µNEST: An IoT+Blockchain System

Neo MA (n@iotee.io)

Ver. 1.0
2018-6-22

Abstract

After years of hearing news about Internet of Things (IoT) and Blockchain, we still cannot see a true IoT + Blockchain project in our real life. It' s still in dispute or discussion stages how to secure the Internet of Things, how to protect the privacy of data generated by IoT, how to guarantee Machine to Machine (M2M) interaction. The Blockchain also has lots of problems such as usability, availability, ease of use, etc. This paper unveils a blockchain designed specifically for IoT which offers the benefit to achieve the value exchange in non-trusted environment.

Introduction

The Internet of things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and network connectivity which enables these objects to connect and exchange data. Gartner, Inc. forecasts that 20.4 billion connected things will be in use worldwide by 2020.

A blockchain, originally block chain, is a continuously growing list of records, called blocks, which are linked and secured using cryptography. Each block typically contains a hash pointer as a link to a previous block, a timestamp and transaction data. By design, blockchains are inherently resistant to modification of the data.

The first blockchain was conceptualized in 2008 by an anonymous person or group known as Satoshi Nakamoto and implemented in 2009 as a core component of bitcoin where it serves as the public ledger for all transactions.

IoT and blockchain have so many matching points that we think they are made for each other. In particular, the deep dependence of applied cryptography makes blockchain technology one of the feasible and preferred solutions to solve the security problems of IoT. Smart contract technology also brings new dawn to M2M interaction. There are several famous pub-chains such as Bitcoin, Ethereum and BitShares (based on Graphene).

DPoS (Delegated Proof-of-Stake)

DPoS is a famous consensus algorithm that has been tested in real life for years. It has been proved stability and security by several projects, such as BitShares and Steem. It is known for its excellent performance like energy-saving, high-efficiency, robustness and security. To complete the consensus mechanism, it is still necessary to assist the corresponding consensus rules. e.g.

1. The longest chain will always win (just like Bitcoin)
2. Trusted node will always follow up the longest chain

And the algorithm is quite simple itself ,

```
for round i
    dlist_i = get N delegates sort by votes
    dlist_i = shuffle(dlist_i)
    loop
        slot = global_time_offset / block_interval
        pos = slot % N
        if dlist_i[pos] exists in this node
            generateBlock(keypair of dlist_i[pos])
        else
            skip
```

Proof-of-i/o

This is not a consensus algorithm! This is a contribution-reward algorithm. We just borrow Proof-of-X style to illustrate the contribution and reward of the μNEST ecosystem. We have designed and reserved 5% of total tokens for eco-node resource contributors, anyone who joins the μNEST p2p network and provides resources such as CPU, RAM, hard disk space, network bandwidth, or online time uptime will receive token rewards.

In the early stage, hard disk resources will only be used to store the μNEST ledger and calculated in bytes (per Byte); CPU and RAM will be used to verify transactions and were calculated per Transaction; while the online uptime will be calculated in terms of the duration of accessing the P2P network (per Minute). The above resource weights will be adjusted according to the different development periods of the μNEST ecosystem.

In the future, this algorithm will be upgraded to the data storage sub-Token and the transaction validator sub-Token, completing the Layer2 related core functions of the μZone. For environmental protection, energy saving and high efficiency, we do not need data backup of the whole network and all transactions verified by the entire online nodes.

