# WhitePaper

## The Internet of Disconnected Things

**“the only secure device is an unreachable device”**

August 2021
Author: Aaron Ardiri, CEO - RIoT Secure AB

# Introduction

When people talk about “the next big thing,” they never really think big enough.

The number of Internet of Things (IoT) devices worldwide is forecast to almost triple from 8.74 billion in 2020 to more than 25.4 billion IoT devices in 2030. IoT devices are used in all types of industry verticals and consumer markets, with the consumer segment accounting for around 60 percent of all IoT connected devices in 2020. This share is projected to stay at this level over the next ten years [1].

IoT security is definitely a glaring concern among professionals in the industry.

Many IoT devices contain significant security concerns; a result of a desperate race to market - where developers are eager to connect everything to the Internet and an industry prioritising production rates and profitability over cybersecurity.

Security by design is a mandatory prerequisite to securing the Internet of Things.

This white paper will highlight a disturbing historical account of real-world problems associated with lack of security in IoT devices and address the concerns to these problems outlining the idea behind the Internet of Disconnected Things.

# The Internet of Things (IoT)

IoT has seen exponential growth of smart devices and has completely transformed how we identify ourselves, communicate with others, shop for products, manage our existence and fulfil our work responsibilities.

A plethora of sensors produce a wealth of new information about environmental status, location, usage, behaviour and device performance that can be used to deliver lifestyle improvements and new business opportunities for companies.

Industries utilising IoT include: Agriculture & Farming, Energy, Enterprise, Finance, Telecom, Healthcare, Smart Buildings, Retail, Transportation and Industrial IoT [2].

IoT involves adding internet connectivity to a system of interrelated computing devices, mechanical and digital machines, objects, animals and/or people. Allowing devices to connect to the internet opens them up to a number of serious vulnerabilities if they are not properly protected.

IoT security refers to methods and ideology for securing network-based devices.

# Timeline of Concern

IoT devices expose a particularly large attack surface due to their internet-supported connectivity. While this accessibility is extremely valuable, it also provides the opportunity to interact with devices from anywhere in the world.

IoT hacking has become a relatively low-effort, high-reward opportunity for cybercriminals - demonstrating a massive potential of disruption and destruction.

Attacks subsequently have made headlines, more prominently in the last decade.

**1990s** Security experts have long warned of the potential risk of large numbers of unsecured devices connected to the internet since the IoT concept first originated.

**2010** Researchers revealed that the Stuxnet [3] virus was used to physically damage Iranian centrifuges (part of its nuclear program), with attacks starting in 2006 but the primary attack occurring in 2009. Stuxnet targeted supervisory control and data acquisition (SCADA) systems in industrial control systems (ICS), using malware to infect instructions sent by programmable logic controllers (PLCs).

**2013** A pair of Armenian software engineers, Sergey Shekyan and Artem Harutyunyan [4] exploited China based Foscam security camera manufacturer to gain remote access to baby monitors and scream obscenities to children in the security of their own bedrooms.

Smart Refrigerators or TVs [5] being hacked to send out spam also appeared.

**2015** Security researchers Charlie Miller and Chris Valasek [6] executed a wireless hack on a Jeep, changing the radio station on the car's media center, turning its windshield wipers and air conditioner on, and stopping the accelerator from working. Conceptually they could also kill the engine, engage the brakes or disable the brakes altogether - the exploit was performed via Chrysler's in-vehicle connectivity system, Uconnect.

**2016** IoT devices exposing default or hard coded credentials became zombies to large botnets such as Mirai [7]; eventually used to perform Distributed Denial of Service (DDoS) attacks on journalist websites, domain name service providers and popular Internet services making them unavailable for hours. The attacks infiltrated the network through consumer IoT devices, including IP cameras and routers.

**2017** The Food and Drug Administration (FDA) also warned [8] that embedded systems in radio frequency-enabled implantable devices manufactured by St. Jude Medical including pacemakers and defibrillators, could be vulnerable to security intrusions and attacks.

The recall would not see the pacemakers removed, which would be an invasive and dangerous medical procedure for the 465,000 people who have them implanted: instead, the manufacturer issued a firmware update which will be applied by medical staff to patch the security holes.

