Risks Inherent in the Use of Third-party Components*. Very creditably motivates every recommendation with examples of real vulnerabilities from the Common Weakness Enumeration (CWE) database.

Secure development lifecycles for IoT

[CSI-firmware] *Secure Firmware Development Best Practices version 1.1*; Cloud Security Industry Summit, Supply Chain Technical Working Group (2019); accessed at https://ogi-cdn.s3.us-east-2.amazonaws.com/csis/firmware-security-best-practices-v1.1.pdf Contains a state-of-the-art discussion of secure coding practices as well as an interesting discussion of the meaning of signatures, specifically in the context of firmware signing.

[ENISA-IOT-SDLC] *Good Practices for Security of IoT - Secure Software Development Lifecycle*; ENISA (2109); accessed at https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1 Very well-researched SDLC with an IoT focus. See in particular section 4.2.2.1 Third-Party Management.

[IIC-software-trustworthiness] *Software Trustworthiness Best Practices - An Industrial Internet Consortium White Paper, Version 1.0*; Industrial Internet Consortium (2020); Marcellus Buchheit (Wibu-Systems), Mark Hermeling (GrammaTech), Frederick Hirsch (Fujitsu), Bob Martin (MITRE), Simon Rix (Irdeto); accessed at https://www.iiconsortium.org/pdf/Software_Trustworthiness_Best_Practices_Whitepaper_2020_03_23.pdf Excellent review of secure software development practices with a focus on embedded systems.

[ISA/IEC-62443-4-1] *IEC 62443-4-1 Security for industrial automation and control systems - Part 4-1: Secure product development lifecycle requirements*; International Electrotechnical Commission (IEC) (2018); accessed at https://webstore.iec.ch/publication/33615 The popular ISA/IEC 62443 Security for Industrial Automation and Control Systems (IACS) family does for operational technology (OT) what the ISO 27000 family does for information technology (IT). This part describes the development and maintenance processes to be used by IACS component vendors including security requirements definition, secure design, secure implementation (including coding guidelines), verification and validation, defect management, patch management and product end-of-life.

Software bills of materials (SBOMs)

[CycloneDX] *CycloneDX* v1.4; OWASP (2022), accessed at https://cyclonedx.org An XML SBOM specification designed for vulnerability identification, license compliance, and outdated component analysis use cases. The project provides tools to generate CycloneDX SBOMs in many language ecosystems. Originated in OWASP Dependency-Track, an OSS Software Composition Analysis (SCA) tool. V1.4 adds the Vulnerability Exploitability Exchange (VEX) feature, designed to automate communication of vulnerabilities and their exploitability for software defined in a bill of materials.

[SPDX] *Software Package Data Exchange (SPDX)*; Linux Foundation (2021); accessed at https://spdx.dev An SBOM format for communicating the components, licenses and copyrights associated with software packages. Largely aimed at OSS it relies on placement of SPDX tags in source files. Maintains a list of standardised license IDs to be included in files and/or package READMEs. Its main use case is compliance with OSS licences, by aiding automated inventory. Also published as ISO/IEC 5962:2021.

[SWID] *ISO/IEC 19770-2:2015 Information technology — IT asset management — Part 2: Software identification tag (SWID)*; ISO/IEC (2015); accessed at https://www.iso.org/standard/65666.html A SWID Tag document is an SBOM identifying a software product, its version, the organizations and individuals involved in producing and developing it, the artifacts comprising it. The original use case was to track usage of paid-for software by organisations, e.g. for billing per instance. Later, software asset tracking for cybersecurity purposes was added. NIST has produced NISTIR 8060 Guidelines for the Creation of Interoperable Software Identification (SWID) Tags to help people use SWID for cybersecurity purposes. However, it seems lightly adopted.

Hardware attacks & controls

IoT device hardware supply chains

[ENISA-hardware-threats] *Hardware Threat Landscape and Good Practice Guide*; ENISA (2017); accessed at https://www.enisa.europa.eu/publications/hardware-threat-landscape Surveys hardware attacks on embedded devices but excluding ICs and supply-chain (meaning pre-deployment) attacks. Provides a good list of hardware attacks to use in attack trees.

[SAE-counterfeit] *Counterfeit Electrical, Electronic, and Electromechanical (EEE) Parts; Avoidance, Detection, Mitigation, and Disposition AS5553D*; Society of Automotive Engineers International (2021); accessed at https://www.sae.org/standards/content/as5553d/ Intended to be used with a higher-level quality standard, this document puts requirements around the purchasing process (specify, check, verify) but also covers supplier management and what to do when counterfeits are discovered.

IC hardware supply chains

[Areno-IC-supply-chain-threats] Supply Chain Threats Against Integrated Circuits; Matthew Areno (Intel) (2020); accessed at https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/supply-chain-threats-v1.pdf Brief but informative survey of supply-chain threats including related studies and demonstrations and documented attacks.

[Huang-Counterfeit-ICs] *On Counterfeit Chips in US Military Hardware*; Bunnie:Studios (Bunnies' Blog); Andrew "Bunnie" Huang (2011), accessed at https://www.bunniestudios.com/blog/?p=2037 Characteristically insightful discussion on the sources and detection of counterfeit ICs in the context of US defense, which was and perhaps still is the security community most concerned about securing the IC supply chain.

[Huang-IC-implants] *Supply Chain Security: If I were a Nation State...*; talk by Andrew "bunnie" Huang at BlueHat IL 2019, accessed at https://www.youtube.com/watch?v=RqQhWitJ1As Highly informative look at how ICs' integrity might be compromised in distribution. Useful comparison of classes of IC supply chain attacks, on axes of how hard they are to spot vs how hard they are to execute.

Appendix B: Attack Trees

The method of attack trees was first widely shared by noted cybersecurity expert Bruce Schneier [3]. The method begins by identifying potential adversaries, then their top-level objectives. Using the understanding of manufacturing systems and operations developed earlier in this document, a list of necessary preconditions is created for each objective, at least one of which must be true in order to realise the objective. Each precondition can likewise have its own preconditions. Each chain of conditions constitutes a potential attack that can be defended against by making any link in the chain impossible, or at least more difficult, to achieve.