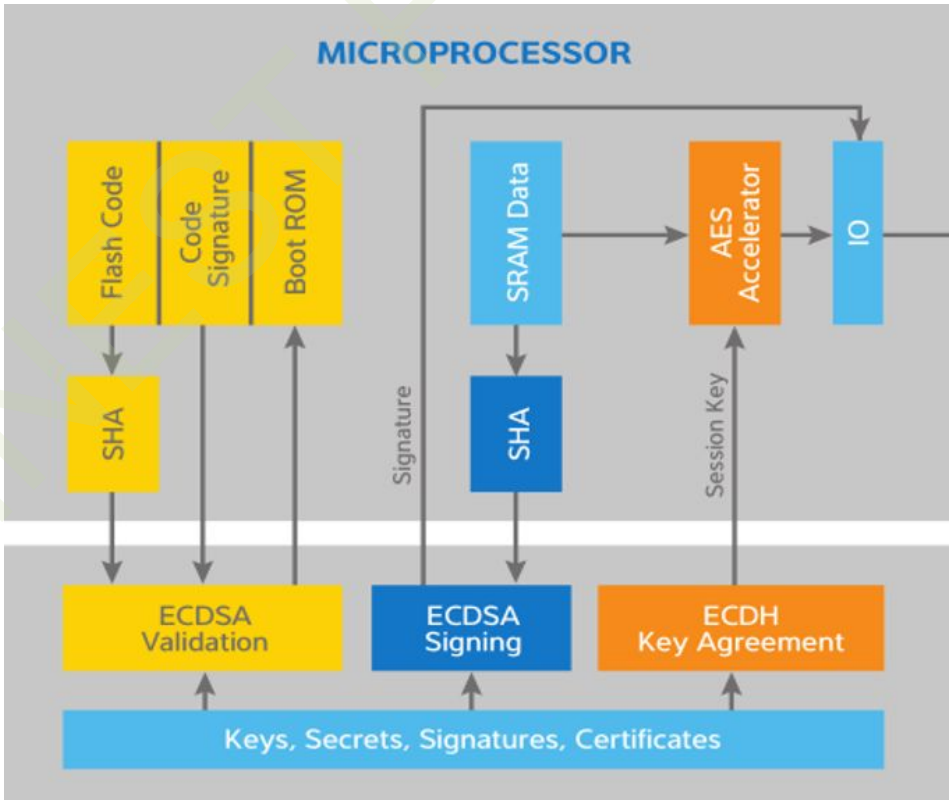
This algorithm is inbuilt part of the µGate and protected by the hardware trust root, which greatly reduces the chance of being hacked while making it almost impossible to cheat.

Hardware Trust Root

TEE (Trusted Execution Environment), TPM (Trusted Platform Module), and SE (Secure Element) have been widely used in industrial and consumer electronics fields. The core feature of them is to provide a physically isolated secure processor core and a secure instruction sets executing environment. It usually includes hardware cryptography algorithm implementation (such as ECC, RSA, AES, etc.), along with a hardware random number generator and a global unique hardware ID. These features make them the most preferred trust root in IoT+Blockchain system.

The trust root is the core of the security system and the root of the entire Chain of Trust. Today, the whole security system of Internet is based on PKI, which depends on the third-party trust roots. For various reasons, the PKI gets compromised by the Man in the Middle (MitM) attack from time to time.

In µNEST, we provide TEE, TPM or SE as the hardware trust root per different scenarios. While guaranteeing the chain of trust, we implement hardware-level transaction signatures for blockchain systems while benefiting from the hardware random number generator. When you need a payment scenario for M2M, you can choose EAL5+ SE.