Devil’s Ivy [9], a stack overflow vulnerability allowing remote code execution was found in an open source library from gSOAP (utilised by the ONVIF protocol) that was used by notable security camera vendor, Axis communications. Avis disclosed over 249 distinct camera models were impacted (exception being 3 older models). The exploit theoretically impacted hundreds of other camera vendors as well.

**2019** Almost half a decade after its introduction, the Mirai (and variants) botnet has continued to adapt, recruiting additional IoT devices, gained ongoing community support amongst hacker circles posing a

continuous ongoing threat to the Internet as a whole.

A cyber attack on video surveillance startup Verkada [10] has seen around 150,000 video cameras, many of them in secure locations (schools, prisons, private factories including Tesla), compromised.

**2021**

IoT platform Kalay was compromised resulting in a reported 83 million IoT devices; the majority real-time cameras - exposing live video feeds and ability to obtain usernames and passwords allowing attackers to execute remote code. The attack stemmed from disassembly of mobile-phone applications used to discover, register, authenticate and establish remote sessions with devices [11].

As firms continue with the digital transformation of their business introducing IoT devices, many companies were simply not prepared or expected to invest the money and resources required to secure such devices.

# Largest Attack Surface in History

> "We inadvertently created the largest attack surface in the history of computing.
>
> All these compute devices, not behind the walls of a fortress or nodes inside a data centre or cloud; they are out sprinkled all over the place, they got default passwords and usernames, they don't have encryption, they don't have a lot of things; they are out there, they are getting hacked, turning into zombies." [12]
>
> **Rob Tiffany, VP & IoT Entrepreneur @ Ericsson**

An IoT device with an excessive amount of system resources is an obvious target for any hacker to exploit and include as a zombie in a growing hacker botnet.

While the majority of exploits can be blamed on lack of security preparedness that could easily be addressed by following well documented hardening guides [13]; there is a constant threat that is inevitable for any exposed service on IoT devices, the public disclosure of zero-day vulnerabilities [14].


## Zero-day Vulnerabilities

Zero-day vulnerabilities are software vulnerabilities discovered by attackers before the vendor has become aware of them; vendors deploy solutions and with no patch available, attacks using the vulnerabilities are guaranteed to succeed.

Exploiting bad code or programming practices, such as memory corruption scenarios that result in a permission escalation [15], the attacker can gain root level access to make modifications and changes; typically implanting a trojan horse to provide shell access to quickly utilise the resource at a later time.

The only way to mitigate zero-day vulnerabilities is to completely disable unused services exposed by default or constantly maintain security updates for the device. In cases where patching physical devices isn't possible - the attacks can be handled in the network layer using virtual patching techniques [16].

In any event; trying to stay ahead of the hackers is a classic chicken and the egg scenario - exploits must be found before patches are available and there is always a window of opportunity to exploit between these two points in time.

# The Internet of Insecure Things

The industry has quickly become the butt of jokes and comical expression.


Gertjan Kleijne, Live Cartoonist / Illustrator

Finding a solution to the problem has become an almost impossible task; an industry hoping that a series of standards and set of design guidelines would magically appear - only to be hindered by the fact since IoT spans across so many different industries and would require global cooperation to pull off such a task.

A variety of international organisations and government groups are working on issues pertaining to security and IoT, but at present there's no unified coordinated vision to implement standards for IoT on a global scale across all industry verticals.

The Internet of Things has suddenly evolved into the Internet of Threats - where sophisticated botnets recruit thousands of IoT devices with every exploit utilised.

# Easy Part: A Thing

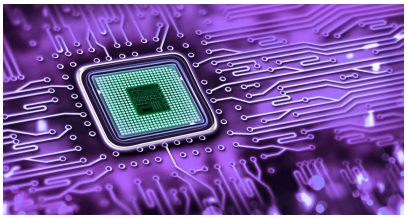
IoT refers to the billions of connected devices, all collecting and sharing data.

When excluding the element of connectivity, a Thing effectively becomes a micro-controller interfacing to various sensors to collect data and driving actuators to control or change the environment based on analysis of the data collected.