Where multiple preconditions at the same level must be true to make their root precondition true, they can be listed in an [AND] relationship.

Where attack trees share a common subtree, it is indicated in square brackets and developed separately.

Adversaries pursuing their own goals have many potential lines of attack, only some of which go through device manufacturing operations. In this analysis, any attacks outside of manufacturing operations are noted in light grey text for context but not developed further.

A. Steal from OEMs of IoT devices

1. Produce identical devices for sale on own account
    1.1. Produce counterfeit devices on black production line
        1.1.1. AND See [Extract OEM's software IP]
        1.1.2. AND Obtain use of OEM's production tools
    1.2. Produce additional devices on legitimate production line
        1.2.1. Produce additional devices
        1.2.2. Untruthfully claim low yield
    1.3. Recondition manufacturing discards for sale
2. Introduce grey market ICs into legitimate production
3. Sell access to OEMs' software IP / production tools
4. Hold production to ransom by denying access to remotely located production assets

B. Disrupt or monitor operators of IoT devices

1. Deploy attacker's devices into operational situations
    1.1. Onboard devices loaded with attacker's firmware
        1.1.1. Obtain use of onboarding credentials to onboard attacker's device IDs to central service
            1.1.1.1. Intercept the onboarding credentials *en route* to the provisioning tool
            1.1.1.2. Extract the onboarding credentials from the provisioning tool

[3] 1999, Bruce Schneier, Dr Dobb's Journal, Attack Trees (see https://www.schneier.com/academic/archives/1999/12/attack_trees.html)

1.1.2. Identify attacker's devices as legitimate to the onboarding entity
1.1.2.1. Introduce attacker's devices into legitimate production process
1.1.2.2. Introduce attackers' device IDs into production logs
1.1.3. Obtain an OEM CA key and issue certificates to attacker's devices
1.1.3.1. Access OEM's parent CA key to create a new device-signing key
1.1.3.2. Intercept the device signing key *en route* to the provisioning tool
1.1.3.3. Extract the device signing key from the provisioning tool
1.1.4. Provision attacker's CA certificate into IOT service (where IOT service relies on that to validate devices)
1.2. Clone keys and certificates of genuine devices to devices with attacker's firmware
1.2.1. See [Obtain private key]
2. Run attacker's code on deployed devices
2.1. Upgrade devices over the air to attacker's firmware
2.1.1. Obtain OEM's firmware signing key
2.1.2. Have OEM sign attacker's firmware
2.1.3. Alter OEM build artefacts
2.1.4. Alter OEM code repositories
2.2. Provision attacker's software onto devices at manufacture (bootloader or firmware trust anchor)
2.2.1. Modify provisioned data during preparation by OEM
2.2.2. Modify provisioned data during transfer from OEM to provisioning tool
2.2.3. Modify provisioned data stored in the provisioning tool
2.2.4. Modify provisioned data as it is programmed onto devices
2.2.5. Modify provisioned data after provisioning
2.3. Develop remote exploit against IOT devices
2.3.1. See [Extract OEM's software IP]
3. Terminate device connections on attacker-controlled central service
3.1. Obtain legitimate service's private identity key
3.2. Obtain a certificate for the attacker-controlled service that is trusted by devices
3.2.1. Have the legitimate service's CA sign the attacker's public key
3.2.2. Obtain the legitimate service's CA's private identity key
3.2.3. Install the trust anchor certificate of the attacker's CA on the device
3.3. Use fault injection to bypass devices' server certificate check
4. Replay stale data
5. Control environmental inputs (for sensor devices)
6. Block data sent to or from devices

The following attack trees are sub-trees that appear in multiple places in the main trees above:

C. [Obtain private key]

1. Read key physically from device
1.1. Access debug port
1.2. Depackage and etch IC, identify and examine fuses/cells
1.3. Apply side-channel attacks e.g., SPA, DPA, DTA, DFA
2. Obtain key from externally generated key pool

2.1. Copy keys at key generator
2.2. Copy keys during transport to where keys will be used
3. Guess (brute-force) keys generated on-board devices
3.1. Modify masks for on-board TRNG so that it is more predictable
3.2. Modify a specific on-board TRNG so that it is more predictable
3.3. Constrain external entropy required for PRNG initial conditions

D. [Extract OEM's software IP]

1. Reverse-engineer device firmware image
1.1. Intercept OEM's unencrypted software IP *en route* to factory
1.2. Access OEM's unencrypted software IP while stored in the factory
1.3. Tap unencrypted programming interfaces
1.4. Present insecure targets for provisioning
1.4.1. Disable or prevent activation of data read and write protections on target
1.4.2. Create fake target, programmable by provisioner, readable by attacker
1.5. Decrypt firmware encrypted for device by OEM for OTA delivery
1.5.1. [Obtain ephemeral encryption key] Note well: the decryptor in this case is the target device, the encryptor is the OEM, the legitimate trust anchor is the public part of the OEM's firmware signing key
1.6. Access unencrypted OEM build artefacts
2. Obtain OEM source code
2.1. Access OEM code repositories
3. Read IP physically from device
3.1. Read out software via debug or programming port

Appendix C: Secure provisioning solutions

Examples of real-life provisioning tools and production environments.

B.1 Production line PC in a CEM facility

Small to medium-scale production lines often provision devices using a production line PC. The PC is connected to a debugger/programmer appropriate to the target IC, and devices are connected using a bed of nails jig or a service socket. The OEM/ODM develops software to execute the provisioning sequence, provide an operator interface and generate logs for periodic delivery back to them. The software includes all necessary provisioning assets including certificate signing keys and can be accessed by many individuals on site.

It is not uncommon for other functions such as manufacturing tests to be integrated in this station. Conversely it is quite possible for provisioning steps to be split across multiple stations, for example a first station may be used to program a standard image, which exposes a configuration interface via a serial line that a second station uses to load custom settings for different customers.

