20 YEARS OF MULTOS SECURITY AND INNOVATION

DIGITAL SECURITY INSIGHTS

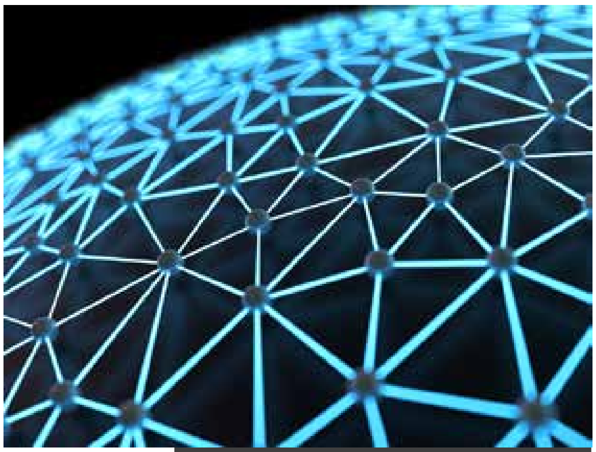
Concise news, information and know-how from the MULTOS Consortium

Issue1, 2017

Welcome to the first edition of the MULTOS Consortium's Digital Security Insights newsletter.

There is so much to keep abreast of in the fast-moving world of digital security. However many of us struggle to justify the time for reading about subjects not directly linked to our day-to-day jobs. We hope, therefore, to make the task of keeping up with technology a little easier by doing some of the up-front research for you and by presenting a concise summary of interesting topics. And of'course the MULTOS platform, its tools and associated services are constantly evolving, so the newsletter will also keep you up-to-date with developments.

We'd very much like to receive contributions or ideas for future issues, so if you have something you'd like to include or want to know more about, please let us know at info@multos.com



Fast News

Trusted Renewables joins the MULTOS Consortium:

TRUSTED RENEWABLES

This innovative company is combining digital trust with green energy harvesting to facilitate new business models in renewable energy and smart metering. Secure and efficient energy management is an expanding IoT market segment, one that Trusted Renewables Limited intend to help drive forward. Collaboration with MULTOS technology and the consortium will support their ongoing developments. Click here for more information.

Ingenious smartphone attack

Newcastle University

Researchers at Newcastle University, UK, have found that each user action such as clicking, scrolling, holding and tapping on a smartphone screen, creates a unique response on the phone's motion sensor. On a known webpage, the team was able to determine what part of the page the user was clicking on and what they were typing. Using this information, for example, they were able to crack four-digit PINs with 70 per cent accuracy on the first guess and 100 per cent by the fifth guess. Click here for more information.

Briefing: An Introduction to Blockchains by Chris Torr

At a show recently I was asked to explain what "The Blockchain" actually was. Considering that this was 50% of the show's content I did wonder why the questioner had not already had that particular question answered.

Luckily for me, my scant knowledge of the subject was enough to satisfy the questioner but it left me determined to find out more.

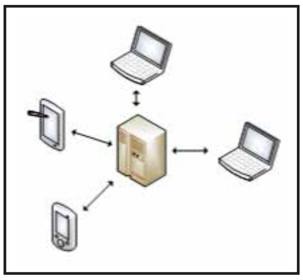
To start with then, we need to take a few tech. steps backwards.

What's in a Ledger?

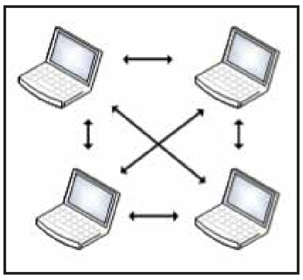
In a cash ledger, for example, transactions are written down and balances are updated. Each page is numbered and at the end of each page is a carried- forward balance. By inspecting the ledger entries it is possible to validate that the current balances are correct by sequentially recalculating the balances from the beginning (or any trusted point). It is also evident if the ledger has been modified at all because something somewhere won't add up, will be out of sequence or be visibly modified.

Each page of the ledger can be considered a "block" and the ordered sequence of the pages is the "chain".

Shared Ledgers



Traditional Centralised Processing Network.



Blockchain P2P (Peer to Peer) Distributed Processing Network.

Instead of using a central party to maintain a single ledger that everyone refers to, the update process is shared so that there is no single controlling central party. This is termed a Shared Ledger. All transactions are distributed to everyone and everyone updates their copy of the ledger. This is what most people are referring to when talking about a blockchain.