IoT prototypes can be assembled and demonstrated in a short period of time.

Developers can easily build proof of concepts using off-the-shelf prototyping hardware available from various vendors ranging from resource constrained 8-bit micro-controllers through to more powerful 32-bit micro-controllers with in-built machine learning functionality using a wide variety of programming languages.

Such programs can be as simple as IFTTT (if this, then that) or complex AI/ML (machine learning) models that process real-time data from sensors and deterministically trigger alarms/events on the device that may require signalling the driving of an actuator.

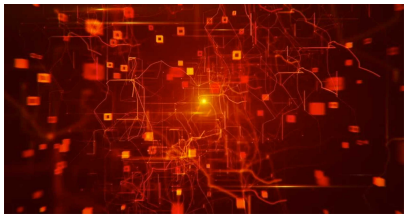


Interfacing between the micro-controller to the sensors and actuators requires a quick lesson in basic electronics (Voltage, Current, polarity, resistors et al.); then connecting components to the micro-controller that communicate over simple digital or analog signals or serial protocols such as SPI, I2C or UART.

Programming and debugging devices is performed over a USB serial connection; at which point the device is completely isolated and secure from external threats unless it explicitly connects to the Internet making it vulnerable to external attacks.

# Hard Part: The I in IoT

The moment a Thing needs to be connected to the Internet - the hard work begins.



Developers suddenly require knowledge of C/C++ and extensive embedded experience, advanced knowledge of real-time operating systems and security concepts.

The Thing should also not only be capable of providing the required level of security - but also encapsulate lifecycle management components into the product to ensure firmware or machine learning model updates can be provided over the air.

### Security and Privacy - Root of Trust

A Root of Trust establishes a secure anchor point for the chain of trust that will uniquely identify the device and be used over its entire lifecycle, effectively protecting the device against counterfeiting, cloning and reverse engineering [17].

Cryptographic algorithms and security protocols are computationally expensive requiring processing and memory requirements that drastically exceed the basic computational requirements of the Thing that is being connected to the network.

Security is not only about cryptography, security also encompasses authentication processes, establishing and maintaining trust, ensuring exchange of information confidential between components and non-repudiation of device activity.

Developers must consider the privacy of sensitive information and its accessibility; information needs to be recorded, stored and securely transmitted between various components both internally and externally in the lifecycle of the device.

## Connectivity

Connecting so many devices is one of the biggest challenges of the future of IoT.

Most IoT connectivity models utilise a centralised network model; however, the future of IoT will very much have to depend on decentralising IoT networks.

Other compatibility issues stem from non-unified cloud services, lack of standardised protocols and diversities in firmware amongst IoT devices.

IoT protocols can be classified in terms of the role they play within the network; there are protocols used in connectivity infrastructure and communications, transmission of data, security through to device management.

Figuring out the right connectivity solution for the use case is a major challenge.

## Resource Constrained

Resource constrained devices have limited processing and memory capabilities; yet provide a functionality with minimal power input while remaining cost-effective.

While a Thing may not actually require significant resources to do the task that they are initially designed to perform; when the micro-controller takes on additional responsibilities - specifically networking logic, security algorithms and associated libraries, the required processing and memory footprint can change drastically.

## Robustness and Stability

When IoT applications are business critical, they require the underlying technology to be dependable and reliable - be capable of delivering services even in the event of failure. IoT devices hence must encapsulate robustness and recovery options.