Typically, the OEM or ODM hands over manufacturing specifications, provisioning assets and specialised production tools for manufacturing test and provisioning, and the CEM undertakes to deliver a certain amount of product. OEMs and ODMs can select CEMs on the basis of quality and security certifications, and they likely help set up and debug production processes, but they do not otherwise have much control over how their assets are used and protected. Most CEMs operate on very thin margins and their main focus and expertise is on cost control, not information security.

Some CEMs have been known to increase their margins by stealing from their customers in various ways. Although this is often detected eventually, they are protected from serious consequences by the inability of OEMs and ODMs to prove complicity.

B.2 Remote certificate authority

Signing devices into a certificate chain of trust is a popular way of onboarding them to central services in volume. Because anyone with access to the required CA key can onboard devices it must be kept secret and use of it must be strictly limited to trusted parties. To serve this need major commercial certificate authorities now offer IoT device certificates as a service, offering OEMs and ODMs the opportunity to outsource the secure management of their CA keys and benefit from associated PKI services, including certificate renewal and revocation and widely trusted root certificates. Note that that the CA signing key is the only production asset protected by this arrangement. Authentication and programming of devices before they are presented for certification must be entrusted to either a general CEM or a specialist secure programming service.

B.3 Remote production counter

Many OEMs and ODMs have suffered from the presence on the market of counterfeits of their products. In many cases those “counterfeits” have been produced on their own production lines, for sale by CEMs on their own account. One approach to this problem has been to have the device software check for the presence of a validly signed certificate containing the IC unique hardware ID. During provisioning the unique hardware ID is extracted from each device and sent offsite to the OEM or ODM to be signed into that certificate. This arrangement permits the OEM/ODM to monitor and control production volumes, preventing overproduction. However, the only production asset it protects is, in effect, the production counter.

B.4 Production line HSM

Mobile subscriber identity modules (SIM cards), cable TV decoders and games consoles all rely on unique device certificates for revenue protection purposes. Manufacturers of these kinds of devices must guard the extremely valuable signing keys jealously, and they do it by keeping them safely in production-line HSMs. On the production line automated provisioning equipment forms a certificate signing request for each target, has the HSM issue a device certificate, and installs the certificate back onto the target.

As with remote certificate authorities, although the assurance credentials are protected in the HSM the provisioned assets, for example the smartcard application and configuration data, are not. Their authenticity, integrity and confidentiality, as well as the authenticity of the target ICs, is protected by operational and physical security in the production facility. Certifiable standards for such facilities exist, including GSMA SAS [4] and the Common Criteria site certification process [5].

B.5 Secure programming facility

Some IC distributors offer high-volume programming services, using specialist equipment to program multiple chips at a time. Some are upgrading the physical and information security of these services and supporting more complex provisioning sequences to better serve IoT OEMs and ODMs. Although such facilities handle bare ICs before they are assembled into PCBs and devices, they can perform most provisioning operations. The only thing they should not do is onboard completed devices, e.g., by signing them into an appropriate certificate chain of trust.

[4] GSMA Security Accreditation Scheme (see https://www.gsma.com/security/security-accreditation-scheme/)

[5] 2007, Common Criteria Supporting Document Guidance, Site Certification, v1.0 rev.1 CCDB-2007-11-001

Using a secure programming facility gives OEMs' and ODMs' devices and IP better protection against cyber-attacks launched via manufacturing operations. Such facilities can justify investments in physical and informational security measures more easily than can generalist CEMs. Being highly automated they can be located cost effectively anywhere in the world, allowing OEMs to manage their exposure to nation state actors who may use national security laws to require the cooperation of manufacturing sites. And because these facilities' value-add is purely in provisioning they lack the incentives that CEMs may have to participate in grey markets and overproduce.

B.6 Pre-provisioned roots of trust with secure provisioning systems

IC vendors increasingly offer products with specialised roots of trust specifically designed for secure provisioning. Pre-provisioned with unique device certificates issued by the IC vendor, these ROTs expose secure provisioning interfaces. By placing secure programming equipment on the device production line, itself hardened against unauthorised inspection and modification, OEMs and ODMs can use these interfaces to authenticate devices and provision them without ever exposing their IP, signing keys, or trusted device assets in the factory.

In practice the capabilities of such systems vary. Those offered by IC vendors focus on preventing overproduction and IP theft rather than guaranteeing the integrity of IoT devices. In consequence they protect the integrity and confidentiality of software images and production count, even in untrusted CEM facilities, but do not guarantee the integrity of assurances such as production logs, device certificates and onboarding. Also, it must be noted that although the CEM is removed from the trust base of the device software the IC vendor is thereby added to it. While there is no guarantee that IC vendors will secure their provisioning operations better than a CEM would, OEMs and ODMs may choose to place their trust in the fact that the IC vendor has stronger incentives to do so and can amortise their costs over a larger volume of products. In any case the IC vendor is already highly trusted by the device OEM or ODM so entrusting them with this provisioning step is a small extension.

B.7 Integrated secure provisioning solutions

Creating a provisioning system is an established part of every new embedded product development. The cost and difficulty of creating these provisioning systems is increasing as IoT products require increasingly complex processes, which as we have seen need to be secured to at least a similar level as the devices themselves. Fortunately, commercial off-the-shelf solutions are available from well-known vendors, aimed specifically at protecting provisioning assets in production environments. Investing in such a system allows OEMs and ODMs to protect their customers from cyber-attacks launched via production sites, and at the same time protect themselves from revenue diversion by untrustworthy CEMs, while reducing project risk and freeing engineering resources to focus on differentiating application features. Feature sets vary but the core proposition of these solutions is a turnkey solution

for securely provisioning devices, extending from development teams to target devices on the production line.

Appendix D: How to use this paper

C.1 Business leaders

Are you a C-level executive in an organisation that uses or supplies IoT?

1. Familiarise yourself with the nature and risks of supply chain attacks described in this paper.
2. Ascertain your organisation's role in the IT supply chain
3. Ensure responsibility for supply chain security is being properly assigned and managed in your organisation

C.2 Production engineers

Are you a production engineer looking to understand the relative benefits of different secure production solutions?

