Using Blockchain Technology in Intelligent Control Systems

Alaa Albakour*

(Received 11 / 12 / 2023. Accepted 28 / 1 / 2024)

□ ABSTRACT □

Data security is especially important in automation systems that require the exchange of data over large distances through both telecommunications and telecommunications, which in turn is a fertile environment for piracy, not far from the attack on the Manouchehr nuclear reactor in Iran.

The use of Blockchain technology may increase the security of the data exchanged because it is difficult to intercept its messages through its encryption system, and it also ensures the integrity of this data.

In this research, a blockchain algorithm was proposed that works on the same microcontroller and uses the RS322 serial connection and was tested by using the Atmega2560 controller (an 8-bit processor with 8 KB SRAM memory) for a smart home system that measures temperature and humidity, as well as senses the presence of gas and measures the intensity of lighting in order to control the equipment of this house, the results proved that it is not possible to Use this technique on microcontrollers.

Keywords: Blockchain, Intelligent Home, DLT, IoT, SHA256 Algorithm.

***Master, Control and Automation Engineering, Aleppo University, Aleppo, Syria. Email: alaaalbakour241@gmail.com**

استخدام تقنية سلاسل الكتل في نظم التحكم الذكية

علاء البكور*

(تاريخ الإيداع 11 / 12 / 2023. قُبِل للنشر في 28/ 1 / 2024)

□ ملخّص □

يعتبر أمن البيانات مسألة مهمة خصوصاً في نظم الأتمتة التي تتطلب تبادل بيانات لمسافات كبيرة سواء من خلال الاتصالات السلكية أو اللاسلكية، والتي تعتبر بدورها بيئة خصبة للقرصنة، وليس بالبعيد الهجوم الذي تعرض له مفاعل منوشهر النووي في إيران.

إن استخدام تقنية **Blockchain** قد يزيد من أمان البيانات المتبادلة لما يتميز به من صعوبة اعتراض رسائله من خلال نظام التشفير الذي يتبعه، كما أنه يضمن سلامة هذه البيانات.

تم في هذا البحث اقتراح خوارزمية **Blockchain** تعمل على المتحكم نفسه وتستخدم بروتكول الاتصال التسلسلي **RS322** وتم اختبارها باستخدام متحكم ATmega2560 (وهو معالج 8 بت بذاكرة 8 KB SRAM) لنظام منزل ذكي يقوم بقياس درجة الحرارة والرطوبة وكذلك يتحسس وجود الغاز ويقيس شدة الإضاءة بغية التحكم بتجهيزات هذا المنزل، اثبتت النتائج أنه لا يمكن استخدام هذه التقنية على المتحكمات الصغرية.

الكلمات المفتاحية: سلاسل الكتل، منزل ذكي، تقنية الحسابات الموزعة، انترنت الأشياء، خوارزمية SHA256.

* **ماجستير - هندسة التحكم والأتمته – كلية الهندسة الكهربائية والالكترونية – جامعة حلب – حلب – سورية. ايميل: alaaalbakour241@gmail.com**

مقدمة:

في الآونة الأخيرة، أتاح التقدم التكنولوجي في شبكات التحكم والأنظمة المدمجة Embedded Systems العديد من تطبيقات المراقبة والتحكم في مجالات مختلفة بما في ذلك أنظمة إنترنت الأشياء (IoT) Internet of Things. والتي، توفر فعالية وفوائد اقتصادية هائلة لتركيب النظام، قابلية الصيانة، الموثوقية، قابلية التوسع، وقابلية التشغيل البيني. عادةً ما يتمكن المهاجم من الوصول إلى نظام الأتمته عبر اتصاله بشبكة أو بعض الوسائط الأخرى المستخدمة لتوصيله بمكونات أخرى. بالإضافة إلى ذلك، مع تزايد وجود نظام مضمن واحد يضم العديد من المكونات الموزعة المتصلة عبر شبكة، يجب على المصمم أن يعالج الأسئلة الأساسية حول الأمان عبر الشبكة وباستخدام بروتوكولات الاتصال المختلفة، أو أمان البروتوكول الذي يطلق عليه الحقل أو أمان الشبكة. والإجابة عن الاسئلة الأساسية المتعلقة بموضوعين لهما أهمية خاصة للأنظمة المضمنة وتطبيقاتها في شبكات الأتمته الصناعية: تبادل المفاتيح وتصميم بروتوكول التشفير. يتمثل المكون الرئيسي لإنشاء أنظمة الشبكات في تقنيات الاتصالات، التي تمكن جميع الأجهزة والآلات من الاتصال وتبادل البيانات معاً. في مثل هذه الحالة، يمكن للنظام مراقبة البيانات، جمعها، تبادلها وتحليلها، وتقديم خدمات قيمة تمكن بدورها شركات الصناعة من اتخاذ قرارات أكثر دقة وأسرع.

أهمية البحث وأهدافه:

إنّ الانتشار الواسع لشبكات انترنت الاشياء من ناحية التطبيق والجغرافيا، أدى لأن تصبح عمليات إرسال، استقبال، ومعالجة البيانات، عبر الانترنت، جزءاً لا يتجزأ من تكنولوجيا الأتمتة العملياتي عبر الانترنت، لذلك لابد من التأكد من حماية هذه البيانات وسيتم في هذا البحث اقتراح تطبيق تقنيات DLT (Blockchain) Distributed Ledger Technology واعتمادها في تصميم شبكات الأتمتة المعتمدة على IOT.

طرائق البحث ومواده:

سيتم في هذا البحث استخدام متحكم ATmega2560 (وهو معالج 8 بت بذاكرة 8 KB SRAM) لنظام منزل ذكي يقوم بقياس درجة الحرارة والرطوبة وكذلك يتحسس وجود الغاز ويقيس شدة الإضاءة ويرسلها لمتحكم آخر بغية التحكم بتجهيزات هذا المنزل، وسيتم استخدام تقنية Blockchain من خلال خوارزمية تم تصميمها لذلك، ثم سيتم اختبارها وفق سيناريوهات وتحليل النتائج.

1- الدراسات المرجعية

أدى الانتقال إلى Industry 4.0 إلى زيادة المخاطر التي تتعرض لها الشبكة بسبب بروتوكولات الأمان الضعيفة وغير الفعالة في أجهزة إنترنت الأشياء [1]. علاوة على ذلك فإن عدم تجانس الأجهزة المنفذة في مصنع ذكي واحد يقيد تنفيذ بروتوكول أمان مشترك للشبكة على مستوى النظام.

تشكل هجمات البوت نت خطراً مزدوجاً على شبكة الصناعة 4.0 [2]، حيث يمكن أن يتم اختراق أمان البيانات، من خلال قيام المهاجمين بتعديل وحقن البيانات الفاسدة، مما يؤثر على أداء الشبكة، كما يمكن أن يؤدي رفض الهجمات الموزعة Distributed Denial of Service (DDoS) إلى زيادة المخاطر على الأداء التشغيلي للشبكة بالكامل، مما يؤدي إلى فشل حساب بيانات الجهاز ونمذجة البيانات وتحليل الأداء التشغيلي [3].

تكتشف أنظمة كشف التسلل المستندة إلى السحابة باستخدام الذكاء الاصطناعي (AI) وتحدد الحالات الشاذة في حركة مرور إنترنت الأشياء الصناعية، لكنها بشكل عام غير فعالة للتطبيقات في الزمن الحقيقي الصلبة مثل مصانع التصنيع التي تعتمد على الكفاءة العالية لتلبي طلبات المستهلكين [4]. تم استكشاف تقنية DT مؤخراً للكشف المبكر عن السلوك الضار في حركة مرور الشبكة الناشئة عن أجهزة إنترنت الأشياء المادية.

تم في البحث [5] تقديم طريقة اكتشاف الروبوتات من خلال تنفيذ نهج التعلم الموحد لاكتشاف شذوذ حركة المرور. حيث تم تدريب العديد من النماذج المحلية مباشرة على أجهزة IoT – Edge باستخدام نموذج Deep Neural Network لضمان خصوصية البيانات. بعد عدة جولات من التدريب على الأجهزة المحلية، تم تحديث النموذج العالمي باستخدام النتائج المجمعة للنماذج المحلية. تُستخدم مجموعة بيانات Bot – IoT لمحاكاة سيناريو الروبوتات التي تحقق معدلات اكتشاف عالية ولكنها تتطلب وقت تدريب طويل مقارنة بنماذج التدريب المركزية الأخرى.