## Data Collection & Analytics

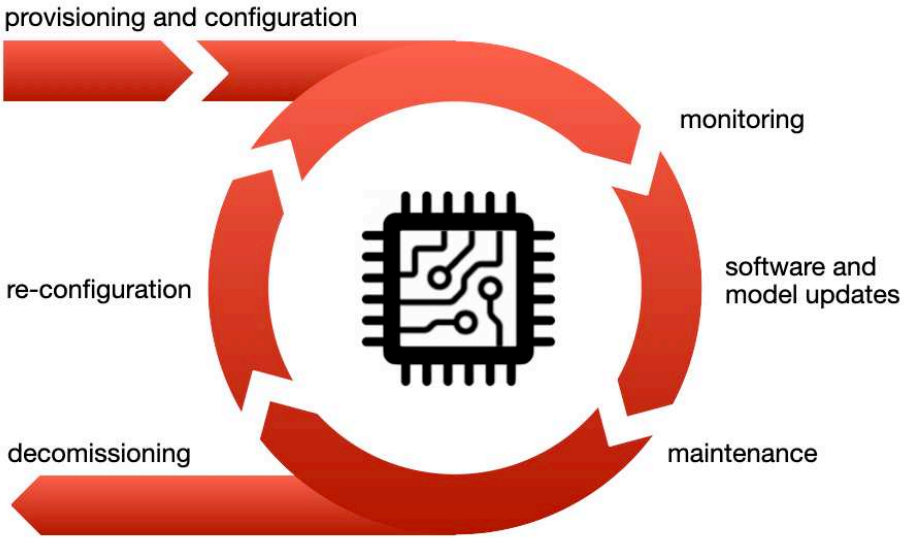
Specialists fear that sending raw telemetry data could expose industrial secrets.

With large amounts of data being generated by sensors; concerns arise to where this data will be stored for eventual transmission to the cloud, or caching for real time on-board analysis where data depends heavily on cognitive technologies and accompanying models that require significant training and adaptability.

## End-to-End IoT Lifecycle Management

Device deployment is just a phase in ongoing management of IoT environments.

Managing large volumes of devices is a significant challenge and concern. IoT lifecycle management ensures that IoT devices are not only deployed and successfully maintain communication; but continue to operate efficiently while also remaining compliant and useful for the lifetime of their service.

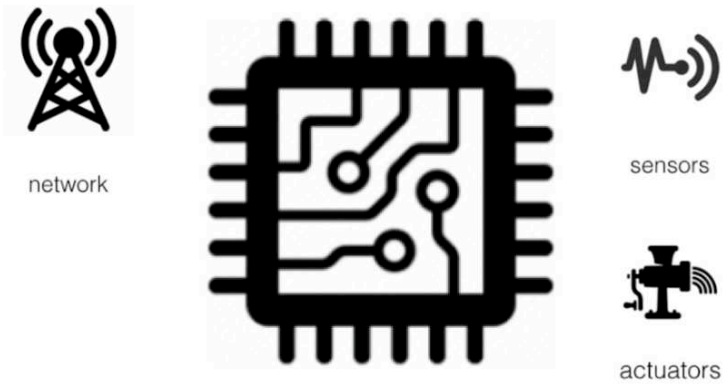



The IoT lifecycle management platform should provide the ability to initially provision and configure an IoT device, deploying it for use, where it then undergoes a constant cycle of monitoring, software (and model) updates and maintenance until it is either re-configured or decommissioned from service.

The 2021 IoT & Edge Computing Commercial Adoption Survey Report [18]; identified that 44% of respondents highlighted IoT lifecycle management as the top operation challenge; followed by device management (40%) and securing network communications and privacy of IoT device data (35%).

# Anatomy of an IoT Device

IoT devices vary in size and function - but at their basic core, they effectively require two components; a brain (micro-controller, typically connected to sensors and actuators) and connectivity component for establishing Internet connectivity.



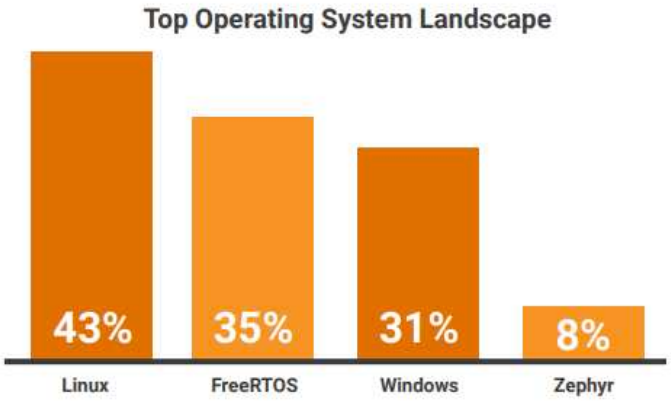

The micro-controller contains a central processing unit (CPU), flash program memory (ROM), volatile storage (EEPROM), random-access memory (RAM) and IO ports to connect to sensors, actuators and network module for communication.