1. Familiarise yourself with the case studies in Secure provisioning solutions
2. Evaluate the solutions being offered to you against the IoT Security Assurance Framework v3 requirements
3. Select a combination that ticks all the boxes

C.3 Security engineers

Are you a security professional or engineer looking to secure your supply chain rigorously but cost-effectively?

1. Familiarise yourself with the provisioning steps in this paper.
2. Describe your own supply chain in the style of Figure 3. Take care to identify the different environments and the operations performed in each.
3. Identify at each step which assets are exposed and to whom. This completes the characterisation of your supply chain.
4. Build your own attack tree, identifying potential attacks against your supply chain. It will be like that in References
5. Supply chain trends

[Finite-State-Huawei] *Finite State Supply Chain Assessment, Huawei Technologies Co. Ltd*; Finite State (2019); accessed at https://finitestate.io/finite-state-supply-chain-assessment/ Fascinating analysis which found no obviously malicious back doors but lots of accidental vulnerabilities, highlighting the fact that bugs and sabotage may be indistinguishable. Higher-quality code is therefore more-secure code.

[ITProPortal-big-game] Into the cyber wilderness: The rise of big game hunting; Ippolito Forni for ITProPortal (2020); accessed at https://www.itproportal.com/features/into-the-cyber-wilderness-the-rise-of-big-game-hunting/

[Schneier-IoT-security] *The Internet of Things Will Upend Our Industry*; IEEE Security & Privacy (2017); accessed at https://www.schneier.com/essays/archives/2017/03/the-internet-of-things-will-upend-our-industry.html

[Sonatype-2020] State of the Software Software Supply Chain; Sonatype (2020); accessed at https://www.sonatype.com/resources/white-paper-state-of-the-software-supply-chain-2020

Cybersecurity frameworks

General ICT

[ETSI-EN303654] *EN 303 645 V2.1.0 Cyber Security for Consumer Internet of Things: Baseline Requirements*; ETSI (2020); accessed at https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.00_30/en_303645v020100v.pdf Many of the provisions of this leading standard for securing connected consumer devices, e.g. concerning software integrity, vulnerability disclosure policy, software maintenance, require cooperation of upstream suppliers to adequately fulfil.

[ISO/IEC27001:2013] ISO/IEC 27001:2013 Information technology — Security techniques — Information security management systems — Requirements; ISO/IEC (2013), accessed at https://www.iso.org/standard/54534.html Part of the ISO/IEC 27000 family of standards which provide best-practice recommendations on information security management for IT-using organisations. The focus is on sound management of information security, much as a company should manage quality. 27001 is the core requirements document in the series. Annex A provides some suggested controls which are expanded upon in ISO/IEC 27002:2022, including supplier relationships.

[NIST-CSF] *NIST Cybersecurity Framework 1.1;* NIST (2018); accessed at https://www.nist.gov/cyberframework One of the leading manuals for securing IT operations, it has been widely adopted internationally. A number of its risk identification goals across business environment, governance, supply chain risk management, and threat protection goals across data protection and maintenance, are explicitly directed at upstream suppliers. Many or all the others could be considered implicitly so.

IoT-specific

[IoTSF-SAFv3] *IoT Security Assurance Framework, Release 3*; IoTSF (2021); accessed at https://www.iotsecurityfoundation.org/wp-content/uploads/2021/11/IoTSF-IoT-Security-Assurance-Framework-Release-3.0-Nov-2021-1.pdf Collection of practical security controls for IoT collected from the IoTSF's broad community of security professionals and IoT users. Their scope is the whole lifecycle of endpoint devices and their supply chain, back-end services, web and mobile interfaces. The controls are organised under six security goals - including secure production processes and supply chain. Five compliance classes are defined. Higher compliance classes need to be achieved in more severe threat environments. It is intended that a device should be easily assignable into an appropriate class, and that this class should then select appropriate controls from the list. IoTSF also provides a spreadsheet of the controls to aid collection of evidence, and a set of Best Practice Guides containing concise descriptions of 14 recommended practices.

Supply chain guidelines from ICT bodies

Analysis

[ENISA-supply-integrity] *Supply Chain Integrity: An overview of the ICT supply chain risks and challenges, and vision for the way forward*; ENISA (2015); accessed at https://www.enisa.europa.eu/publications/sci-2015 An early (non-defense) overview of ICT supply chain security with a focus on identifying gaps in knowledge, standards and regulation. Notes earlier (defense) work on ensuring authenticity and integrity of semiconductors.

[ENISA-supply-threats] *Threat Landscape for Supply Chain Attacks*; ENISA (2021); accessed at https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks Interesting analysis of 24 known attacks. Notes attackers increasingly shifted their attention to suppliers, who are often less well-defended than their customers. An unwelcome but unavoidable conclusion is that third party components should be both validated and tested before use.

Guidelines

[BSIMMsc] *BSIMMsc - Applying the BSIMM to the software supply chain;* Synopsys (2019); accessed at https://www.bsimm.com/about/bsimm-for-vendors.html Based on BSIMM9, identifies 22 out of BSIMM9's 116 activities on which to assess software suppliers development processes, putting into practice the idea that suppliers should be held to the same security standards as your internal development.

[ISO/IEC-20243-1] *ISO/IEC 20243-1:2018 Information technology — Open Trusted Technology ProviderTM Standard (O-TTPS) — Mitigating maliciously tainted and counterfeit products — Part 1: Requirements and recommendations*; ISO/IEC (2018); accessed at https://standards.iso.org/ittf/PubliclyAvailableStandards/index.html A sophisticated and certifiable ICT supply chain security standard for suppliers of ICT products. It also takes a whole-life view of "supply chain" from design to disposal. It attempts to generalise so that its provisions work for any (and every) supplier-acquirer link in a supply chain. Give particular attention to counterfeiting as a serious risk to end users, counterfeit products' integrity being unverifiable and, being unsupported by the original provider, liable to expose their users to significant financial and productivity losses.