تم في البحث [6] التركيز على تحديد سلوك الروبوتات قبل الهجوم الإلكتروني على الشبكة، تضمنت مجموعة البيانات التي تم إنشاؤها ثلاث مجموعات بيانات متاحة للجمهور 33 نوعاً من الفحص و60 هجوماً من هجمات DDoS. باستخدام نهج التعلم الآلي المزدوج، يقوم نموذج ResNet-18 الأول بمسح أجهزة إنترنت الأشياء المحلية لمسح النشاط الذي يشير إلى سلوك الروبوتات. يركز نموذج ResNet-19 الثاني على تحديد هجمات DDoS وتنبيه النظام للهجوم المستند إلى الروبوتات. على الرغم من أن البحث يشير إلى أن الكشف المبكر عن سلوك الروبوتات أمر ضروري لأمن الشبكة، إلا أن نهج التعلم الآلي المزدوج لا يزال يعتمد على تحديد هجوم DDoS لسلوك اكتشاف الروبوتات.

ركز البحث [7] على تحديد شبكات الروبوت في بيئة مدينة ذكية. يحدد نموذج التعلم العميق المكون من مستويين أولاً استعلامات DNS الأكثر شيوعاً التي يتم إجراؤها على اتصالات Ethernet جنباً إلى جنب مع قيمة العتبة، كما تصنف خوارزمية إنشاء المجال القائمة على التعلم العميق عناوين المجال الحميدة والخبيثة.

تركز حلول اكتشاف الحالات الشاذة المستندة إلى Blockchain في بيئات إنترنت الأشياء على تأمين الشبكة باستخدام حلول موزعة وقابلة للتطوير. حيث تم في البحث [8] تنفيذ طريقة تخفيف هجوم DDoS القائمة على التعلم الآلي من خلال التسجيل المبدئي لجميع الأجهزة في الشبكة اللامركزية. اقترحت الدراسة إخراج الأجهزة الخبيثة التي فشلت في المصادقة مع شبكة Blockchain. يتطلب كل جهاز متفاعل التسجيل المسبق مع الشبكة قبل الاتصال البيني بالأجهزة الأخرى.

اقترحت الدراسة [9] تنفيذ الروبوتات الحديثة ببنية P2P لزيادة فرصتها في إصابة منطقة أوسع من أجهزة إنترنت الأشياء، كما اقترح المؤلفون طريقة ديناميكية لاكتشاف الروبوتات باستخدام Blockchain لبناء الثقة بين أجهزة إنترنت الأشياء ومقدمي خدمات الإنترنت. يتم الحفاظ على خصوصية الممثلين المختلفين من خلال ربط معرفات زائفة لكل عضو. يتطلب نموذج الكشف إجماعًا بين جميع أعضاء Blockchain لإخراج الجهاز من الشبكة عند اكتشاف نشاط ضار.

اقترحت الدراسة [10] شبكة Blockchain خفيفة الوزن ومرخصة لإنترنت الأشياء حيث أن الأجهزة التي يتم فيها تعيين مفتاح عام لكل جهاز استناداً إلى سمات هويته لتصفية الأجهزة المصرح بها من الطلبات غير الصالحة. لتجنب التأخير في معالجة البيانات في البيئة اللامركزية، تحدد خوارزمية نقطة قرار السياسة قرارات Blockchain في الزمن الحقيقي خارج السلسلة.

وفي الدراسة [11] تم اقتراح خوارزمية Blockchain ثنائي الاتجاه مقاوم للهجوم باستخدام وظائف تجزئة الحرباء لبيئة إنترنت الأشياء. توفر هذه الخوارزمية قابلية عالية للتوسع ومقاومة هجوم الشبكة اللامركزية.

وفي الدراسة [12] تم اقتراح إطار عمل رقمي مزدوج Digital Twin Framework (DT) يدعم تقنية Blockchain للكشف المبكر عن نشاط الروبوتات في بيئة إنترنت الأشياء. يدمج الإطار DT، وأجهزة إنترنت الأشياء المادية، ونموذج التعلم العميق، وBlockchain، والعقود الذكية لتأمين تدفق البيانات لبيئة المصنع الذكي، يطبق إطار العمل Blockchain خاص يديره بائع أمن لتسجيل DTs، وعقدة افتراضية، PA، مسؤولة عن مزامنة بيانات DT بشكل آمن مع الأجهزة المادية باستخدام العقود الذكية. تقوم مراقبة حركة مرور الشبكة باستخدام التعلم العميق بفحص وتحليل كل من حركة المرور المشفرة وغير المشفرة باستخدام رؤوس الحزم. يفحص النموذج الكشف المبكر عن سلوك الروبوتات، ومعالجة المخاوف المتزايدة من الهجمات الإلكترونية القائمة على الروبوتات على شبكة الصناعة 4.0. حيثُ تتيح بيئة DT الخاصة بـ Industrial Internet of Things (IIoT) تنفيذ بروتوكولات الأمان القوية التي تستفيد من الموارد من مراكز البيانات المتطورة. وتسجل Blockchain الخاصة DTs على أنها معاملات في الكتل تمنع العقد الخبيثة من حقن البيانات الفاسدة في تدفق البيانات، مما يؤثر على سلامة البيانات التي تم جمعها. يفحص نموذج اكتشاف الروبوتات للتعلم العميق عناوين IP الفريدة والتوصيلات نصف المفتوحة، مما يشير إلى قناة اتصال مفتوحة بين الجهاز وخادم التحكم والتحكم في الروبوتات. تم تصميم Digital Twins of the Smart Factory على طبقة الحافة؛ يقومون بمزامنة البيانات مع أجهزة إنترنت الأشياء الخاصة بهم. وهو عقد ذكي تم تصميمه لتأمين التسجيل والمصادقة لجميع كيانات الشبكة مع شبكة Blockchain الخاصة. يقوم Digital Twins and Packet Auditor بالمصادقة والتحقق بشكل آمن من خلال شبكة Blockchain ومزامنة البيانات، مما يضمن عدم تعديل بيانات الحزمة أثناء الإرسال. تمت مناقشة سير العملية لكل عملية بالتفصيل. يفحص النموذج القائم على التعلم العميق حركة مرور كل من الحزم المشفرة وغير المشفرة ويحدد الأجهزة المصابة بالبرامج الضارة. تمنع سياسة عزل الجهاز إصابة التوائم الرقمية الأخرى، مما يمنع نمو نشاط الروبوتات.

تم في جميع الدراسات التي تم استعراضها قتراح بنية أنترنت الأشياء والتي تعتمد على مخدمات وسيطة في بنية الشبكات الصناعية، سيتم في هذه الدراسة اقتراح خوارزمية Blockchain تعمل على المتحكم نفسه وتستخدم برتوكل الاتصال التسلسلي RS322.

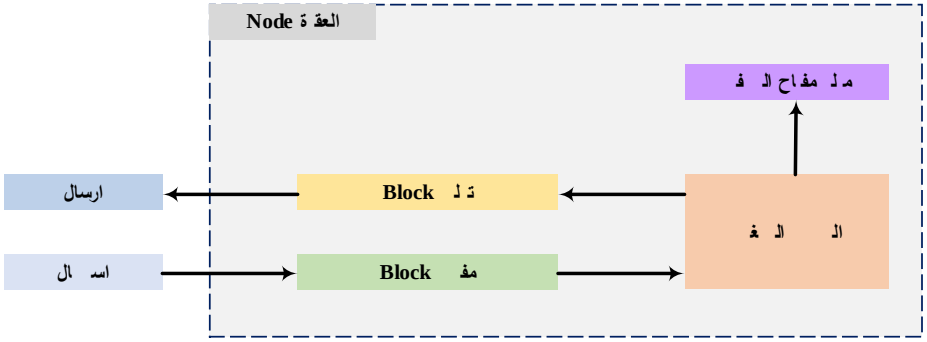
2- بنية العقدة المقترحة في Blockchain

تعتبر Blockchain عبارة عن دفتر أستاذ رقمي لامركزي يحتوي على سلسلة منتظمة من الكتل المتصلة ببعضها البعض، ومن هنا يأتي الجزء المتسلسل من Blockchain. تحتوي هذه الكتل على بيانات تتعلق لغرض معين (المعلومات المصرفية للعملاء، وتفاصيل المريض، وإمكانية تتبع مصدر التصنيع، والمعرف الآمن، وما إلى ذلك). يتم تخزين كل معاملة تحدث في الكتلة وإضافتها إلى Blockchain. تحتوي هذه الكتل على قائمة من السجلات المتزايدة في شكل قاعدة بيانات موزعة. يعنى هذا الفصل بتصميم شبكة Blockchain باستخدام المتحكمات الصغرية، يبين الشكل (1) بنية العقدة المقترحة في شبكة Blockchain، حيثُ تتكون من:

1. المتحكم الصغري: الوحدة الأساسية التي تقوم بقراءة الحساسات وتشفير البيانات وكذلك استقبال هذه البيانات وفك تشفيرها.
2. مولد مفتاح التشفير: يقوم المعالج بتوليد مفتاح التشفير اللازم لتشكيل الكتلة.