The processing power of the CPU is directly proportional to the complexity of the tasks the Thing needs to accomplish - some IoT devices will be sufficient to use low powered, resource constrained micro-controllers while others will require higher processing power to execute more demanding tasks and analysis.

The amount of memory available is equally important as overall performance and capabilities of a micro-controller depend on the memory (ROM and RAM) size.

Every IoT device needs to communicate to serve its purpose; some devices only need to send information, while others need to both send and receive. When a device accepts incoming network connections - it is susceptible to exploitation.

The choice of underlying operating system and the services exposed to the network directly determine the attack surface area and threat level of hacking.




The IoT Developer Survey 2020 [19]; reported that 74% of constrained devices and edge nodes utilised traditional operating systems such as Linux (43%) or Windows (31%) - the remaining operating systems targeted micro-controllers directly.

Targeting micro-controllers directly; the operating system may lack the ability to run IoT applications in user space [20] - an execution environment that protects memory and hardware against malicious or erratic software behaviour; when an memory management unit (MMU) or memory protection unit (MPU) is present. Even with protection; a kernel panic can occur that can halt the micro-controller rendering the IoT device useless and unable to communicate to the cloud.

## Bare-metal / No Operating System

The most simplistic of IoT Applications, typically written in C/C++ are delivered as binary firmware compliant to the micro-controller. The application reads sensors, drives actuators and network communication in a single threaded environment.

## Real Time Operating Systems (RTOS)

RTOS builds a suite of kernel level functionality using hardware timers that expose a pre-emptive threaded environment; providing tasks, mutexes, semaphores and software timers to deliver a rich, task oriented development environment. The IoT Application will be written in C/C++, divided into multiple tasks and delivered as a binary firmware together with the RTOS kernel for the micro-controller.

## Real Time Operating Systems (RTOS) - Sandbox Environment

The micro-controller itself may run a dedicated operating system (such as RTOS) and provide developers the ability to run IoT Applications in a user space or sandbox environment where the code is executed in a protected environment.

A sandbox creates an operational environment in which the execution, operation and processes of a program without affecting the environment or platform on which they are executed; via the use of a virtual machine or runtime environment.

The IoT application can be written in C/C++, but also a variety of different programming languages such as Java, JavaScript, Python and other scripting languages when the appropriate runtime environment is available to the developer.

## Linux and Variations

Linux began life as a hobbyist operating system designed by Linus Torvalds originally targeting personal computers (PCs) using 80386 processors, a 32-bit micro-processor introduced by Intel. Since its introduction in 1991, Linux has grown into a broadly used operating system, with numerous distributions deployed on PCs, servers, mainframes, mobile phones, and IoT devices.

Factors such as its open-source operating system, scalability, security features, integrated development toolchains coupled with a wide range of distributions, make Linux (and its variants) a popular choice for IoT development.

IoT devices running Linux are both application and service orientated - executing on top of the Linux kernel are system services exposing open network ports [21] in parallel to the IoT application. Linux is a common target of IoT device attacks due to the large attack surface and usage exposed in default configurations [22].

## Windows 10 IoT

Windows 10 IoT is a member of the Windows 10 product family that brings enterprise-class power, security, and manageability to the Internet of Things [23].

Microsoft currently has three different subfamilies of operating systems for embedded devices targeting a wide market, ranging from small-footprint, real-time devices to point of sale (PoS) devices like kiosks.

Windows based IoT devices are increasingly popular due to the rise of Azure IoT cloud services that allow seamless integration however are still in the sights of hackers with known cases of exploits also being discovered and published [24].

Microsoft has recently redirected its efforts on a project called Azure Sphere, a Linux based operating system requiring specific hardware and interoperability with the Azure IoT cloud and is so convinced of its security that it has offered a $100,000 USD bounty to hackers to break the security [25].