[ISO/IEC-27036-1] *ISO/IEC 27036-1: Cybersecurity — Supplier relationships — Part 1: Overview and concepts*; International Standards Organisation (ISO) (2014); accessed at https://www.iso.org/standard/82905.html The introductory part of a four-part standard covering the management of information risks involved in acquiring goods and services from suppliers. This standard does not limit itself to ICT but acquirers of ICT can require vendors to certify to ISO/IEC 27001 including additional requirements from ISO/IEC 27036-2C. Essentially it extends acquirer's security posture onto their suppliers using three groups of contractually binding requirements: information security, quality and audit. A very sound approach in principle but not always feasible in practice. Part two contains the requirements. Part three contains guidelines for specifically ICT supply chain security.

[NCSC-supply-chain] *Supply chain security guidance*; UK National Cyber Security Centre (2018); accessed at https://www.ncsc.gov.uk/collection/supply-chain-security Twelve practical and accessible principles for general use. Not intended for critical infrastructure or defence operators.

[NIST-IR8276] *NISTIR 8276 Key Practices in Cyber Supply Chain Risk Management: Observations from Industry (2021)*; NIST (2021); accessed at https://csrc.nist.gov/publications/detail/nistir/8276/final

“Organizations can no longer protect themselves by simply securing their own infrastructures since their electronic perimeter is no longer meaningful; threat actors intentionally target the suppliers of more cyber-mature organizations to take advantage of the weakest link. … This document provides the ever-increasing community of digital businesses a set of Key Practices that any organization can use to manage cybersecurity risks associated with their supply chains.” Based on 24 industry case studies [https://csrc.nist.gov/projects/cyber-supply-chain-risk-management/key-practices]

[NIST-SP800-161] *SP 800-161 Rev.1 Supply Chain Risk Management Practices for Federal Information Systems and Organizations*; NIST (2022); accessed at https://csrc.nist.gov/publications/detail/sp/800-161/rev-1/final Provides guidance for identifying and mitigating risks including insertion of counterfeits, unauthorized production, tampering, theft, insertion of malicious software and hardware, as well as poor manufacturing and development practices. Takes the view that an IT/OT product or service might be compromised at any point in its life via supply chain vulnerabilities, including in design, development, distribution, deployment, acquisition, maintenance, and destruction. Notes the equivalence of quality and security. Interestingly, observes that even for general IT systems “There is no gap between physical and cybersecurity.”

[Synopsys-procurement-language] *Procurement Language for Supply Chain Cyber Assurance*; Synopsys (2016); accessed at http://globalforum.items-int.com/gf/gf-content/uploads/2016/10/Jarzombek-Procurement-Language-SCM.pdf See also https://www.synopsys.com/blogs/software-security/software-supply-chain-risk-management/ and https://www.linkedin.com/pulse/podcast-securing-software-supply-chain-through-part-2-van-elderen/

Supply chain guidelines from IoT bodies

[IIC-IoT-security] *Industrial Internet of Things - Volume G4: Security Framework IIC:PUB:G4:V1.0:PB:20160926*; Industrial Internet Consortium (IIC) (2016); accessed at https://www.iiconsortium.org/IISF.htm An educational document accompanying s the IIC's Industrial Internet Reference Architecture v 1.9 section 6 on the *Permeation of Trust in the IIoT System Lifecycle* is an excellent discussion of how multiple actors create and handle multiple assets through chains to custody to create IIoT devices, concluding that "The IIoT system owner/operator must trust that each prior step in the process has been implemented correctly to support the trust assumptions in the layers above him."

[ENISA-IOT-supply-chain] *Guidelines for Securing the Internet of Things*; ENISA (2020); accessed at https://www.enisa.europa.eu/publications/guidelines-for-securing-the-internet-of-things Building on [ENISA-IOT-SDLC] provides a description of the IoT supply chain, lists 22 threats (attack vectors) and 30 high-level good practices (controls).

Supply chain guidelines from industry bodies in other verticals

[TIA-SCS9001] *SCS 9001: Supply chain security standard*; Telecoms Industry Association (TIA) (2022); accessed at https://tiaonline.org/what-we-do/scs-9001-supply-chain-security-standard/ Produced after wide consultation, this is the US telecoms industry’s response to emerging supply chain attacks. It was strongly motivated by national defence and regulatory interest in the issue. It is certifiable. The handbook defining 55 controls is paid-for and relatively costly, but an accompanying whitepaper is available gratis.

[MITRE-deliver-uncompromised] *Deliver Uncompromised*; MITRE (2016); Chris Nissen, John Gronager, Robert Metzger, Harvey Rishikof; accessed at https://www.mitre.org/sites/default/files/publications/pr-18-2417-deliver-uncompromised-MITRE-study-26AUG2019.pdf Develops a 15 point action plan for the US DoD, a major acquirer of ICT systems, to better protect its assets in their supply chains. Much of the action plan is relevant for other acquirers, including requirements to incorporate security measures, preference for suppliers demonstrating superior security, imposition of contractual security obligations.

[NCC-device-manufacturing] *Secure Device Manufacturing: Supply Chain Security Resilience*; NCC Group (2015); Rob Wood; accessed at https://research.nccgroup.com/wp-content/uploads/2020/07/secure-device-manufacturing-supply-chain-security-resilience-whitepaper.pdf Insightful analysis of counterfeiting in mobile phone production.

[NERC-CIP-013-2] *Cyber Security - Supply Chain Risk Management CIP-013-2*; North American Electric Reliability Corporation (NERC) (2021); accessed at https://www.nerc.com/pa/Stand/Reliability%20Standards/CIP-013-2.pdf The second generation of this regulatorily-enforceable standard for US grid operators, requiring security controls for supply chain risk management. Requires acceptance testing of third-party components.

[NCC-provisioning] *Secure Device Provisioning Best Practices: Heavy Truck Edition*; NCC Group (2019); Rob Wood; accessed at https://www.nccgroup.trust/globalassets/us-web-images/nick/secure-device-provisioning.pdf Fascinating IoT security case study, highly notable for its attention to manufacturing and maintenance processes.