Pic 1. a type of SE architecture from Microchip

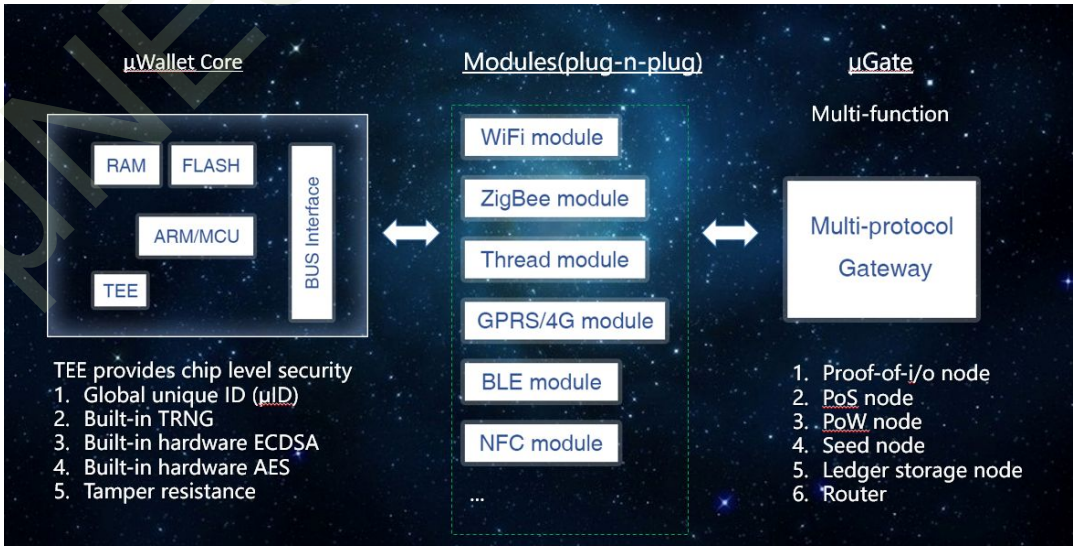
In addition to the above functions, the hardware trust root can also assist in completing hardware-level data encryption (e.g. AES) and various integrity checks (e.g. firmware integrity, data integrity, etc.).

µGate（Multi-protocol Multi-function Gateway）

Now and for a long time to go, we will still rely on TCP/IP based Internet. Therefore, while choosing the appropriate "local" interconnection protocol for the IoT scenarios such as ZigBee, WiFi, Thread, BLE, 5G/4G/2G, etc. it will still need to consider interaction with the Internet's TCP/IP protocol. The IoT system itself needs a protocol coordinator; On the blockchain side, not all the p2p nodes are completely equal, there are Full Nodes, Light Nodes, Block Producers, and Block Validators. Different nodes have different requirements on hardware and software carriers for running. For example, the hardware requirement of the full node is usually relatively high which needs a large memory, a large storage space and a high-frequency processor, etc., while the light nodes can run on smart phones, smart terminals or other smart devices.

We combined the above two requirements and designed a multi-protocol multi-function gateway, called µGate. It has the functions of both a protocol coordinator required by the IoT and a node running capability required by the blockchain. With the help of the security chip (aka the hardware trust root), the firmware and the application programs (e.g. Proof-of-i/o program) in µGate can run securely. µGate will have kinds of product line with different hardware resources in order to achieve high cost-effectiveness.

We do not develop or produce any RF modules. The µGate is designed to be modular. RF modules will be chosen from big mature manufactures based on the needs of the scenarios.



Pic 2. µNEST hardware architecture

µWallet

Blockchain is a natural financial system. From the public and private key pair to the account system, financial character is everywhere. Wallet is an integral part of the blockchain, µNEST is no exception. We designed a dedicated hardware and software wallet - µWallet. On the hardware side, µWallet Core is the joint of the IoT and the blockchain. The private key of µWallet Core can never be read out of the hardware. That is to say, it is tamper resistance. The globally unique ID of the security chip also provides the "original source" for the µID of the µNEST account system. The µID is derived from the hardware ID through hashing and shaping algorithms.

As shown in Figure 2, the µWallet Core topology is extremely simple which will connect outside via standard UART, SPI, I²C, or other bus interfaces. This allows the µWallet Core to be mounted onto motherboards such as smart locks, T-Boxes, etc., as well as Industrial 4.0 systems. With an industrial design surface, µWallet Core can be transformed into a secure cryptocurrency hardware wallet from the view of a traditional blockchain (the final product needs other auxiliary functions, such as USB interface, power supply, etc.).

On the software side, µWallet is an APP that contains µID and Status. It is the entrance to the blockchain. It provides kinds of features such as balance checking, transaction tracking, token receiving, token sending, etc.