# IoT Development is Hard

Developing an IoT solution is one of those concepts that appears deceptively simple; connect a device to the network and send data to a cloud service for analysis and processing. Unfortunately, it is not as straight forward as that.

IoT developers not only require a wide range of skills depending on the nature of the project; but significant experience to build industry grade end-to-end solutions. Finding a combination of multiple disciplines in a market where each of them is already scarce makes hiring for IoT an uphill battle for companies [26].

Technology concepts within the field of IoT are constantly changing at rapid pace, this means that developers must be more flexible than ever in their ability to adapt, learn and grow - concepts that may be valid one day, may not necessarily be applicable the next as requirements and the technical landscape evolve.

What approaches are used in software development to solving complex problems?

## Separation of Concerns

Separation of Concerns (SoC) is a design principle for separating a computer program into distinct sections such that each section addresses a separate concern [27]. When concerns are well-separated, there are more opportunities for module upgrade, reuse, and independent development.

SoC is achieved by the establishment of boundaries; any logical or physical constraint which delineates a given set of responsibilities resulting in more degrees of freedom for some aspect of the program's design, deployment or usage [26].

The goal is to more effectively understand, design, and manage complex interdependent systems, so that functions can be reused, optimised independently of other functions, and insulated from the potential failure of other functions.

SoC can also be visualised as divide and conquer model - splitting a problem into sub-problems that are much easier to comprehend and solve independently. When all sub-problems are solved, the compounded problem is inadvertently solved.

## Side Car Methodology

The Side Car methodology dictates modular components of an application exist a separate process or container that provides supporting features in isolation and encapsulation [28], unlike SoC where the components exist in the same process.

A key advantage of the Side Car service is the service can be developed in completely different programming languages and runtime environments while still being utilised from the parent using interprocess communication APIs.

A Side Car process shares the same process lifecycle (start, running and finish) as the parent application, effectively running as a parallel dependent service.

# Internet of Disconnected Things

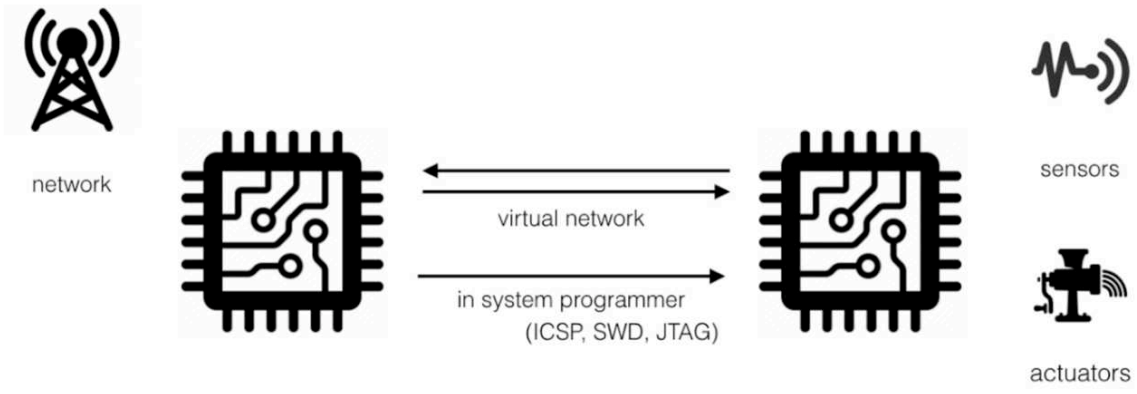
There is a clear SoC between the Thing implementation and network connectivity.

The exposed attack surface of an IoT device has been clearly identified as the single largest cause of recent security breaches - hackers have been able to utilise vulnerabilities of open services and network ports gain remote access.

The concept of Air Gapping [29], otherwise known as a disconnected network, is a network security measure that ensures that a computer is physically isolated from insecure networks - such as a public Internet or unsecured local network.

Applying the Side Car methodology, we can separate the network from the Thing.

Instead of utilising a single micro-controller; the hardware design dictates the use of multiple micro-controllers - one dedicated for the networking; providing security and communication, while the other provides the Thing implementation.