[FS-ISAC-third-party] *Appropriate Software Security Control Types for Third-Party Service and Product Providers, Version 2.3*; Financial Services Information Sharing and Analysis Center (FS-ISAC) (2015), accessed at https://www.fsisac.com/resources/thirdpartysecuritycontroltypes-whitepaper “Third-party software is the new perimeter for every financial institution.” Suggests that vendors should undergo the BSIMMsc maturity assessment, an audit of application security testing, an audit of OSS management, deliver an SBOM, and be made responsible for software security in contract language. Includes sample contract terms and sample vBSIMM questionnaire.

IoT provisioning problems

Keys and trust anchors

[RFC-6024-TA-management] *Trust Anchor Management Requirements*; R. Reddy (NSA), C. Wallace (Cygnacom) (2010); accessed at https://datatracker.ietf.org/doc/html/rfc6024 Protecting the confidentiality of devices’ private keys is pointless if an attacker can simply tell a targeted service to trust devices of their choice. Credentials used to manage trust anchors on central services are even more valuable than individual devices’ private keys.

[Richardson-mfr-key-security] *A Taxonomy of operational security considerations for manufacturer installed keys and Trust Anchors*, Michael Richardson (Sandelman Software Works) (2021); accessed at https://www.ietf.org/archive/id/draft-richardson-t2trg-idevid-considerations-06.html#name-table-of-contents Highly insightful and thorough survey of methods used by manufacturers of silicon and devices to secure i) installation of trust anchors and private keys into devices, and ii) their own private certificate-signing keys. This equivalence, often overlooked, is critical to the security of many IoT device supply chains. Describes different types of common trust anchors and identities. The work is intended to inform

development of protocols concerning automated onboarding of IoT devices to local services, e.g. BRSKI, but is of general interest even for less complex scenarios.

[SAM-L11-provisioning] *Entrust provisions root identity for Microchip's IoT-ready SAM L11 Microcontrollers*; Entrust (2020); accessed at https://www.entrust.com/-/media/documentation/casestudies/microchip-entrust-hsm-cs.pdf

Initial bootloaders

[OCC-firmware-ownership] *Ownership and Control of Firmware in Open Compute Project Devices*; Open Compute Project (2018); Elaine Palmer (IBM), Tamas Visegrady (IBM), Michael Osborne (IBM), accessed at https://www.opencompute.org/documents/ibm-white-paper-ownership-and-control-of-firmware-in-open-compute-project-devices The integrity of a server in a data centre, like an IoT device, depends in large part on its initial secure bootloader. This paper defines "ownership" of such devices as the right to update their software, which is to say, the right to set the bootloader's validation key. Interesting discussion of how that right can be established, represented and in particular transferred – an interesting reprovisioning scenario.

Firmware update

[RFC9124-update-info-model] *RFC 9124 A Manifest Information Model for Firmware Updates in Internet of Things (IoT) Devices*; Brendan Moran, Hannes Tschofenig, Henk Birkholz (2022); accessed at https://datatracker.ietf.org/wg/suit/documents/ Fine development of a threat model around IoT device software updates, used to motivate RFC 9019 A Firmware Update Architecture for Internet of Things.

Securing provisioning facilities

[CC-site-certification] *Common Criteria Supporting Document Guidance, Site Certification, v1.0 rev.1 CCDB-2007-11-001*; Common Criteria (2007); accessed at https://www.commoncriteriaportal.org/files/supdocs/CCDB-2007-11-001-SiteCertificationProcessv1-0.pdf

[GSMA-SAS] *GSMA Security Accreditation Scheme*; accessed at https://www.gsma.com/security/security-accreditation-scheme/ GSMA's scheme for certifying UICC (SIM) card manufacturing sites, including their security-critical certificate installation. Covers both physical and logical security. The requirements document is available freely, the implementation guidelines document is not.

Late onboarding

[FIDO-device-onboard-wp] *FIDO Device Onboard: A specification for automated, secure IoT provisioning technology*; FIDO Alliance (2021); accessed at https://fidoalliance.org/intro-to-fido-device-onboard/ Solves a similar use case to RFC8995 except that it is aimed more at late-onboarding devices to instances of central management services running in the cloud, e.g. where a manufacturer ships a standard SKU of streetlight which needs to onboard to the appropriate city's light management system during installation. Very similar to Intel's earlier Secure Device Onboarding (SDO), from which it is developed.

[IETF-RFC8995-BRSKI] *Bootstrapping Remote Secure Key Infrastructure (BRSKI) RFC 8995*; Max Pritikin, Michael Richardson, Toerless Eckert, Michael H. Behringer, Kent Watsen, (2021); accessed at https://datatracker.ietf.org/doc/rfc8995/ Proposed standard Internet protocol for automatically installing

the certificates of local control servers onto newly-connecting IoT devices, e.g. where luminaires need to connect to an in-building lighting control system. Amongst other things, provides an excellent illustration of how onboarding is a two-way operation in which devices and central services exchange certificates.

[Interact-Pro-install-guide] *Installation Guide - Interact Pro - Release 1.3*; Interact Lighting (2020); accessed at https://sme.interact-lighting.com/web/help/interact-pro/1.3/_attachments/IA-Pro_1-3_IG_R03.pdf This connected product from Signify lighting makes extensive use of installers to configure the system and application.

Method of attack trees

[Schneier-attack-trees] *Attack trees: Modeling security threats*; Bruce Schneier, Dr. Dobbs' Journal (1999); accessed at https://www.schneier.com/academic/archives/1999/12/attack_trees.html

Secure software development lifecycles (SDLCs)

Also known as **application security**.