3. توليد Block: يقوم المعالج بتوليد تابع التجزئة المناسب للكتلة واكمال توليد الكتلة.
4. مفسر Block: عندما تكون الوحدة الأساسية مستقبلة للبيانات، فلا بد أن يقوم المعالج الخاص بها بتفسير الكتلة التي تم استقبالها.
5. الارسال: بعد اكتمال انشاء الكتلة، تقوم الوحدة بإرسال الكتلة المولدة.
6. الاستقبال: ففي حالة الوحدة المستقبلة، فلا بد من تأمين عملية الاستقبال للكتلة.

ويبين الجدول (1) المكونات البرمجية للعقدة المستخدمة.



الشكل (1) بنية العقدة في شبكة Blockchain

الجدول (1) المكونات البرمجية للعقدة

المكون	الوصف
Index	رقم تعريف العقدة
proof_number	رقم الاثبات والمصادقة
prev_hash	تابع التجزئة للعقدة السابقة
Data	البيانات التي سيتم ارسالها
data_len	طول البيانات التي سيتم ارسالها
Timestamp	الوقت المستخدم لقياس مدة الارسال
HASH_SIZE	حجم تابع التجزئة التي سيتم استخدامه من أجل توليد مفاتيح التشفير

إنّ ناتج خوارزمية التجزئة مثل SHA256 سيكون 256 بت، أي 32 بايت، والذي يتم عرضه على شكل 64 حرفاً أبجدياً رقمياً. تظهر جميع المخرجات بشكل عشوائي تماماً ولا تقدم أي معلومات حول المدخلات التي أنشأتها.

3- نموذج شبكة Blockchain المقترحة

تُعرّف Blockchain بأنها عبارة عن تقنية Distributed Ledger تحتوي على سلسلة منتظمة من الكتل المتصلة ببعضها البعض، حيث تحافظ Blockchain على سلامة بياناتها من خلال التحقق من صحة كل كتلة بمساعدة التشفير، كما ويقوم كل مشارك في Blockchain بالتحقق من صحة الكتلة، لا يمكن تعديل البيانات في Blockchain دون موافقة المشاركين. يُعرف هذا بإثبات العمل Proof of work، يبين الشكل (2) نموذج شبكة Blockchain، حيثُ تتكون من توصيل عدد من العقد وارسال المعلومات بين هذه العقد ضمن سلسلة الكتل، كما يوضح الجدول 2 المكونات الأساسية للسلسلة.

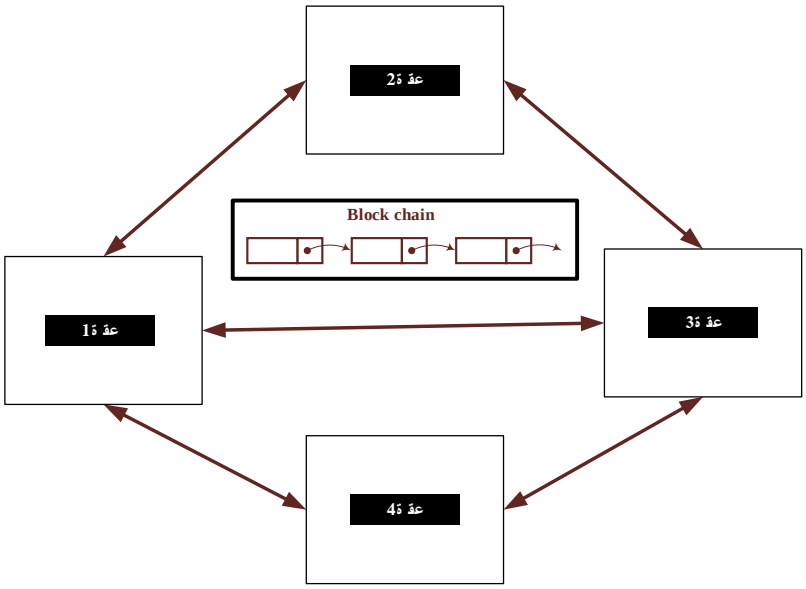
الجدول (2) مكونات السلسلة الأساسية

المكون	الوصف
chain	السلسلة
chain_len	لتخزين طول السلسلة
current_data	لتشكيل البيانات الحالية التي سيتم ارسالها
current_data_len	طول البيانات الحالية التي سيتم ارسالها

4- تصميم سلسلة الكتل المقترحة Blockchain Design

4-1 عملية البناء

يتم بناء سلسلة الكتل انطلاقاً من تشكيل الكتل بشكل تدريجي وفق الشكل (3) بإنشاء سلسلة فارغة، حيث، يتم تخصيص تابع التجزئة اللازم لعملية التشفير وتهيئته بقيمة ابتدائية صفرية، ومن ثم البدء بإنشاء الكتلة وفق الشكل (4)، حيثُ يتم التخصيص الديناميكي لكتلة جديدة وذلك بتحديد طول السلسلة chain_length، رقم الاثبات proof_number، تابع التجزئة السابق prev_hash، البيانات الحالية current_data، وطول البيانات الحالية current_data_len، ومن ثم الغاء التخصيص للبيانات من أجل تشكيل بيانات جديدة للكتلة الجديدة بقيم ابتدائية صفرية، ومن ثم إضافة العقدة إلى السلسلة.



الشكل (2) نموذج شبكة Blockchain

4-2- إضافة block جديدة للسلسلة

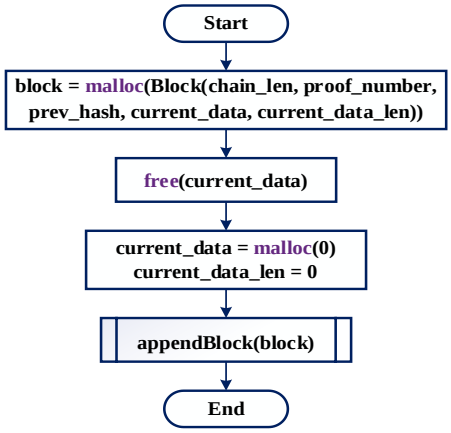
يبين الشكل (5) المخطط الصندوقي لإجرائية إضافة الكتلة الجديدة إلى السلسلة. حيثُ يتم إعادة التخصيص الديناميكي للكتلة (من أجل إمكانية تعديل حجم الكتلة بما يتناسب مع البيانات المخزنة وطولها)، يتم ادراج الكتلة الجديدة في نهاية السلسلة في حال تم إعادة التخصيص بنجاح، فيما عدا ذلك تفشل عملية الإضافة.

4-4- التحقق من صلاحية الكتلة

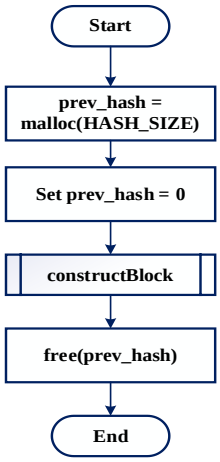
يبين الشكل (6) المخطط الصندوقي لإجرائية التحقق من صلاحية الكتلة المضافة إلى السلسلة. حيثُ يتم إعادة التحقق من الكتلة الحالية والكتلة السابقة لها من حيث (تتابع أرقام الكتل (الفهارس)، تابع التجزئة اللازم لربط الكتلتين معاً، رقم الاثبات الخاص بالكتلتين) فشل التحقق، يؤدي إلى عدم قبول الكتلة المضافة للسلسلة.