µZone

Not all scenarios of IoT are "globalized". For example, smart homes, industrial internet of things, smart cities, such scenarios often occur in the geographical "local area". As we all know, the blockchain itself is not a TPS-friendly technology. In order to solve the TPS problem, we have creatively introduced the µZone's solution which combined hardware and software. Given layer2 technology like Lightning Network, the transaction authentication and data storage in a specific scenario is settled in a "local area". At a conservative estimate, µZone can reach 100,000 TPS.

Smart Contract

Ethereum was the first public-chain project to promote smart contract since Nick Szabo proposed Smart Contact in the 1990s. Smart contract, as a rich extension and supplement to the blockchain, needs a running container in the p2p network. In Ethereum chain, the smart contract runs on the Ethereum virtual machine (EVM), while the smart contract in Bitcoin chain runs on the stack. Based on Wren, we have

developed μNEST's own lightweight virtual machine, called nVM, which provides a lightweight, fast and convenient environment and a programmer friendly high-level programming language interface.

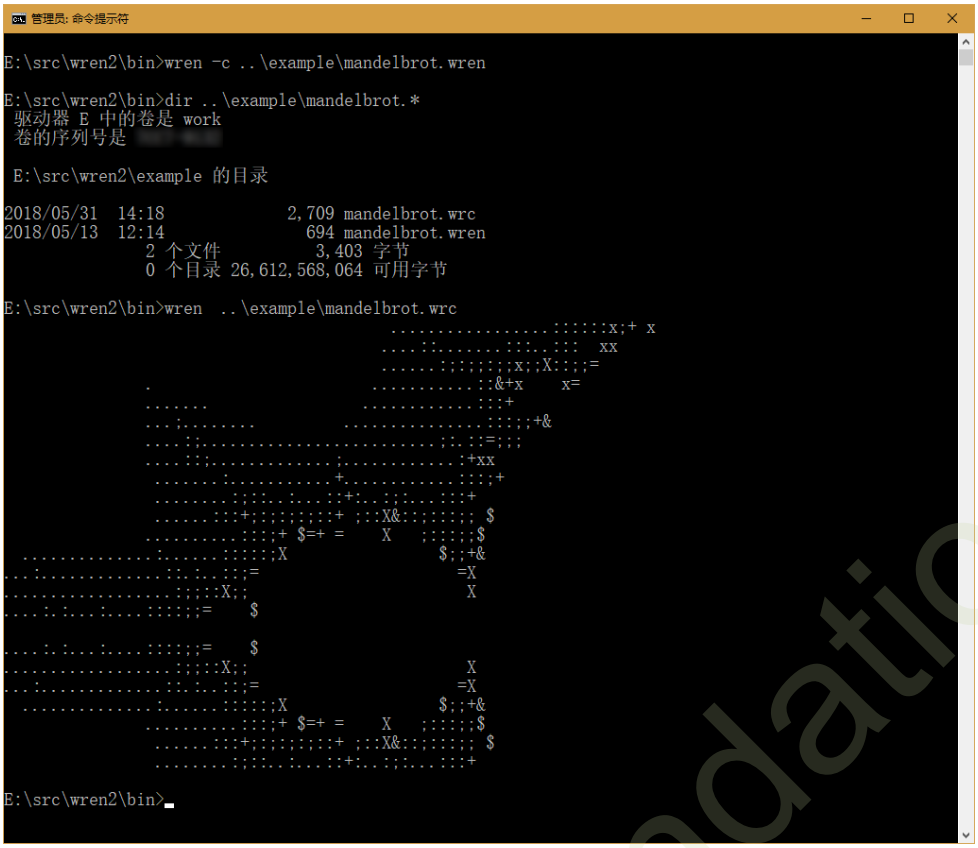
In order to reduce the probability of developers making mistakes when writing smart contracts, we have provided many examples and solidified many modules for developers to use. We also deploy various hardening methods to achieve the security of underlying level of the nVM. Enough is Better, between Turing completion and security, we are trying to find the best trade-off point.

```
881     }
882
883     // save all info of a single module to a file.
884     bool wrenSaveCompiledModule(WrenVM *vm, ObjModule *module)
885     {
886       bool ret = false;
887
888       uint32_t entryFnIndex = (uint32_t)-1;
889
890       Buffer methodNameBuffer;
891       Buffer variableBuffer;
892       Buffer fnBuffer;
893       Buffer fnIndexBuffer;
894
895       BufferInit(&methodNameBuffer);
896       BufferInit(&variableBuffer);
897       BufferInit(&fnBuffer);
898       BufferInit(&fnIndexBuffer);
899
900       if (SaveMoudleVariableToBuffer(vm, module, &variableBuffer)
901         && SaveModuleFnToBuffer(vm, module, &fnIndexBuffer, &fnBuffer, &entryFnIndex)
902         && entryFnIndex != (uint32_t)-1
903         && SaveModuleMethodToBuffer(vm, module, &methodNameBuffer))
904       {
905         const char *moduleName = MODULE_NAME(module);
906
907         ModuleFileHeader header;
908         memset(&header, 0, sizeof(header));
909
```