[BSA-SDLCv1.1] *The BSA Framework for Secure Software – A New Approach to Securing the Software Lifecycle v1.1*; BSA (2020); accessed at https://www.bsa.org/reports/updated-bsa-framework-for-secure-software Excellent and progressive document which takes a risk-based approach, meaning that one of the first things it recommends is conduct of a threat analysis, and is outcomes-focused, meaning that by stating the desired outcomes without stating how to achieve them the requirements are technology-agnostic. Offers a set of requirements for developing secure software. Under Secure Development > Supply Chain there are recommendations to track third party components, that the software is easily identifiable, that is it protected from tampering. Draws on SAFECode but is different in its organisation, its broader scope and in how it references many other standards. Fully implements NIST's Secure Software Development Framework, with mapping.

[BSIMM] *Building Security In Maturity Model (BSIMM) version 11*; Synopsys (2021); accessed at https://www.bsimm.com Completely empirical annual report documenting current secure software development practices across a wide membership.

[ISO/IEC-27034-1] *ISO/IEC 27034-1:2011 Information technology — Security techniques — Application security — Part 1: Overview and concepts;* ISO/IEC (2011); accessed at https://www.iso.org/standard/44378.html Closely aligned with ISO/IEC 27005 for information security risk management, this is part one of a seven-part standard describing how an organisation can set up effective, efficient processes for building security into applications and application development. It explicitly is not a standard for secure applications or application development, only of how those can be achieved. It is aimed at those specifying, designing, developing or procuring software applications and is notable in this context for how it makes no distinction between in-house and 3rd-party developments.

[NIST-SP800-218] *SP 800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities*; NIST (2022); Murugiah Souppaya (NIST), Karen Scarfone (Scarfone Cybersecurity), Donna Dodson; accessed at https://csrc.nist.gov/publications/detail/sp/800-218/final

[OWASP-SAMMv2] *Software Assurance Maturity Model (SAMM) version 2*; Open Web Application Security Project (OWASP) Foundation (2020); accessed at https://owaspsamm.org Intended to give organisations developing web applications a to analyse and improve their web application security. Covers governance, design, implementation, verification and operations. Presents 15 security practices each with three levels of maturity. Supplier security is considered explicitly under the Security Requirements and Secure Build practices.

[PCI-SSFv1.1] *Software Security Framework version 1.1*; Payment Card Industry (PCI) (2021); accessed at https://www.pcisecuritystandards.org/document_library?category=sware_sec#results Security requirements for card payment software, and its development, aimed at protecting the integrity and confidentiality of payment transactions and data. All vendors of card payment software must adhere to this standard. Like BSA it used an objective-based approach to requirements, instead of requiring a checklist of controls. Draws on SAFECode's Fundamental Practices for Secure Software Development [SAFEcode-SSD] and NIST's Cybersecurity Framework [NIST-CSF]. Certification requires that third-party components are inventoried, properly used, correctly functional, monitored for vulnerabilities.

[SAFEcode-SSD] *Fundamental Practices for Secure Software Development - Essential Elements of a Secure Development Lifecycle Program, Third Edition*; SAFEcode (2018); accessed at https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf Presents a collection of used practices, similar to BSIMM's approach, relating to secure design, development and testing of software. This version added management of vulnerabilities and third-party components, building on SAFEcode's 2017 paper *Managing Security Risks Inherent in the Use of Third-party Components*. Very creditably motivates every recommendation with examples of real vulnerabilities from the Common Weakness Enumeration (CWE) database.

Secure development lifecycles for IoT

[CSI-firmware] *Secure Firmware Development Best Practices version 1.1*; Cloud Security Industry Summit, Supply Chain Technical Working Group (2019); accessed at https://ogi-cdn.s3.us-east-2.amazonaws.com/csis/firmware-security-best-practices-v1.1.pdf Contains a state-of-the-art discussion of secure coding practices as well as an interesting discussion of the meaning of signatures, specifically in the context of firmware signing.

[ENISA-IOT-SDLC] *Good Practices for Security of IoT - Secure Software Development Lifecycle*; ENISA (2109); accessed at https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1 Very well-researched SDLC with an IoT focus. See in particular section 4.2.2.1 Third-Party Management.

[IIC-software-trustworthiness] *Software Trustworthiness Best Practices - An Industrial Internet Consortium White Paper, Version 1.0*; Industrial Internet Consortium (2020); Marcellus Buchheit (Wibu-Systems), Mark Hermeling (GrammaTech), Frederick Hirsch (Fujitsu), Bob Martin (MITRE), Simon Rix (Irdeto); accessed at https://www.iiconsortium.org/pdf/Software_Trustworthiness_Best_Practices_Whitepaper_2020_03_23.pdf Excellent review of secure software development practices with a focus on embedded systems.

[ISA/IEC-62443-4-1] *IEC 62443-4-1 Security for industrial automation and control systems - Part 4-1: Secure product development lifecycle requirements*; International Electrotechnical Commission (IEC) (2018); accessed at https://webstore.iec.ch/publication/33615 The popular ISA/IEC 62443 Security for Industrial Automation and Control Systems (IACS) family does for operational technology (OT) what the

ISO 27000 family does for information technology (IT). This part describes the development and maintenance processes to be used by IACS component vendors including security requirements definition, secure design, secure implementation (including coding guidelines), verification and validation, defect management, patch management and product end-of-life.

Software bills of materials (SBOMs)

[CycloneDX] *CycloneDX* v1.4; OWASP (2022), accessed at https://cyclonedx.org An XML SBOM specification designed for vulnerability identification, license compliance, and outdated component analysis use cases. The project provides tools to generate CycloneDX SBOMs in many language ecosystems. Originated in OWASP Dependency-Track, an OSS Software Composition Analysis (SCA) tool. V1.4 adds the Vulnerability Exploitability Exchange (VEX) feature, designed to automate communication of vulnerabilities and their exploitability for software defined in a bill of materials.

[SPDX] *Software Package Data Exchange (SPDX)*; Linux Foundation (2021); accessed at https://spdx.dev An SBOM format for communicating the components, licenses and copyrights associated with software packages. Largely aimed at OSS it relies on placement of SPDX tags in source files. Maintains a list of standardised license IDs to be included in files and/or package READMEs. Its main use case is compliance with OSS licences, by aiding automated inventory. Also published as ISO/IEC 5962:2021.