"A Shared Ledger is what most people are referring to when talking about a Blockchain"

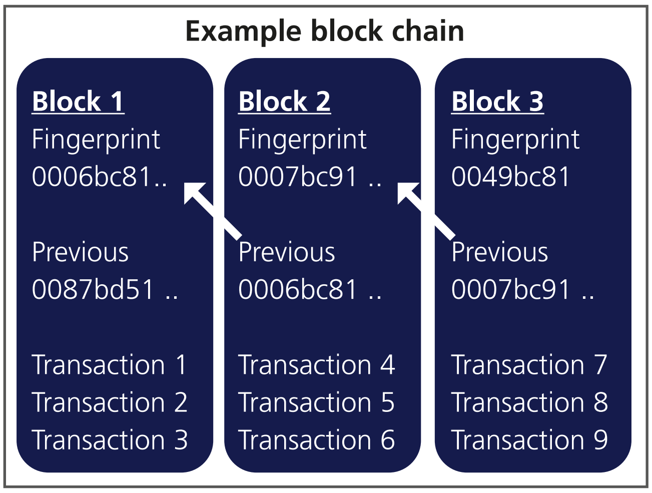
The three most obvious reasons for doing this are a) independence from any single controlling entity like a government or bank, b) for resilience and c) for non- repudiation of historical actions.

The single biggest issue is making sure that nobody cheats!

Implementation

A computer algorithm (usually based on cryptographic hashing techniques) is used to generate a fingerprint of the block. The block contains the transactions themselves and the fingerprint of the previous block in the chain.

If any details of a block are changed, the fingerprint calculation will change and the chain will break. Of course the chain's fingerprints could be reconstructed from the modified point onwards, but the resulting chain would look different to everybody else's and therefore be obviously fake.



Public blockchains

The detail of how a blockchain is implemented depends very much upon whether it is **public** (anyone is allowed to add blocks to the chain) or **private** (only a closed set are allowed).

In Bitcoin, a public blockchain, the processing of a block's finger-print is designed to be intensive taking up much computing time. Essentially the cryptographic hashing process used is made more complicated by adding additional dependencies linked to a random number.

So, if an attacker was trying to falsify an entry in a historical block it would be unfeasible for them to recalculate the remainder of the chain (given that every subsequent block would need to be reprocessed) before new blocks (that are being generated constant-ly) have been added to legitimate chains by others. The legitimate blockchain is therefore always going to be the longest one.

Private blockchains

In a private blockchain, the participants have agreed to collaborate so different mechanisms can be used to ensure the integrity of the blockchain.One such mechanism is for the work to be done in turn, with each participant signing the block they have just processed with their individual private key. The chain can still be verified by anyone by using each participant's shared public key, but cannot be modified because presumably the private keys are not available to a would-be counterfeiter in order to rebuild the chain.

Blockchains—continued

Common terms

Bitcoin: A digital currency that uses a blockchain shared ledger.
Proof of Work: The fingerprint generation algorithm. This is specific to the blockchain but usually uses a hashing technique or Public/Private key cryptography.
Mining: The process by which transactions are verified and added to a blockchain. This process of solving cryptographic problems using computing hardware also triggers the release of cryptocurrencies.
Miner: An entity in the blockchain network performing Mining.
Public key / Private key: Asymmetric cryptography uses keys that come in two parts; one part is shared (the public key) the other is key secret (the private key).

Potential issues

Blockchain technology is not without its potential hazards.
Scalability: Blockchains always grow. Bitcoin's blockchain is 108Gb at the time of writing. This could limit the number of participants who can eventually store the whole blockchain. Mining time is not necessarily linear potentially causing delays. Also, if blocks are too large, efficient distribution to all participants may be difficult.
Security: All private keys MUST be handled securely. Wallet applications on the web and on mobile phones may be (are) vulnerable.

Tech Tips: SDK Changes

The MULTOS platform is constantly evolving to meet the ever-changing needs of the markets it is used in. The latest SDK version, 3.1, was released in March 2017.

As well as bugfixes, several new features requested by consortium members have been included. Here we introduce a few of the enhancements.

Simulator static memory

One big part of the SDK is the card simulator. This allows for applications to be tested and debugged "off card". It has to closely match the implementation of MULTOS primitives and faithfully execute the virtual machine.

Until now however, the state of the non-volatile memory (or "static") used by an application has not been preserved between runs of the simulator. That is, each time the simulator starts, the static memory is initialised to the values specified in the source code. There are situations where this is less than convenient, for example where an application has to be personalized via APDU commands before being used, or perhaps you wish to perform tests based on a specific state that the application may have reached over time.

The solution introduced then allows for static memory to be stored in a binary file between runs of the simulator. The file is loaded when the application is selected and updated each time static memory is written to. The file must be named <aid>.mem where <aid> is the application identifier. For example **f3000001.mem**. Initially the file should be empty and reside in the current working folder. When using Eclipse, this is the root folder of the project. Note that if you change your application's static memory layout or code you will need to empty the file manually.

Comments in debugging files

When debugging MULTOS applications using Eclipse, a text file (commonly called debug.txt or debugging.txt) is used to define the debugging session (the simulator to use, application to debug, APDUs to send etc). One simple improvement in this release is to allow comments to be added to this file. If the first character of the line is a semi-colon ';' then the line is now ignored.

Delegation during debugging.

MULTOS applications have the ability to call one another through a secure process called Delegation. Previous versions of the debugger did not support this. This is now implemented, making it possible to step over a call to an application (the application will fully execute before returning) or step into the delegate application and debug it too.