4-5- توليد البيانات

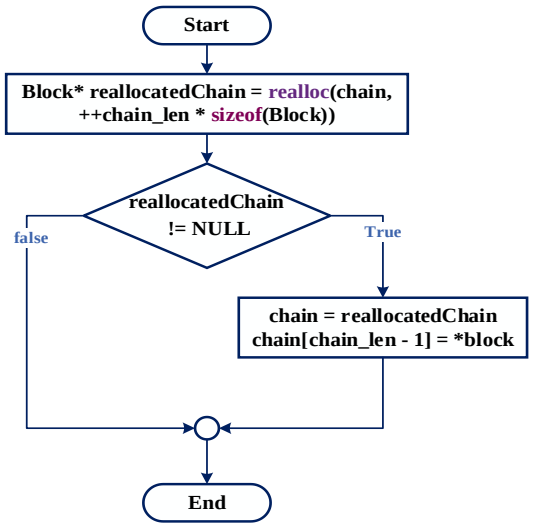
يبين الشكل (7) المخطط الصندوقي لخوارزمية توليد البيانات التي سيتم ادراجها في كل كتلة على حدى، والتي سيتم ضمها لسلسلة الكتل.



الشكل (4) بناء Construct Block



الشكل (3) بناء Genesis

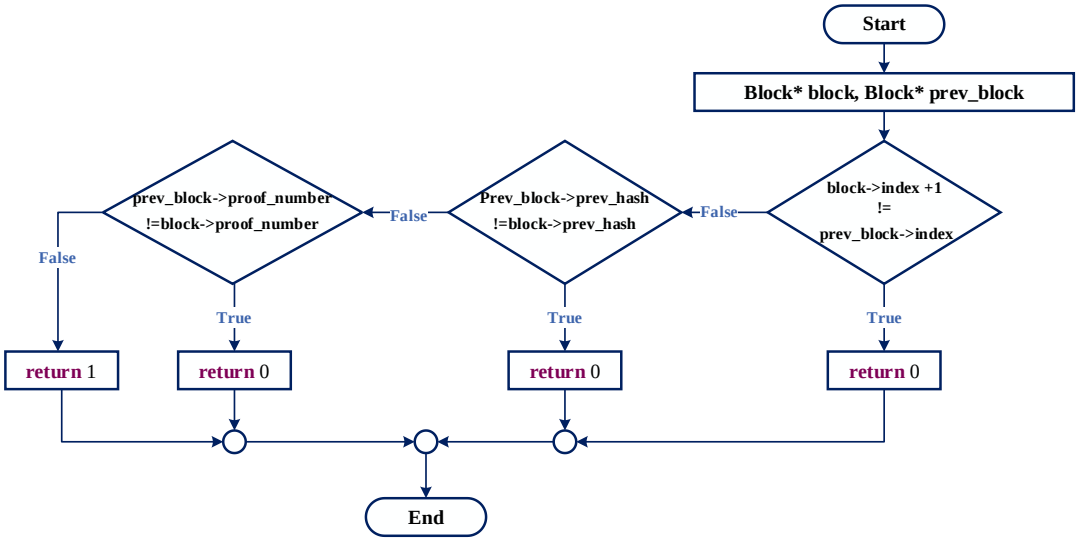


الشكل (5) إضافة عقدة للسلسلة append Block Flowchart Diagram

4-6- اثبات العمل Proof of Work

يصف إثبات العمل (PoW) نظاماً يتطلب قدراً غير مهم ولكن ممكنًا من الجهد لردع الاستخدامات العبثية أو الخبيثة لقوة الحوسبة، مثل إرسال رسائل بريد إلكتروني غير مرغوب فيها أو إطلاق هجمات رفض الخدمة.

إثبات العمل (PoW) هو آلية إجماع لامركزية تتطلب من أعضاء الشبكة بذل جهد لحل لغز رياضي تعسفي لمنع أي شخص من التلاعب بالنظام. يتطلب إثبات العمل على نطاق واسع كميات هائلة من الطاقة، والتي تزداد فقط مع انضمام المزيد من عمال المناجم إلى الشبكة. تتطلب PoW عقداً على شبكة لتقديم دليل على أنها أنفقت قوة حسابية (أي العمل) من أجل تحقيق توافق في الأداء بطريقة لامركزية ومنع الجهات الفاعلة السيئة من تجاوز الشبكة، يظهر الشكل (8) الخوارزمية المتبعة لإثبات العمل.



الشكل (6) التحقق من الصلاحية check Validity Flowchart Diagram

4-7- الارسال في الشبكة

يبين الشكل (9) المخطط الصندوقي لإجرائية ارسال الكتل.

النتائج والمناقشة:

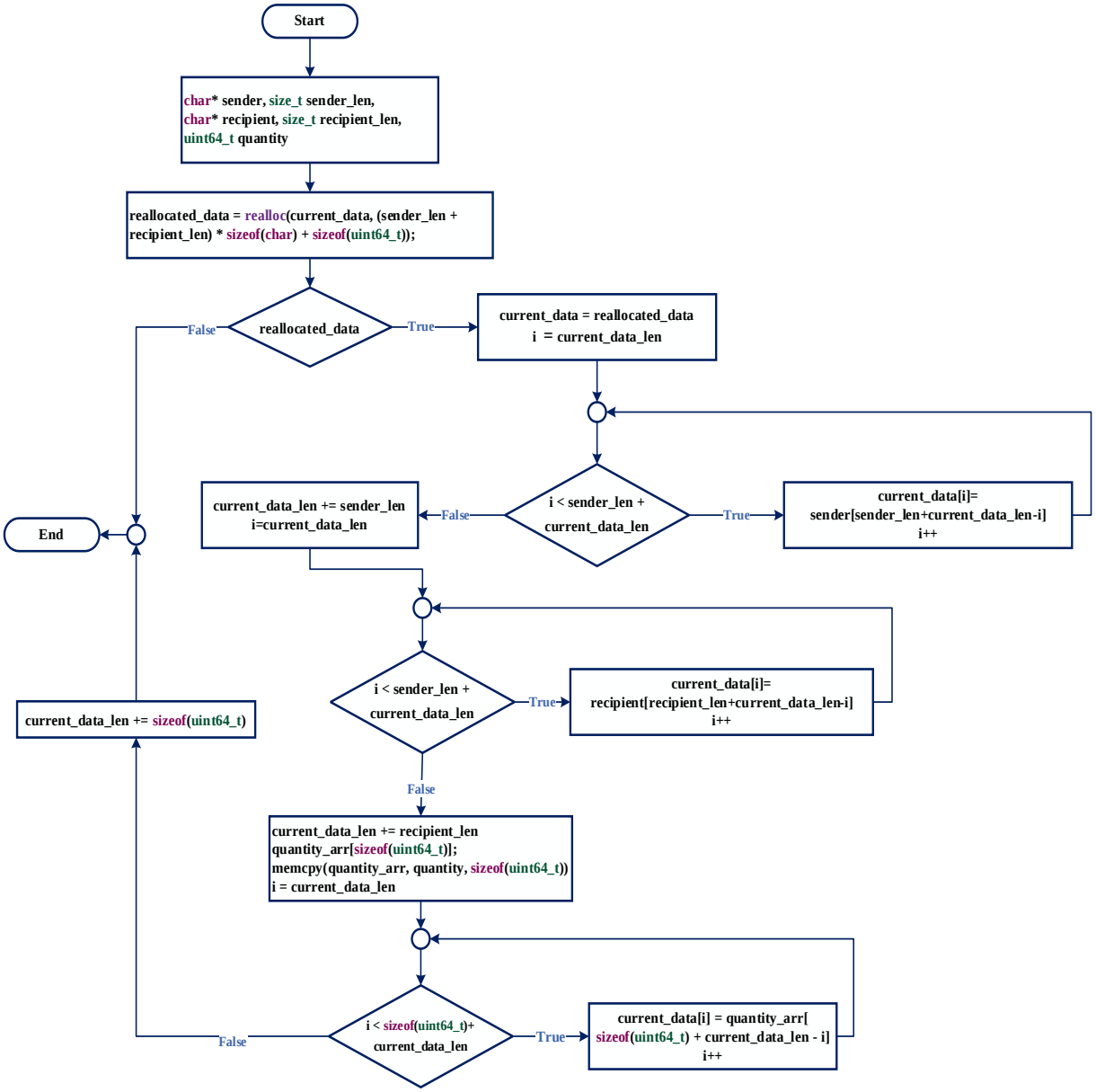
تم اختبار تصميم شبكة Blockchain باستخدام متحكم ATmega2560 (وهو معالج 8 بت بذاكرة 8 KB SRAM) لنظام منزل ذكي يقوم بقياس درجة الحرارة والرطوبة وكذلك يتحسس وجود الغاز ويقيس شدة الإضاءة بغية التحكم بتجهيزات هذا المنزل.