[SWID] *ISO/IEC 19770-2:2015 Information technology — IT asset management — Part 2: Software identification tag (SWID)*; ISO/IEC (2015); accessed at https://www.iso.org/standard/65666.html A SWID Tag document is an SBOM identifying a software product, its version, the organizations and individuals involved in producing and developing it, the artifacts comprising it. The original use case was to track usage of paid-for software by organisations, e.g. for billing per instance. Later, software asset tracking for cybersecurity purposes was added. NIST has produced NISTIR 8060 Guidelines for the Creation of Interoperable Software Identification (SWID) Tags to help people use SWID for cybersecurity purposes. However, it seems lightly adopted.

Hardware attacks & controls

IoT device hardware supply chains

[ENISA-hardware-threats] *Hardware Threat Landscape and Good Practice Guide*; ENISA (2017); accessed at https://www.enisa.europa.eu/publications/hardware-threat-landscape Surveys hardware attacks on embedded devices but excluding ICs and supply-chain (meaning pre-deployment) attacks. Provides a good list of hardware attacks to use in attack trees.

[SAE-counterfeit] *Counterfeit Electrical, Electronic, and Electromechanical (EEE) Parts; Avoidance, Detection, Mitigation, and Disposition AS5553D*; Society of Automotive Engineers International (2021); accessed at https://www.sae.org/standards/content/as5553d/ Intended to be used with a higher-level quality standard, this document puts requirements around the purchasing process (specify, check, verify) but also covers supplier management and what to do when counterfeits are discovered.

IC hardware supply chains

[Areno-IC-supply-chain-threats] Supply Chain Threats Against Integrated Circuits; Matthew Areno (Intel) (2020); accessed at https://www.intel.com/content/dam/www/public/us/en/documents/white-

papers/supply-chain-threats-v1.pdf Brief but informative survey of supply-chain threats including related studies and demonstrations and documented attacks.

[Huang-Counterfeit-ICs] *On Counterfeit Chips in US Military Hardware*; Bunnie:Studios (Bunnies' Blog); Andrew "Bunnie" Huang (2011), accessed at https://www.bunniestudios.com/blog/?p=2037 Characteristically insightful discussion on the sources and detection of counterfeit ICs in the context of US defense, which was and perhaps still is the security community most concerned about securing the IC supply chain.

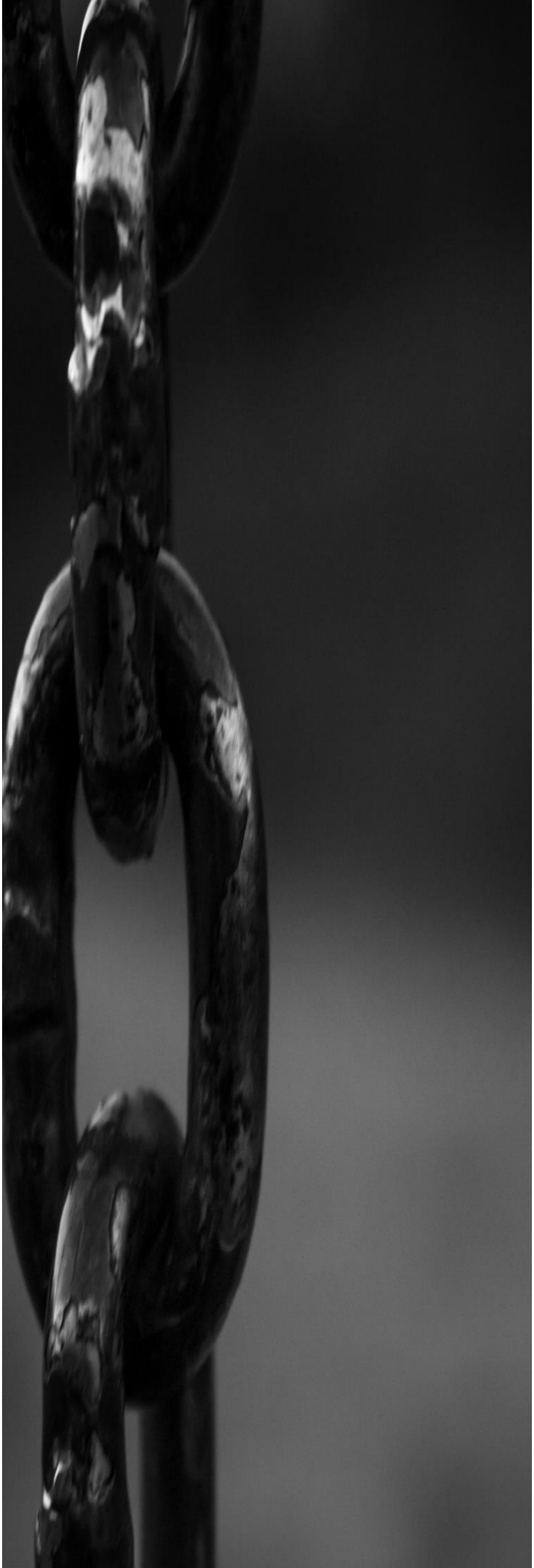
[Huang-IC-implants] *Supply Chain Security: If I were a Nation State...*; talk by Andrew "bunnie" Huang at BlueHat IL 2019, accessed at https://www.youtube.com/watch?v=RqQhWitJ1As Highly informative look at how ICs' integrity might be compromised in distribution. Useful comparison of classes of IC supply chain attacks, on axes of how hard they are to spot vs how hard they are to execute.

6. Attack Trees but it can be more specific. For a variety of insight, it is recommended to approach this as a group exercise in collaboration with product managers and design engineers.
7. Assess the cost of the identified attacks. Decide what cost of attack is necessary to sufficiently deter attackers. Take steps to eliminate lower-cost attacks. Following IoT Security Assurance Framework recommendations will be helpful.

Document End

IoTSF-SCIP V1.0.0 (2022-005-17)

An introduction to the new supply chain security requirements in the IoTSF Security Assurance Framework (r3)



IoT
Security Foundation