If you have any requests for new features please e-mail dev.support@multos.com.

**Don't forget that there is a MULTOS developer forum at http://www.multos.com/forums/view-forum/5
To join e-mail dev.support@multos.com**

**The MULTOS SDK, SmartDeck, is fully integrated with the Eclipse CDT development environment.
Applications can be developed in 'C' or the MULTOS assembly language, MEL (or both!).**

MULTOS Q&A

Question: How do I tell how much free space there is on my MULTOS card?

Answer: The APDU command to use is Get Configuration Data as follows:
CLA=0x80 INS=0x10 P1=0x01 P2=0x00 Le=0x03
You can also use the Remaining E2 Size button in **MUtil.**

***

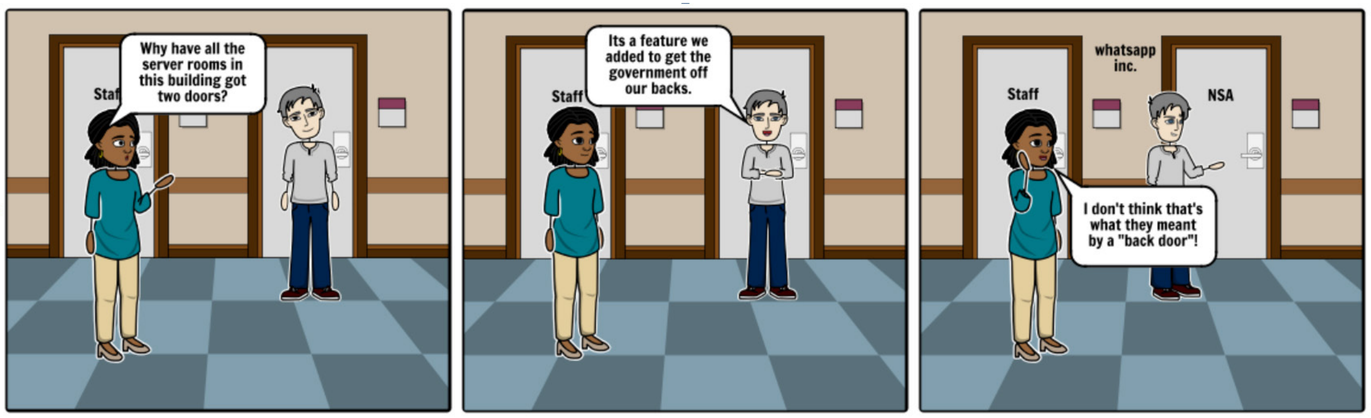
Question: Is MULTOS FIPS140-2 compliant?

Answer: All FIPS 140-2 certifications are for complete products; hardware + firmware (O/S) + software + particular configurations thereof.

Ref [FIPS 140-2] 4.1 "A cryptographic module shall be a set of hardware, software, firmware, or some combination there of that implements cryptographic functions or processes, including cryptographic algorithms and, optionally, key generation, and is contained within a defined cryptographic boundary."

MULTOS implementations can provide the hardware and O/S parts of such products and as relevant variants are already Common Criteria certified, it is a good choice for products wishing to achieve FIPS140-2 certification.

Digital Doodle

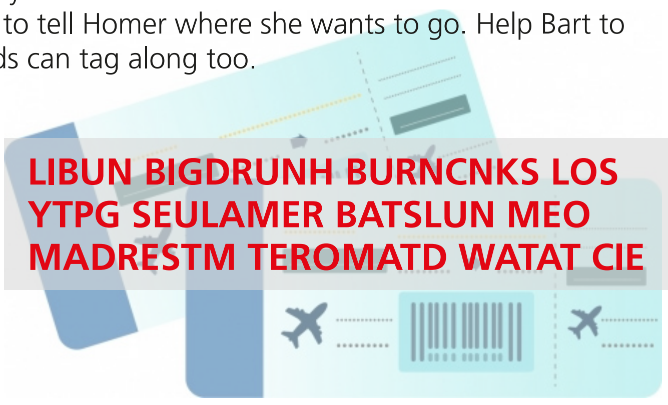

Staf
Why have all the server rooms in this building got two doors?
Staff
Its a feature we added to get the government off our backs.
Staff
whatsapp inc.
NSA
I don't think that's what they meant by a "back door"!

Prize Puzzle

Marge dreams of going on holiday without the kids and asks Homer to book the tickets. She uses a secret message to tell Homer where she wants to go. Help Bart to decipher the destination so the kids can tag along too.

Email your solution to
dev.support@multos.com

The first correct answer
wins a US$30 Amazon voucher.



LIBUN BIGDRUNH BURNCNKS LOS YTPG SEULAMER BATSLUN MEO MADRESTM TEROMATD WATAT CIE

Something to say? If you would like to contribute a short article or have a question you would like answered we'd like to hear from you.
Please e-mail us on **info@multos.com.**

www.multos.com

COMING UP...

» 1 billion MULTOS devices and growing fast.

» Physically Unclonable Functions.

» Using MUtil with KMA Webservices.