البارامترات المقاسة في هذا السيناريو (درجة الحرارة tem وهي من النمط float، الرطوبة hum وهي من النمط float، الغاز gaz وهي من النمط Boolean، شدة الإضاءة وهي من النمط integer)، تم الاختبار من خلال عدة سيناريوهات.

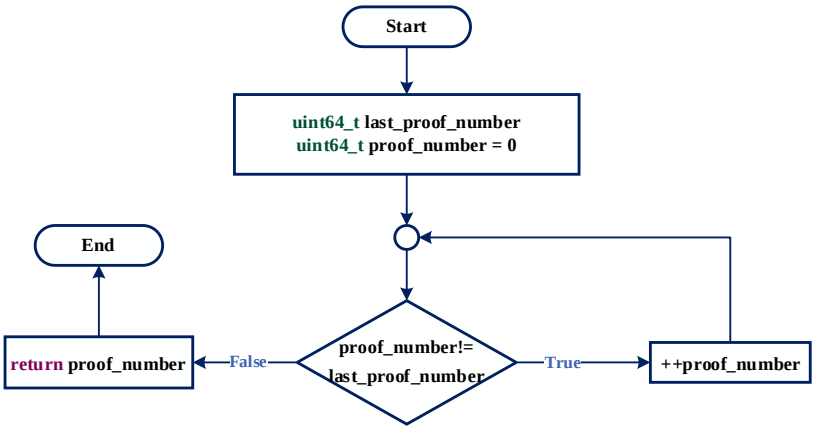
1- السيناريو الأول

سيتم إرسال قيم الحساسات وفق الخوارزمية المقترحة بطول هاش 32bits، سيتم في هذا السيناريو عرض كامل النتائج ومناقشتها باستفاضة ومن ثم في باقي السيناريوهات سيتم انتقاء النتائج المهمة. يتم عرض النتاج من خلال قراءتها عبر المنفذ التسلسلي، والتي تم تشكيلها كما هو وارد:

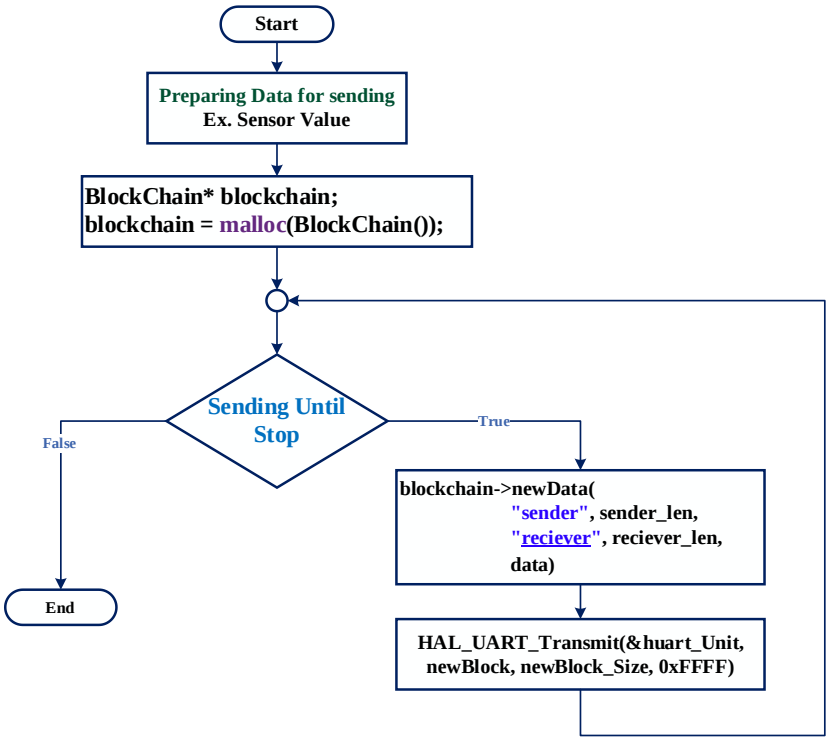
13:01:04.167 -> >>>>>>>>>> Begin Read Sensors >>>>>>>>>>>>>>>>>>>>>>>>
13:01:04.261 -> tem= 30 hum= 95.00
13:01:04.261 -> Gaz= 1
13:01:04.261 -> light= 687
13:01:04.308 -> >>>>>>>>>>>> Start BlockChain setup >>>>>>>>>>>>>>>>>>>>>>
13:01:04.355 -> HASH SIZE: 32
13:01:04.402 -> Appending block
13:01:04.402 -> Realocated chain nillnes: 1
13:01:04.450 -> >>
13:01:04.497 -> Printing out all blocks:
13:01:04.544 -> 0,0,0000000000000000000000000000000000, >>> [], 664
13:01:04.874 -> <<<<<<<<<<< finsh setup <<<<<<



الشكل (7) توليد بيانات جديدة new Data Flowchart Diagram



الشكل (8) اثبات العمل proof of work Flowchart Diagram



الشكل (9) الارسال في الشبكة Sending Flowchart Diagram

تم تشكيل العرض وفق مرحلتين:

- الأولى هي مرحلة الإعداد والتي تتضمن قراءة الحساسات للتأكد من عملها، ومن ثم عرض حجم Hash، والبدء بتكوين البلوك الأول وعند الانتهاء منه طباعته، ومن ثم يظهر سطر ليبين انتهاء مرحلة الإعداد، يتم ترقيم كل سطر بتوقيت إرساله.
- المرحلة الثانية هي مرحلة قراءة الحساسات الدورية وتشكيل السلاسل كما هو واضح فيما يلي:

13:01:05.015 <- tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
13:01:05.063 <- Appending block
13:01:05.063 <- Realocated chain nillnes: 1
13:01:05.110 <- index: 1, proof_number: 0, previous hash:
E100000A863E5009A6003E5519547A261806D7C2211F0, data: 007A2100800000, timestamp: 1342
13:01:05.251 <- Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1

13:01:05.298 <- Appending block
13:01:05.346 <- Realocated chain nillnes: 1
13:01:05.393 <- index: 2, proof_number: 0, previous hash: E100000B06476009A600476619C1700180DA7C2211F0, data: 007A219A0005C214C, timestamp: 1607
13:01:05.534 <- light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 686
13:01:05.581 <- Appending block
13:01:05.581 <- Realocated chain nillnes: 1
13:01:05.675 <- index: 3, proof_number: 0, previous hash: E100000B06577009A6005777183F8BB6180588C2211F0, data: 007A21655206B636F6C, timestamp: 1879
13:01:07.418 <- tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
13:01:07.465 <- Appending block
13:01:07.512 <- Realocated chain nillnes: 1
13:01:07.559 <- index: 10, proof_number: 0, previous hash: E1000007A7BBE009A600BBEE08F95F8180A89C2211F0, data: 007A2100800000, timestamp: 3771
13:01:07.700 <- Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
13:01:07.748 <- Appending block
13:01:07.748 <- Realocated chain nillnes: 1
13:01:07.842 <- index: 11, proof_number: 0, previous hash: E1000007A7C4F009A600C4FF04095F8180599C2211F0, data: 007A219A0005C214C, timestamp: 4036
13:01:07.983 <- light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 686
13:01:08.030 <- Appending block
13:01:08.030 <- Realocated chain nillnes: 1
13:01:08.077 <- index: 12, proof_number: 0, previous hash: E1000007A7D410009A600D41010156C001801FCC2211F0, data: 007A2161655206B636F, timestamp: 4308

حيثُ من أجل كل حساس يتم إرسال القراءة ومن ثم تشكيل البلوك الخاص به ثم البدء بعملية إضافة هذا البلوك للسلسلة وتظهر نتيجة الإضافة بعد ذلك والتي تعني أنه تم إضافة البلوك إلى السلسلة، يتم بعدها عرض البلوك المضاف من خلال عرض بارامترات هذا البلوك وهي:

1. Index: رقم فهرسة البلوك في السلسلة.
2. proof_number: رقم اثبات العمل المستخدم في منع المهاجمين وتصعيب عملهم، والذي يتم توليده انطلاقاً من previous_hash.
3. previous hash: الهاش السابق الذي يساعد بربط البلوكات داخل السلسلة.
4. Data: المعطيات التي تم تشفيرها وهي قيمة الحساسات مشفرة.
5. Timestamp: التوقيت المحسوب الذي تمت عنده قراءة الحساس.