Pic 3. Saving module of nVM

```
wrenLoadCompiledMod  bool wrenLoadCompiledModule(WrenVM *vm, const char *moduleName, bool runClosure, ObjClosure **objClosure)
1385    bool wrenLoadCompiledModule(WrenVM *vm, const char *moduleName,
1386      bool runClosure, ObjClosure **objClosure)
1387    {
1388      Value name;
1389
1390      if (_stricmp(moduleName, CORE_MODULE_NAME))
1391        name = wrenNewString(vm, moduleName);
1392      else
1393        name = NULL_VAL;
1394
1395      ObjModule* module = getModule(vm, name);
1396      if (module && name != NULL_VAL)
1397        return true;
1398
1399      if (name != NULL_VAL)
1400      {
1401        wrenPushRoot(vm, AS_OBJ(name));
1402        module = wrenNewModule(vm, AS_STRING(name));
1403        wrenPushRoot(vm, (Obj *)module);
1404
1405        // Implicitly import the core module.
1406        ObjModule* coreModule = getModule(vm, NULL_VAL);
1407        for (int i = 0; i < coreModule->variables.count; i++)
1408        {
1409          wrenDefineVariable(vm, module,
1410            coreModule->variableNames.data[i]->value,
1411            coreModule->variableNames.data[i]->length,
1412            coreModule->variables.data[i]);
1413        }
```

Pic 4. Loading module of nVM



Pic 5. An Easter egg

Distributed Matching Engine

μNEST provides a distributed matching engine to match various "demands" and "services." In our point of view, many real-life scenarios can be abstracted into service demands and service offerings. Traditionally, this will be completed by various Web companies which providing various matching platforms. The blockchain system can provide such kind of matching mechanism, too. The Status of μWallet can contain unit price, location, time, availability and other information. Here is a smart parking example: The parking lot broadcasts the "service offering" message to the p2p network:

```
{
  "μID": "0x6192E6A705",
  "Status": {
    "PoR": "P",
    "Availability": "Y",
    "Price": 10,
    "Unit": "SET",
    "Location": [
      "37.341075",
      "-121.886838"
      ]
    }
}
```

Pic 6. "service offering" broadcast

A “demand needed” message is broadcasted by a car owner:

```
{
  “μID”: “0x1357A258BD”,
  “Status”: {
    “PoR”: “R”,
    “Price”: {
      “Mean”: 10,
      “Margin”: [
        -0.1,
        0.1
        ]
    },
    “Unit”: “SET”,
    “Location”: {
        "Center": [
          “37.123456”,
          “-121.789012”
          ],
        "Radius": 500,
        "Unit": "M"
      },
     “EST” : “40min”
    }
  }

```

Pic 7. “demand needed” broadcast

SET is the sub-Token（Sharing Economy Token）.