In this design; the simplicity of developing a Thing is retained - there is a clear network isolation of the micro-controller interfacing with the sensors and actuators, invisible to the public network; effectively acting as a traditional network firewall that would make connecting from the external network impossible.

A virtual network interface exists between the micro-controllers; effectively acting as a VPN tunnel to exchange bi-directional data between the Thing and the cloud.

## Micro-controller: Communications

The networking micro-controller exclusively provides communication with the cloud; providing lifecycle management (OTA firmware updates) by utilising the micro-controller's in-system programming interface and secure communication.

While the micro-controller is technically exposed to the insecure network; it has a specific role and purpose which can ensure tight security hardening is implemented; regardless of the purpose of the Thing that it is connected to.

The micro-controller only requires enough resources to provide end-to-end secure communications; meaning it can technically be a low powered, resource constrained micro-controller and not a desirable or feasible target to hackers.

### Micro-controller: Thing

The use case of the Thing will typically drive the selection of micro-controller used.

The developer now also has freedom with regards to the choice of:

- micro-controller
- operating system,
- programming language,
- run-time environments and
- libraries and frameworks


Ultimately the choice of technologies used in the development of the Thing will come down to the experience, knowledge and personal preferences of the developer and team working on the project.

Take advantage of the freedom to choose a micro-controller that best suites the requirements the project utilising the skills of your own workforce without limiting your micro-controller choices and avoid vendor lock-in with no escape.

Comprehensive documentation and support networks is critical for anyone using a micro-controller for the first time that may need guidance and information to make informed decisions on the specifications, features and how to program and use it.

# What's Next?

Perhaps one of the top priorities is managing expectations.

IoT is everywhere, gradually invading every aspect of consumers and businesses.

The technical world has very high expectations for IoT technology in the future as there are endless possibilities and customisable options for almost all tools everyone uses, across so many vertical markets, on a daily basis.

While new IoT products are constantly being developed, there is also still a practical opportunity for companies to retrofit their equipment and infrastructure rather than replace them with the latest and greatest at a fraction of the cost.

The Internet of Disconnected Things will not change the way IoT technology and platforms are being built - due to substantial investment already committed and that some technology providers have worked through the development pain.

Nor will it solve all aspects of IoT Security - it specifically targets edge IoT devices.

It offers an alternative approach and mindset for new IoT technologies and solutions where the simplicity of IoT device prototyping can be extended to production seamlessly - focus entirely on the Thing, not worrying about security.

Developing an IoT solution end-to-end is a cross-functional task that requires many skill-sets and a great deal of technology expertise to be successful. Few companies will be able to deliver solutions alone - it is vital to ensure that partners within the ecosystem are of knowledgeable and of complementary capabilities.

# About RIoT Secure

RIoT Secure was founded in Stockholm, Sweden.

RIoT Secure is a technology enabler within the IoT (Internet of Things) industry - created with a vision to ensure security is available to all IoT things, regardless of the computational and memory resources available and ensure information is secure within the IoT ecosystem.

The company provides a range of technologies from cryptographic technology optimised for low-powered resource constrained devices, network agnostic developer modules through to a complete device lifecycle management platform tailored specifically to the requirements of any IoT ecosystem.

The team, while scattered across the globe, is not only highly motivated but has extensive management and technical expertise specifically within the embedded platforms and the Internet of Things (IoT).

## SCOUT - AN EIT DIGITAL PROJECT

Secure Lifecycle Management and Network Intrusion detection for Industrial IoT.

The SCOUT platform focuses on secure lifecycle management of IoT devices, designed with security in its foundations and architecture, specifically for complimentary inclusion in small or large scale Industrial IoT environments.

INGWAZ
SECURE DEVICE LIFECYCLE MANAGEMENT
AND NETWORK INTRUSION DETECTION OF INDUSTRIAL IoT

www.ingwaz.io

The Ingwaz project has received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme.