أظهرت النتائج ثبات proof_number عند الصفر وكذلك previous hash، برغم أن تشفير المعطيات تم لكل حساس إلا أنه بقي ثابتاً لكل حساس مع ثبات proof_number وprevious hash، مع استمرار تغير التوقيت، استمر الوضع هكذا حتى العينة رقم 48 بدأ توليد الـ proof_number وبدأت previous hash بالتغير إلا أنه لم يتم إضافة سوى هذه العينة للسلسلة، استمر النظام بتوليد البلوكات وحسابها بدون إضافتها إلى السلسلة حتى العينة 66 كما هو واضح:

13:01:17.883 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 687
13:01:17.930 -> Appending block
13:01:17.930 -> Realocated chain nillnes: 1
13:01:18.024 -> index: 48, proof_number: 18321341163347116184, previous hash: E107AB7AAC7A77E37009A6007E37375DA1A4911180F31AC2211F0, data: 007A215206B636F6C62, timestamp: 14206
13:01:18.166 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00

13:01:18.213 -> Appending block
13:01:18.260 -> Realocated chain nillnes: 1
13:01:18.307 -> index: 49, proof_number: 11070246323900997275, previous hash: E106A610314B11AF38009A600AF38384981A48E180B11AC2211F0, data: 007A2100800000, timestamp: 14511
13:01:18.495 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> **1**
13:01:18.542 -> Appending block
13:01:18.542 -> Realocated chain nillnes: 0
13:01:18.589 -> Unable to reallocate memory for chain APP
13:01:18.636 -> index: 50, proof_number: 8318770821577315225, previous hash: E1035FCCC04B11D739009A600D739394491A48E180621AC2211F0, data: 007A219A0005C214C, timestamp: 14807
13:01:18.825 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> **688**
13:01:20.334 -> Appending block
13:01:20.334 -> Realocated chain nillnes: 0
13:01:20.381 -> Unable to reallocate memory for chain APP
13:01:20.428 -> index: 51, proof_number: 5485665824035373742, previous hash: E1000AE214B11941009A600941414FA1948E180131AC2211F0, data: 007A215206B636F6C62, timestamp: 16649
13:02:05.517 -> index: 63, proof_number: 3108366801109840, previous hash: E10BBB34B110F1009A6000F1F10461648E1805F16C2211F0, data: 007A2161655206B636F, timestamp: 61696
13:02:05.659 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> **30.00**
13:02:08.301 -> Appending block
13:02:08.348 -> Realocated chain nillnes: 0
13:02:08.395 -> Unable to reallocate memory for chain APP
13:02:08.442 -> index: 64, proof_number: 217020518518440771, previous hash: E10333434B1180FC009A60080FCFC0F71548E1801016C2211F0, data: 007A2100800000, timestamp: 64640
13:02:08.631 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> **1**
13:02:11.270 -> Appending block
13:02:11.270 -> Realocated chain nillnes: 0
13:02:11.333 -> Unable to reallocate memory for chain APP
13:02:11.364 -> index: 65, proof_number: 15063145520741601745, previous hash: E10BBBD14B11FC7109A600FC770A81548E180C115C2211F0, data: 007A219A0005C214C, timestamp: 67580
13:02:11.506 -> 687
13:02:14.146 -> Appending block
13:02:14.146 -> Realocated chain nillnes: 0
13:02:14.192 -> Unable to reallocate memory for chain APP
13:02:14.240 -> index: 66, proof_number: 59374520500224, previous hash: A304C215C0009A611EDC22001000010302120 5151C211E0, data:
232323232323232323232323232323232323232323232323232323232323232323232323232323232323232
323232323232323232323232323232323232323232323232323232323232323232323232323232323232323
232323232323232323232323232323232323232323232323232323232323232323232323232323232323232
32323232323232323232323232323232323232323232323232323232323232323, timestamp: 17487994994557696

بدأ عمل النظام في التوقيت **13:01:05.015 وتم تشكيل آخر عينة في التوقيت** 13:02:14.240 أي استمر التوليد لمدة 1 دقيقة 9 ثواني تقريباً، تم حساب البلوك الأول بشكل كامل في التوقيت 13:01:17.741 أي بعد 12 ثانية تقريباً، وكما هو ملاحظ لم يتم بعد هذه العينة إضافة البلوك المولد إلى السلسلة، يمكن ملاحظة أن الرسائل في بعض الأحيان لم تكن كاملة، مع آخر بلوك تم توليده، بعد البلوك ذي الفهرسة 66 تم إعادة تشغيل المتحكم بمفرده، يمكن تفسير ما حدث لمحدودية الذاكرة SRAM في المتحكم، خصوصاً أنه تم استخدام المصفوفات الديناميكية في بناء البلوكات والسلسلة، يظهر فيما يلي كيف أن المتحكم أعاد إقلاعه وعاد الأمر كما هو منذ البدء:

13:02:18.720 -> >>>>>>>>>>>>>> Begin Read Sensors >>>>>>>>>>>>>>>>>>>>>>
13:02:18.815 -> tem= 30 hum= 95.00
13:02:18.815 -> Gaz= 1
13:02:18.815 -> light= 688
13:02:18.862 -> >>>>>>>>>>>>>> Start blockChain setup >>>>>>>>>>>>>>>>>>>>>
13:02:18.909 -> HASH SIZE: 32
13:02:18.956 -> Appending block
13:02:18.956 -> Realocated chain nillnes: 1
13:02:19.003 -> >>
13:02:19.050 -> Printing out all blocks:
13:02:19.097 -> 0,0,00000000000000000000000000000000,>>>[],665
13:02:19.428 -> <<<<<<<<<<<<<< finsh setup <<<<<<

بدأ التوليد عند البلوك بشكل كامل 48 وتم إضافته للسلسلة إلا أن البلوك 49 لم يضاف برغم إنشائه.

13:02:32.573 -> index: 48, proof_number: 3151552188522697880, previous hash: E109049007A79437009A6009437375DA1A4911180F31AC2211F0, data: 007A21655206B636F6C, timestamp: 14228
13:02:32.762 -> tem= >>>>>>>>>>>>>> >>>>>>>>>>>>>> 30.00
13:02:32.808 -> Appending block
13:02:32.808 -> Realocated chain nillnes: 1
13:02:32.903 -> index: 49, proof_number: 15297680121074404380, previous hash: E104750314B11C338009A600C338384981A48E180B11AC2211F0, data: 007A2100800000, timestamp: 14531
13:02:33.044 -> Gaz= >>>>>>>>>>>>>> >>>>>>>>>>>>>> 1
13:02:33.091 -> Appending block
13:02:33.138 -> Realocated chain nillnes: 0
13:02:33.138 -> Unable to reallocate memory for chain APP
13:02:33.232 -> index: 50, proof_number: 3684115099788646523, previous hash: E109B29FA8E4B11E939009A600E939394491A48E180621AC2211F0, data: 007A219A4343003E3C, timestamp: 14825

بدأت الرسائل تظهر بشكل ناقص وتم إعادة تشغيل المتحكم.

13:03:25.936 -> index: 65, proof_number: 15063145520741601745, previous hash: E10BBBD14B111F8109A6001F880A81548E180C115C2211F0, data: 007A219A4343003E3C, timestamp: 67615
13:03:26.124 -> 687
13:03:28.717 -> Appending block
13:03:28.717 -> Realocated chain nillnes: 0
13:03:28.764 -> Unable to reallocate memory for chain APP
13:03:28.811 -> index: 66, proof_number: 59373981532160, previous hash: A303C3E0043439A611EDC22001000010302120515 1C211E0, data: 2323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232 3232323232323232323232323232323232323232323232323232323232323232323232323232323232323232323 2323232323232323232323232323232323232323232323232323232323232323232323232323232323232323232 3232323232323232323232323232323232323232323232323232323232323232323, timestamp: 17487994994688768
13:03:33.305 -> >>>>>>>>>>>>>> Begin Read Sensors >>>>>>>>>>>>>>