This can be accomplished by a simple matching algorithm, and the state will be locked by the smart contract on the chain until the "service" is completed and the payment is settled.

Multi-token

Single token blockchain system has an appreciation paradox: token appreciates with supply and demand, but with the appreciation of the token, the blockchain network is too “expensive” to use due to the transaction fee. Bitcoin chain is this kind, so is Ethereum chain. IoT is generating huge amounts of data every minute. This determines that Bitcoin, Ethereum and other single token chains are not suitable for IoT. μNEST is designed as a multi-token public chain. NEST is its main token. NEST is such a token that it can circulate inside and outside the μNEST ecosystem. Different scenarios of the IoT will have different sub-tokens. The SET mentioned above is a sub-token of the sharing economy scenario. It is designed as a transaction medium which anchoring the scenario. It cannot be circulated outside the μNEST ecosystem, which will make it relatively stable with respect to the sharing economy scenario, but does

not have the potential for hype. The sub-token will have a floating price ratio against NEST, which is dynamically adjusted by the chain and the contract according to some parameters. Similarly, there will be supply chain sub-tokens, V2X sub-tokens, smart home sub-tokens and so on.

Account System

μNEST wallet is designed with its special own account system. In addition to the conventional main public and private key pair managing different functions of sub key pairs, we designed an on-chain "watchdog" mechanism to solve the issue of the permanent assets loss or lock-up caused by password lost. This is an on-chain contract. The account holder will be asked a "question" in a certain number of days (for example, 2 weeks). If it cannot be "fed" within 1 day, it will enter the "closing door" countdown (7 days), during this period, if it still cannot be satisfied by the right "answers", it will enter the "closed" state and transfer the assets to an account pre-defined by the holder as an "inheritance" account after 7 days.

Data

IoT is Data, Data means Everything! We don't produce data , we provide secure guard to tons of data generated by IoT.

μNEST uses the combination of the above-mentioned parts to provide data of "integrity," "privacy free," and "purity." μWallet Core can intervene in the data integrity protection system from the nearby area of hardware level data gathering module. The hardware trust root pushes the security of the system and application to an unprecedented new height; the private data is masked just after gathered in the low level, leaving privacy alone forever inside the terminal. The privacy masked data can be exchanged or traded. The data exchange in untrusted environment will become the best experimental field for IoT + blockchain technology. μNEST will not only open source the blockchain code but also the hardware's layout and BOM.

μNEST will build up a Data Market , which will make the trusted data become the cornerstone of the new times.

Conclusion

We illustrate the core part of μNEST project from the low level of hardware and software to consensus algorithms. μNEST is an IoT + blockchain underlying framework. We provide the underlying blockchain, as well as the underlying hardware architecture,

which are all open source. We solved the trust chain problem with the hardware trust root, solved the TPS problem with µZone, solved the ledger and data storage problem with Proof-of-i/o, and solved the smart contract problem with a lightweight nVM. With the advancement the project, we will pick up representative scenarios and display application demos. In the future, based on our SDK, API, Layout, and BOM, all ecological co-constructors with their creative ideas can easily conduct further development for their own IoT + Blockchain Application (DApp).

Reference

https://en.wikipedia.org/wiki/Internet_of_things
https://en.wikipedia.org/wiki/Blockchain
https://steemit.com/dpos/@dantheman/dpos-consensus-algorithm-this-missing-white-paper
https://bitshares.org/technology/delegated-proof-of-stake-consensus/
https://en.wikipedia.org/wiki/Trusted_execution_environment
http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html
https://github.com/ethereum/wiki/wiki/White-Paper/f18902f4e7fb21dc92b37e8a0963eec4b3f4793a
http://wren.io/
https://lightning.network/