RIoT SECURE

**The Internet of Disconnected Things**

August 2021
Author: Aaron Ardiri

RIoT Secure AB
www.riotsecure.se

# References

[1] Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2030
https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[2] IoT Verticals
https://www.c2m.net/iot-verticals.aspx

[3] The real story of Stuxnet
https://spectrum.ieee.org/the-real-story-of-stuxnet

[4] The Half-Baked Security Of Our 'Internet Of Things'
https://www.forbes.com/sites/kashmirhill/2014/05/27/article-may-scare-you-away-from-internet-of-things/

[5] Smart Refrigerators Hacked to Send out Spam: Report
https://www.nbcnews.com/tech/internet/smart-refrigerators-hacked-send-out-spam-report-n11946

[6] Hackers Remotely Kill a Jeep on the Highway—With Me in It
https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

[7] The Mirai Botnet Attack and Revenge of the Internet of Things
https://www.varonis.com/blog/the-mirai-botnet-attack-and-revenge-of-the-internet-of-things/

[8] Hacking risk leads to recall of 500,000 pacemakers due to patient death fears
https://www.theguardian.com/technology/2017/aug/31/hacking-risk-recall-pacemakers-patient-death-fears-fda-firmware-update

[9] Devil's Ivy: Flaw in Widely Used Third-party Code Impacts Millions
https://blog.senr.io/blog/devils-ivy-flaw-in-widely-used-third-party-code-impacts-millions

[10] Attack on surveillance cameras a warning over security, ethics
https://www.computerweekly.com/news/252497593/Attack-on-surveillance-cameras-a-warning-over-security-ethics

[11] Critical IoT security camera vulnerability allows attackers to remotely watch live video - and gain access to networks
https://www.zdnet.com/article/critical-iot-security-camera-vulnerability-allows-attackers-to-remotely-watch-live-video-and-gain-access-to-networks/

[12] The Rob Tiffany Digital Podcast #6 (IoT represents the largest attack surface in history)
https://www.linkedin.com/feed/update/urn:li:activity:6829798453653196800/

[13] Raspberry Pi Hardening Guide
https://chrisapproved.com/blog/raspberry-pi-hardening.html

[14] What is a Zero-day Attack? - Definition and Explanation
https://www.kaspersky.com/resource-center/definitions/zero-day-exploit

[15] Privilege escalation
https://en.wikipedia.org/wiki/Privilege_escalation

[16] Security 101 : Virtual patching
https://www.trendmicro.com/vinfo/us/security/news/security-technology/security-101-virtual-patching

[17] IoT Security: It Starts with a Strong Root of Trust
https://www.intrinsic-id.com/sram-puf/iot-security-it-starts-with-a-strong-root-of-trust/

[18] 2021 IoT & Edge Computing Commercial Adoption Survey Report
https://iot.eclipse.org/community/resources/iot-surveys/

[19] 2020 IoT Developer Surveys
https://iot.eclipse.org/community/resources/iot-surveys/

[20] Operating Systems: User Space
https://en.wikipedia.org/wiki/User_space

[21] Common TCP/UDP Ports Used By Red Hat Enterprise Linux
https://www.hostingreviewbox.com/rhel-tcp-and-udp-ports/

[22] Hackers using Linux flaws to attack IoT devices
https://internetofbusiness.com/hacker-linux-flaws-attack-iot-devices/

[23] Windows 10 IoT
https://en.wikipedia.org/wiki/Windows_IoT

[24] New exploit lets attackers take control of Windows IoT Core devices
https://www.zdnet.com/article/new-exploit-lets-attackers-take-control-of-windows-iot-core-devices/

[25] Microsoft Offers $100,000 If You Can Hack This Linux Operating System
https://www.forbes.com/sites/daveywinder/2020/05/06/microsoft-offers-100000-if-you-can-hack-this-linux-operating-system/

[26] Why is IoT talent so hard to find?
https://www.ciodive.com/news/why-is-iot-talent-so-hard-to-find/449576/

[27] Design Pattern: Separation of Concerns
https://en.wikipedia.org/wiki/Separation_of_concerns

[28] Design Pattern: SideCar Pattern
https://docs.microsoft.com/en-us/azure/architecture/patterns/sidecar

[29] Air gap (networking)
https://en.wikipedia.org/wiki/Air_gap_(networking)