2- السيناريو الثاني

سيتم إرسال قيم الحساسات وفق الخوارزمية المقترحة بطول هاش 8bits، أظهرت النتائج ثبات proof_number عند الصفر وكذلك previous hash، برغم أن تشفير المعطيات تم لكل حساس إلا أنه بقي ثابتاً لكل حساس مع ثبات proof_number وprevious hash، مع استمرار تغير التوقيت، استمر الوضع هكذا حتى العينة رقم 70 بدأ توليد الـ proof_number وبدأت previous hash بالتغير إلا أنه لم يتم إضافة أي عينة للسلسلة، استمر النظام بتوليد البلوكات وحسابها بدون إضافتها إلى السلسلة حتى العينة 85 كما هو واضح:

12:48:18.569 -> index: 69, proof_number: 0, previous hash: E100000F1C, data: 007A2164657461636F6C, timestamp: 17539
12:48:18.663 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
12:48:19.984 -> Appending block
12:48:19.984 -> Realocated chain nillnes: 0
12:48:20.031 -> Unable to reallocate memory for chain APP
12:48:20.078 -> index: 70, proof_number: 0, previous hash: E100000F1C, data: 007A2100800000, timestamp: 19136
12:48:20.220 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
12:48:20.267 -> Appending block
12:48:20.267 -> Realocated chain nillnes: 0
12:48:20.314 -> Unable to reallocate memory for chain APP
12:48:20.361 -> index: 71, proof_number: 606982522415637860, previous hash: E106F636174F1C, data: 007A219A1A22012420, timestamp: 19355
12:48:20.503 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 695
12:48:20.550 -> Appending block
12:48:20.550 -> Realocated chain nillnes: 0
12:48:20.597 -> Unable to reallocate memory for chain APP
12:48:20.691 -> index: 72, proof_number: 296693317150703616, previous hash: E1011000F1C, data: 007A212064657461636F, timestamp: 19673
12:48:20.833 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
12:48:20.880 -> Appending block
12:48:20.880 -> Realocated chain nillnes: 0
12:48:20.927 -> Unable to reallocate memory for chain APP
12:48:21.022 -> index: 73, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A2100800000, timestamp: 20018
12:48:21.116 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
12:48:21.163 -> Appending block
12:48:21.163 -> Realocated chain nillnes: 0
12:48:21.210 -> Unable to reallocate memory for chain APP
12:48:41.210 -> index: 83, proof_number: 56293181147945369 60, previous hash: E105B000F1C, data: 007A219A1A22012420, timestamp: 40202
12:48:41.351 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 694
12:48:41.398 -> Appending block
12:48:41.398 -> Realocated chain nillnes: 0
12:48:41.445 -> Unable to reallocate memory for chain APP
12:48:41.492 -> index: 84, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A2161636F6C616552, timestamp: 40496
12:48:41.634 -> 30.00
12:48:41.634 -> Appending block
12:48:41.634 -> Realocated chain nillnes: 0
12:48:41.681 -> Unable to reallocate memory for chain APP
12:48:41.728 -> 0
12:48:41.728 -> index: 85, proof_number: 227634187520, previous hash: A300421220A21, data: [], timestamp: 68312480446979
12:48:41.823 -> 1

بدأ عمل النظام في التوقيت 12:48:01.314 **وتم تشكيل آخر عينة في التوقيت** 12:48:41.728 أي استمر التوليد لمدة 40 ثانية تقريباً، تم حساب البلوك الأول بشكل كامل في التوقيت 12:48:20.361 أي بعد 19 ثانية تقريباً، وكما هو ملاحظ لم يتم إضافة البلوك المولد إلى السلسلة مع بدء توليد البلوك بشكل كامل، يمكن ملاحظة أن الرسائل في بعض الأحيان لم تكن كاملة، مع آخر بلوك تم توليده، البلوك ذي الفهرسة 85 هو آخر بلوك تم توليده بعدها لم يقم النظام بتوليد أي بلوك.

3- السيناريو الثالث

تم تكرار السيناريو الثاني ولكن تم إعادة تشغيل المتحكم بشكل يدوي بعد توقفه عن التوليد، أظهرت النتائج ثبات proof_number عند الصفر وكذلك previous hash، برغم أن تشفير المعطيات تم لكل حساس إلا أنه بقي ثابتاً لكل حساس مع ثبات proof_number و previous hash، مع استمرار تغير التوقيت، استمر الوضع هكذا حتى العينة رقم 72 بدأ توليد الـ proof_number وبدأت previous hash بالتغير إلا أنه لم يتم إضافة أي عينة للسلسلة، استمر النظام بتوليد البلوكات وحسابها بدون إضافتها إلى السلسلة حتى العينة 85 كما هو واضح:

12:48:20.691 -> index: 72, proof_number: 296693317150703616, previous hash: E1011000F1C, data: 007A212064657461636F, timestamp: 19673
12:48:20.833 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
12:48:20.880 -> Appending block
12:48:20.880 -> Realocated chain nillnes: 0
12:48:20.927 -> Unable to reallocate memory for chain APP
12:48:21.022 -> index: 73, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A2100800000, timestamp: 20018
12:48:21.116 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
12:48:21.163 -> Appending block
12:48:21.163 -> Realocated chain nillnes: 0
12:48:21.210 -> Unable to reallocate memory for chain APP
12:48:21.305 -> index: 5447948221986577992, previous hash: E100B1E3BF1C, data: 007A219A1A22012420, timestamp: 20278
12:48:21.446 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 696
12:48:21.493 -> Appending block
12:48:21.493 -> Realocated chain nillnes: 0
12:48:21.540 -> Unable to reallocate memory for chain APP
12:48:41.492 -> index: 84, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A2161636F6C616552, timestamp: 40496
12:48:41.634 -> 30.00
12:48:41.634 -> Appending block
12:48:41.634 -> Realocated chain nillnes: 0
12:48:41.681 -> Unable to reallocate memory for chain APP
12:48:41.728 -> 0
12:48:41.728 -> index: 85, proof_number: 227634187520, previous hash: A300421220A21, data: [], timestamp: 68312480446979
12:48:41.823 -> 1

استمر العمل لمدة 40 ثانية ليتوقف المتحكم عن التوليد، وتم توليد البلوك الأول كاملاً بعد 19 ثانية، بعد إعادة التشغيل تكررت نفس الحالة مع تغير فهرس البلوك الأول الذي تم توليده الذي أصبح 71 بعد زمن حوالي 20 ثانية.

12:50:21.144 -> >>>>>>>>>>>>>>>>>>>>>>>> Begin Read Sensors >>>>>>>>>>>>>
12:50:39.862 -> index: 70, proof_number: 0, previous hash: E100000F1C, data: 007A2100800000, timestamp: 19062
12:50:39.956 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
12:50:40.003 -> Appending block
12:50:40.003 -> Realocated chain nillnes: 0
12:50:40.051 -> Unable to reallocate memory for chain APP
12:50:40.098 -> index: 71, proof_number: 607536388529483619, previous hash: E106720626CF1C, data: 007A219A00E1080, timestamp: 19281
12:50:40.239 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 683
12:50:40.286 -> Appending block
12:50:40.286 -> Realocated chain nillnes: 0
12:50:40.334 -> Unable to reallocate memory for chain APP
12:50:40.428 -> index: 72, proof_number: 296693317150703616, previous hash: E1011000F1C, data: 007A21636F6C6220676E, timestamp: 19595

12:50:40.570 -> tem= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 30.00
12:50:40.616 -> Appending block
12:50:40.616 -> Realocated chain nillnes: 0
12:50:40.664 -> Unable to reallocate memory for chain APP

العينة الأخيرة التي تم توليدها وتوقف المتحكم عن التوليد هي نفسها ذات الفهرس 85 مع ملاحظة أن الجمل لم تكن كاملة، استمر العمل لمدة 37 ثانية فقط.

12:50:58.070 -> index: 84, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A21636F6C6220676E, timestamp: 37280
12:50:58.212 -> 30.00
12:50:58.212 -> Appending block
12:50:58.258 -> Realocated chain nillnes: 0
12:50:58.258 -> Unable to reallocate memory for chain APP
12:50:58.306 -> 0
12:50:58.306 -> index: 85, proof_number: 227636687616, previous hash: A30080E100, data: [], timestamp: 68312480447491
12:50:58.447 -> 1

4- السيناريو الرابع

تم تكرار السيناريو الثاني ولكن تم إفراغ مخزن المنفذ التسلسلي بعد كل عملية إرسال، تم التفكير بذلك بسبب مشاكل الإرسال التي بدت واضحة مع البلوك الأخير، أظهرت النتائج ثبات proof_number عند الصفر وكذلك previous hash، برغم أن تشفير المعطيات تم لكل حساس إلا أنه بقي ثابتاً لكل حساس مع ثبات proof_number وprevious hash، مع استمرار تغير التوقيت، استمر الوضع هكذا حتى العينة رقم 71 بدأ توليد الـ proof_number وبدأت previous hash بالتغير إلا أنه لم يتم إضافة أي عينة للسلسلة، استمر النظام بتوليد البلوكات وحسابها بدون إضافتها إلى السلسلة حتى العينة 85 كما هو واضح:

12:55:50.761 -> index: 70, proof_number: 0, previous hash: E100000F1C, data: 007A2100800000, timestamp: 18838
12:55:50.855 -> Gaz= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 1
12:55:50.902 -> Appending block
12:55:50.902 -> Realocated chain nillnes: 0
12:55:50.949 -> Unable to reallocate memory for chain APP
12:55:50.997 ->
12:55:51.043 -> index: 71, proof_number: 576551347683094371, previous hash: E105265616CF1C, data: 007A219A10E1080, timestamp: 19057
12:55:51.138 -> light= >>>>>>>>>>>>>>> >>>>>>>>>>>>>>> 691
12:55:51.185 -> Appending block
12:55:51.232 -> Realocated chain nillnes: 0
12:55:51.232 -> Unable to reallocate memory for chain APP
12:56:10.192 -> index: 84, proof_number: 966368165888, previous hash: E100E108F1C, data: 007A217461636F6C6165, timestamp: 38255
12:56:10.287 -> 30.00
12:56:10.334 -> Appending block
12:56:10.334 -> Realocated chain nillnes: 0
12:56:10.381 -> Unable to reallocate memory for chain APP
12:56:10.428 -> 0
12:56:10.428 -> index: 85, proof_number: 227635042816, previous hash: A30080E101, data: [], timestamp: 68312480446723
12:56:10.523 -> 1

استمر العمل لمدة 36 ثانية ليتوقف المتحكم عن التوليد، وتم توليد البلوك الأول كاملاً بعد 19 ثانية، نتائج هذا السيناريو تظهر كيف أن المسبب الرئيسي لتوقف العمل في المتحكم هو الذاكرة SRAM وليس مخزن المنفذ التسلسلي.

ويبين الجدول (3) ملخصاً للنتائج الموضحة في السيناريوهات المبينة في الفقرات أعلاه.

الجدول (3) ملخص النتائج

السيناريو	المتحكم	الذاكرة	طول الهاش	رقم فهرس البلوك الأول	رقم فهرس البلوك الأخير	زمن توليد البلوك الأول	زمن توليد البلوك الأخير (التوقف)	عدد البلوكات المولدة	الزمن
الأول	Atmega2560	8	32	48	66	13:01:17.741	13:02:14.240	66	56.499
الثاني	Atmega2560	8	8	70	85	12:48:01.314	12:48:41.728	85	40.414
الثالث	Atmega2560	8	8	72	85	12:48:01.314	12:48:41.728	85	40.414
الرابع	Atmega2560	8	8	71	85	12:55:32.278	12:56:10.523	85	38.245

الاستنتاجات والتوصيات:

تم في هذا البحث تصميم خوارزمية **BlockChain** نعمل على المتحكمات الصغرية تقوم بتشفير وفك تشفير البيانات المتبادلة بين العقد، كما تم اختبار هذه الخوارزمية على نظام منزل ذكي العقدة فيه هي المتحكم **ATmega2560** ATmega2560 (وهو معالج 8 بت بذاكرة 8 KB SRAM) وفق عدة سيناريوهات تم تلخيص نتائجها في الجدول 3، وكانت الاستنتاجات كما يلي:

1. اختيار hash بطول أقل من 16 بت يخفف الزمن ولكنه يقلل عدد الـ hash المتولدة وبالتالي أصبح التعامل مع الانترنت بدون جدوى. (هناك محدودية بالعمل بسبب عدد الكتل التي يمكن توليدها).
2. البيانات هنا عبارة عن قيم قراءات الحساسات أو إشارات التشغيل وبالتالي لا حاجة لتطبيقات ضخمة أو فيزيائية.
3. يتم تحرير الـ hash بعد الارسال والاستقبال وبالتالي يمكن إعادة استخدامه في عمليات إرسال أخرى.
4. بالنسبة لمفتاح التشفير، يمكن استخدام الـ hash ذاته ولكن باستخدام مفتاح تشفير عشوائي آخر يزيد من أمن الارسال والاستقبال.
5. فشل جميع السيناريوهات في تحقيق شبكة تتضمن كافة المعلومات.
6. ضعف إمكانيات الذاكرة في معظم المتحكمات مما لا يسمح بتطبيق شبكة Blockchain.
7. يعد استخدام Proof_number لمقاطعة المهاجمين أمراً جيداً إلا أنه يستغرق زمناً طويلاً لتوليده.

العمل المستقبلي

يمكن تطوير الشبكة، وتحسين عملها، من خلال استخدام حاسب في شبكة Blockchain في توليد الكتل وارسالها عبر المتحكمات، كما يمكن استخدام ESP وصيغة JSON وتوسيع ذواكر المتحكمات لنتمكن من توليد الكتل وارسالها عبر الشبكة.

References:

1. Farahani, B.; Firouzi, F.; Luecking, M. The convergence of IoT and distributed ledger technologies (DLT): Opportunities, challenges, and solutions. J. Netw. Comput. Appl. **2021**, 177, 102936.
2. Qiao, H.; Novikov, B.; Blech, J.O. Concept Drift Analysis by Dynamic Residual Projection for effectively Detecting BotnetnCyber-attacks in IoT scenarios. IEEE Trans. Ind. Inform. **2021**, 18, 3692–3701.
3. Ashraf, J.; Keshk, M.; Moustafa, N.; Abdel-Basset, M.; Khurshid, H.; Bakhshi, A.D.; Mostafa, R.R. IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities. Sustain. Cities Soc. **2021**, 72, 103041.
4. Cviti´c, I.; Perakovi´c, D.; Gupta, B.; Choo, K.K.R. Boosting-based DDoS detection in internet of things systems. IEEE Internet Things J. **2021**, 9, 2109–2123.
5. Popoola, S.I.; Ande, R.; Adebisi, B.; Gui, G.; Hammoudeh, M.; Jogunola, O. Federated deep learning for zero-day botnet attack detection in IoT edge devices. IEEE Internet Things J. 2021, 9, 3930–3944.
6. Hussain, F.; Abbas, S.G.; Pires, I.M.; Tanveer, S.; Fayyaz, U.U.; Garcia, N.M.; Ghalib, A.S.; Shahzad, F. A Two-Fold Machine Learning Approach to Prevent and Detect IoT Botnet Attacks. IEEE Access **2021**, 9, 163412–163430
7. Vinayakumar, R.; Alazab, M.; Srinivasan, S.; Pham, Q.V.; Padannayil, S.K.; Simran, K. A visualized botnet detection system based deep learning for the Internet of Things networks of smart cities. IEEE Trans. Ind. Appl. 2020, 56, 4436–4456.
8. Hayat, R.F.; Aurangzeb, S.; Aleem, M.; Srivastava, G.; Lin, J.C.W. ML-DDoS: A Blockchain-Based Multilevel DDoS Mitigation Mechanism for IoT Environments. IEEE Trans. Eng. Manag. **2022**, 1–14.
9. Lekssays, A.; Landa, L.; Carminati, B.; Ferrari, E. PAutoBotCatcher: A Blockchain-based privacy-preserving botnet detector for Internet of Things. Comput. Netw. **2021**, 200, 108512.
10. Sun, S.; Du, R.; Chen, S.; Li,W. Blockchain-based IoT access control system: Towards security, lightweight, and cross-domain. IEEE Access **2021**, 9, 36868–36878.
11. Xu, C.; Qu, Y.; Luan, T.H.; Eklund, P.W.; Xiang, Y.; Gao, L. A Light-weight and Attack-Proof Bidirectional Blockchain Paradigm for Internet of Things. IEEE Internet Things J. **2021**, 9, 4371–4384.
12. Mikail Mohammed Salim , Alowonou Kowovi Comivi, Tojimurotov Nurbek, Heejae Park and Jong Hyuk Park, A Blockchain-Enabled Secure Digital Twin Framework for Early Botnet Detection in IIoT Environment, Department of Computer Science and Engineering, Seoul National University of Science and Technology (SeoulTech), Seoul 01811, Korea, Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland.