Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202295752
Vol. 9, No. 4, Agustus 2022, hlm. 675-682 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

SISTEM BERBASIS PRIVATE BLOCKCHAIN SEBAGAI PENYEDIA LAYANAN AUTENTIKASI PUBLISHER-BROKER-SUBSCRIBER PADA PROTOKOL MESSAGE QUEUE TELEMETRY TRANSPORT

Muhammad Naufal Dzakie[1], Adhitya Bhawiyuga*[2], Achmad Basuki[3]

[1,2,3]Universitas Brawijaya, Malang
Email: [1]duscae047@gmail.com, [2]bhawiyuga@ub.ac.id, [3]abazh@ub.ac.id
*Penulis Korespondensi

(Naskah masuk: 29 Oktober 2021, diterima untuk diterbitkan: 12 Agustus 2022)

Abstrak

Protokol MQTT pada umumnya menggunakan username dan password untuk memvalidasi klien yang terhubung ke broker. Salah satu cara yang biasa dilakukan untuk melakukan hal ini adalah dengan membuat dedicated server yang berfungsi untuk memvalidasi klien yang terhubung pada broker. Akan tetapi hal ini membuat proses validasi klien bergantung pada entitas yang umumnya dibuat terpusat (centralized). Sistem yang terpusat rentan mengalami kegagalan yang dapat menyebabkan sistem kehilangan data dan ketersediaan yang rendah ketika ingin digunakan. Oleh karena itu, peneliti mengusulkan penggunaan Ethereum blockchain sebagai pengganti dari authentication server. Blockchain merupakan teknologi penyimpanan data terdistribusi secara peer to peer yang dapat mencegah perubahan data tanpa izin. Selain itu platform Ethereum blockchain mempunyai teknologi smart contract, dengan teknologi ini pengguna dapat mengunggah program kecil pada blockchain. Pada penelitian ini, smart contract akan dijadikan pengganti dari authentication server yang biasanya digunakan pada broker MQTT. Penulis berharap terciptanya authentication server yang terdistribusi guna membantu broker MQTT dalam memvalidasi klien setelah menerapkan teknologi blockchain dan smart contract pada MQTT authentication server. Hasil dari pengujian fungsional yang didapat bahwa implementasi MQTT authentication server pada platform blockchain sudah dapat berjalan sesuai dengan fungsinya dalam melakukan authentication dan authorization pada klien. Hasil dari pengujian non fungsional menunjukkan bahwa distribusi data sudah dapat dijaga konsistensi nya pada tiap–tiap node. Berdasarkan hasil penelitian tersebut dapat disimpulkan bahwa sistem yang dibuat dapat dijadikan sebagai solusi permasalahan MQTT authentication server yang terpusat.

Kata kunci: *MQTT, authentication, authorization, blockchain, distributed, ethereum, smart contract.*

IMPLEMENTATION OF MQTT AUTH SERVER ON THE ETHEREUM BLOCKCHAIN PLATFORM

Abstract

Normally MQTT protocol uses username and password for klien validation with broker. One of many ways to this is to have a dedicated server that functions wholly on handling klien validation with broker. However, there were draw backs to this, as centralized server has a higher chance of failure, which can cause data loss. Therefore, in this study, we will propose a solution using Ethereum blockchain as a substitute for the authentication server. Blockchain is a peer-to-peer data storage technology that is distributed and immutable. With Ethereum blockchain user can upload a smart contract to the blockchain that acts as a mini program. Because of this the writer propose to make a smart contract that functions as an authentication server. If implemented correctly, the writer hopes to create a distributed authentication server that helps MQTT broker to validate kliens. The result of functional testing shows that the authentication server is running by its function to authenticate and authorize kliens that connects to the broker. While the result of non-functional testing shows that the system distributed function can maintain data consistency. Therefore, based on these results, this system can be a solution for a centralized MQTT authentication server.

Keywords: *MQTT, authentication, authorization, blockchain, distributed, ethereum, smart contract.*

1. PENDAHULUAN

Protokol MQTT merupakan protokol komunikasi yang ideal dikarenakan pemakaian sumber daya yang efisien ketika mengirimkan pesan (Amron, Yahya, & Mohammad, 2018). Komunikasi tersebut terdiri dari tiga bagian yang saling berhubungan yaitu subscriber, publisher, dan broker.

Di antara tiga bagian tersebut broker merupakan perantara komunikasi antara publisher dan subscriber. Komunikasi di antara ketiga komponen tersebut termasuk komunikasi asynchronous. Protokol MQTT tidak mempunyai sistem keamanan tersendiri. Hal ini dikarenakan protokol MQTT dikembangkan untuk berkerja pada lingkungan yang sudah dipastikan aman sebelumnya (Latvina, 2017). Menurut Andy (S. Andy & Hanindhito, 2017), terbatasnya sumber daya dan banyaknya jumlah perangkat yang terkoneksi kepada jaringan merupakan sebab–sebab jaringan MQTT mudah diserang oleh penjahat siber.

Untuk menambahkan keamanan pada protokol MQTT, mekanisme username dan password pada umumnya akan ditambahkan pada klien MQTT. Seperti yang diterapkan pada riset (Fatwa Fauzi, 2021), dimana mekanisme OTP (One Time Password) digunakan. Penambahan keamanan biasanya didampingi dengan penambahan mekanisme untuk melakukan validasi terhadap username dan password tersebut Salah satu bentuk dari mekanisme tersebut adalah authentication server.

Pada umumnya authentication server yang digunakan pada broker MQTT hanya terletak pada satu entitas saja. Authentication server biasa dijumpai di dalam broker sendiri atau pada dedicated server yang digunakan menjadi authentication server. Penelitian terdahulu (Wildan, Bhawiyuga, & Basuki, 2018) menggunakan infrastruktur cloud sebagai authentication server untuk memvalidasi klien yang terhubung dengan broker. Pada penelitian tersebut proses authentication hanya melibatkan satu entitas yaitu cloud yang dapat memungkinkan terjadinya kematian sistem dan kehilangan data ketika cloud yang digunakan mengalami kegagalan.

Oleh karena itu teknologi yang terdistribusi, dapat memastikan skalabilitas, privasi, dan keandalan merupakan hal penting yang harus diperhatikan untuk mengurangi masalah ketika membuat suatu authentication server untuk protokol MQTT. Pada waktu belakangan ini teknologi blockchain sudah mulai matang dan dilihat sebagai solusi untuk menyelesaikan masalah diatas. Blockchain merupakan sistem yang Secure By Design, hal ini berarti masalah – masalah keamanan dapat dimitigasi karena desain Blockchain yang tidak dapat diubah, transparan, dapat dipertanggungjawabkan, dan sudah mempunyai sistem keamanan tersendiri. Blockchain juga merupakan sistem yang terdesentralisasi. Blockchain merupakan struktur data terdistribusi yang direplikasi dan dibagikan kepada anggota–anggota blokchain (Devetsikiotis & Christidis, 2016). Selain itu, pengguna blokchain juga dapat mencatat, memeriksa, dan mengaudit transaksi yang dilakukan pada sistem blockchain (Ana, Martin, Chen, Soler, & Díaz, 2018).

Akan dirancang suatu cara untuk membuat authentication server untuk broker MQTT dengan menggunakan platform blockchain Ethereum. Cara ini akan menggunakan blockchain Ethereum sebagai sarana autentikasi dan otorisasi identitas perangkat yang akan terhubung dengan broker MQTT. Hal ini dilakukan dengan penggunaan fitur blockchain yaitu smart contract. Smart contract merupakan salah satu inovasi yang ada pada blockchain Ethereum. Dengan menggunakan smart contract, pengguna dapat melakukan komputasi pada blockchain Ethereum. Pada platfrom Ada beberapa bahasa pemrograman tingkat tinggi yang dapat digunakan pada smart contract Ethereum. Di antara bahasa pemrograman yang dapat digunakan pada platform Ethereum adalah Solidity, Vyper, dan LLL. Hasil dari pengembangan smart contract setelah dilakukan compile adalah bytecode. Karena bytecode merupakan opcode, node Ethereum dapat mengikuti opcode sebagai instruksi untuk melakukan sesuatu (Hu, Cho, & Kim, 2017). Smart contract akan dirancang untuk melakukan proses autentikasi dan otorisasi pada klien dan server MQTT.

Smart contract akan dideploy kepada salah satu node pada private blockchain yang sudah dibuat. Setelah dilakukan deploy, smart contract akan digandakan pada masing–masing node dan dapat diakses pada setiap node yang ada pada private blockchain dikarenakan sifat blockchain yang terdistribusi. Hal ini merupakan upaya untuk meminimalisir kegagalan sistem ketika authentication server mengalami fail.

Broker akan melakukan authentication dengan menggunakan alamat wallet Ethereum. Jika alamat broker sudah terdaftar pada smart contract, broker akan mendapatkan token unik dari smart contract untuk dijadikan parameter otorisasi ketika berinteraksi dengan smart contract. Broker juga akan berinteraksi dengan smart contract selang beberapa waktu yang dapat ditentukan untuk melakukan refresh pada token yang dimiliki.

Pada klien, proses autentikasi berupa pengiriman id, username, dan password oleh broker kepada smart contract. Smart contract menentukan apakah identitas klien terdaftar pada smart contract tersebut dengan membandingkan data klien dengan data smart contract. Begitu juga dengan proses otorisasi, broker akan mengirimkan id dan topik yang digunakan oleh klien pada smart contract untuk dibandingkan.

2. KAJIAN PUSTAKA

Penelitian ini merupakan penelitian yang dikembangkan dari penelitian–penelitian sebelumnya. Pertama ada penelitian survei dari Alfonso Paranello pada tahun 2018 yang berjudul "Blockchain and IoT Integration: A Systematic Survey". Pada penelitian ini dibahas berbagai macam aspek tentang integrasi antar IoT dan teknologi blockchain. Menurut Alfonso teknologi IoT yang data nya bersifat sentral dan sering menghasilkan masalah pada bagian privasi nya dapat diintegrasikan dengan teknologi blockchain. Blockchain merupakan

teknologi yang teknologi yang terdistribusi, selain itu blockchain juga dapat dipastikan scalability, privasi, dan keandalan nya. Menurut nya aspek–aspek blockchain tersebut dapat menutupi kekurangan IoT (Panarello, Tapas, Merlino, Longo, & Puliafito, 2018).

Selanjutnya ada penelitian yang dilakukan oleh Moh. Wildan Habibi yang berjudul "Rancang Bangun IOT Cloud Platform Berbasis Protokol Komunikasi MQTT" pada tahun 2018. Penelitian ini merancang suatu manajemen perangkat IoT berbasis cloud. Cloud platform akan digunakan menjadi alternatif daripada direct platform yang mampu mendukung komunikasi berbagai perangkat IoT dan menjamin integritas dan validasi perangkat maupun data (Wildan, Bhawiyuga, & Basuki, 2018). Wildan menyebutkan bahwa alasan penggunaan protokol MQTT sebagai metode berkomunikasi dikarenakan protokol MQTT dapat mengirimkan pesan dengan bandwidth dan resource yang kecil dibandingkan protokol yang lainnya.

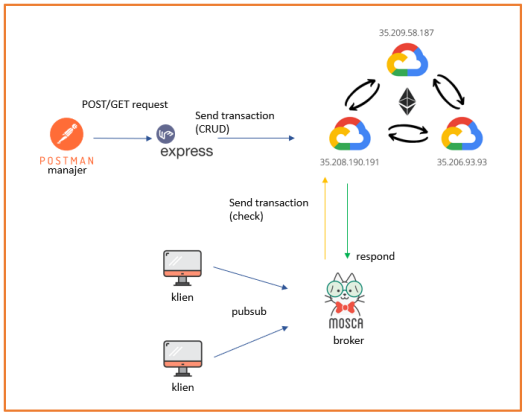
Setelah itu ada penelitian oleh Joni Fat dan rekan yang berjudul "Sekuritisasi Data Sensor Pada Aplikasi Internet of Things (IoT) Dengan Menggunakan Blockchain Ethereum Di Jaringan Testnet" pada tahun 2019. Penelitian ini menunjukkan mulai adanya upaya untuk mengaplikasikan teknologi blockchain pada lingkungan IoT. Pada akhir penelitian ini dilakukan pengiriman data sensor yang sudah dihubungkan dengan perangkat ESP32. Perangkat ESP32 akan melakukan koneksi ke jaringan percobaan Ropstein untuk ditransmisikan data nya (Fat, Candra, & William, 2019). Setelah mentransmisikan data ke jaringan Ropstein, klien akan menulis status pengiriman dan harga pada serializer ESP32.

3. DESAIN SISTEM

Tidak seperti model client-server pada umumnya, platfrom Ethereum blokchain menggunakan model terdesentralisasi (Wood, 2021). Pada bagian ini akan dijelaskan desain dari sistem yang dibuat dengan model terdesentralisasi. Gambar 1 mengilustrasikan desain umum yang sudah dibuat.

Pada Gambar 1 juga dapat ditemukan komponen–komponen yang digunakan pada penelitian. Komponen yang dimaksud adalah klien pubsub, broker, smart contract, private blockchain, dan web service.

Klien pubsub merupakan klien MQTT. Tiap klien pubsub memiliki kredensial yang terdiri dari id, username, dan password serta hanya memiliki kuasa pada topik–topik pesan tertentu. Klien akan mengirimkan data–data tersebut sebagai payload ketika ingin melakukan pubsub melewati broker. Broker merupakan entitas yang menerima request publish dan subscribe dari klien. Broker akan melakukan pengecekan payload dengan mengirimkan transaksi yang berisikan payload tersebut pada smart contract.



Gambar 1 Desain Umum Sistem

Smart contract merupakan mini program yang menyimpan set instruksi dan data yang dapat dijalankan, Pada penelitian ini smart contract digunakan untuk mengautentikasi dan mengotorisasi klien yang terhubung pada broker berdasarkan kredensial dan topik. Selain itu smart contract akan dirancang dengan memikirkan aspek–aspek keamanan tertentu dikarenakan smart contract bersifat public. Smart contract terletak di dalam jaringan private blockchain. Private blockchain terdiri dari tiga node yang saling terhubung. Di antara ketiga node tersebut, dua di antaranya merupakan node miner. Smart contract mempunyai fungsi utama untuk melakukan autentikasi dan otorisasi klien. Hal ini dilakukan dengan membandingkan payload dengan data yang disimpan pada blockchain.

Komponen terakhir adalah web service. Komponen ini bertindak sebagai interface yang dapat digunakan untuk berinteraksi dengan smart contract. Web service digunakan oleh satu–satunya stakeholder pada sistem yaitu manajer. Manajer dapat menggunakan web service untuk menambahkan dan merubah data klien dan broker pada smart contract.

Adapun aspek–aspek keamanan informasi yang akan diimplementasikan pada penelitian ini adalah authenticity, confidentiality, integrity, dan auditability. Aspek authenticity diterapkan pada proses autentikasi klien.

Aspek confidentiality saat smart contract menjaga data agar tidak dapat dilihat oleh pihak yang tidak mempunyai akses. Hal ini dibutuhkan karena smart contract dapat diakses oleh semua orang yang mempunyai akses pada private blockchain. Oleh karena itu perlu dirancang smart contract yang dapat membatasi akses fungsi maupun data pada smart contract tersebut. Aspek integrity pada pengujian penjaminan integritas smart contract. Hal ini dilakukan untuk memastikan tidak ada unsur perubahan pada smart contract yang digunakan. Terakhir aspek auditability pada pengujian audit transaksi smart contract. Karena setiap transaksi yang dilakukan oleh smart contract memiliki harga tertentu. Oleh karena itu smart contract harus dapat melakukan pencatatan transaksi agar dapat diaudit.

Implementasi desain sistem yang serupa juga sudah dilakukan oleh Buccafurri (Buccafurri, De Angelis, & Nardone, 2020). Bucaffuri menggunakan smart contract pada platform Ethereum untuk melakukan verifikasi password yang digunakan oleh perangkat MQTT. Selain Buccafuri, Ramachandran (Ramachandran, Wright, & Krishnamachari, 2018) juga menggunakan blockchain untuk memverifikasi data perangkat MQTT. Pada penelitiannya, Ramachandran membuat jaringan blockchain Trinity untuk menciptakan broker MQTT yang terdistribusi. Akan tetapi dari kedua implementasi diatas, implementasi yang dibuat oleh penulis lebih serupa dengan penelitian Gong (Gong, M. Alghazzawi, & Cheng, 2021). Pada penelitian yang dilakukan oleh Gong, diciptakan mekanisme registrasi perangkat MQTT pada jaringan blockchain.

Data hasil dari registrasi selanjutnya akan digunakan untuk mengautentikasi client yang ingin terhubung dengan broker MQTT. Manajemen data yang dilakukan pada blockchain seperti ini memiliki beberapa kelebihan. Menurut (David, 2016) di antara kelebihannya adalah keamanan dan privasi. Pada sisi keamanan, blockchain akan mengenkripsi data untuk melindungi data tersebut dari pengguna yang jahat. Sedangkan pada sisi privasi, blockchain dapat menerapkan sistem otoritas yang mengakibatkan data tidak bisa sembarang diakses.

4. IMPLEMENTASI SISTEM

Implementasi sistem terdiri dari implementasi komponen–komponen yang diperlukan sesuai rancangan yang sudah dibuat sebelumnya. Di antara komponen–komponen yang akan diimplementasikan adalah komponen private blockchain, smart contract, MQTT, dan web service.

Private blockchain dibangun dengan menggunakan tiga instance virtual server milik GCP (Google Cloud Platform). Instance yang digunakan mempunyai spesifikasi sebagai berikut:

- Tipe Mesin: n1 standard (1v CPU).
- RAM: 3.75 GB
- Sistem Operasi: Ubuntu 20.04 LTS

Tiap - tiap instance akan pasangkan blockchain menggunakan aplikasi geth (Go Ethereum). Akun wallet dan fake ether ditambahkan saat membuat file genesis menggunakan library puppeth. Penambahan akun miner juga ditambahkan menggunakan library ini. Node pertama dan ketiga akan dipasangkan dengan akun miner sedangkan node kedua merupakan node biasa. Pada pengujian node pertama lah yang akan digunakan untuk melakukan deploy dan berinteraksi dengan smart contract, oleh karena itu saat inisialisasi atau dimulainya program blockchain, akun miner pertama akan dibuka kunci nya secara langsung melalui parameter command. Karena node ketiga merupakan miner, hal yang sama akan dilakukan untuk membuka kunci akun miner yang kedua.

Smart contract dibangun menggunakan bahasa pemrograman Solidity. Implementasi dari smart contract meliputi implementasi struktur data beserta fungsi–fungsinya. Struktur data pada smart contract terdiri dari struktur data klien dan broker. Sedangkan fungsi dari smart contract terdiri dari mendapatkan, penambahan, pengurangan data broker dan klien seperti kredensial dan topik, selanjutnya ada validasi klien dengan autentikasi dan otorisasi, setelah itu ada fungsi memperbaharui token milik broker dan mendapatkan audit transaksi yang merubah state. Sebelum smart contract dapat dideploy kepada jaringan private blockchain, smart contract harus dilakukan compile terlebih dahulu. Kode untuk melakukan compile smart contract ditulis dengan menggunakan bahasa pemrograman Javascript dan library Web3. Hasil dari compile tersebut adalah file JSON yang berisikan ABI (Application Binary Interface) dan bytecode dari smart contract. Selanjutnya ABI dan bytecode dapat digunakan untuk melakukan deploy terhadap smart contract pada jaringan private blockchain. Kode untuk melakukan deploy smart contract ditulis dengan bahasa pemrograman Javascript dan library Web3.

Implementasi komponen MQTT terbagi menjadi tiga bagian. Di antaranya adalah implementasi broker, publisher, dan subscriber. Broker akan dibangun dengan bahasa pemrograman JavaScript dan library Aedes JS. Selain fungsi broker pada umum nya, broker akan ditambahkan fitur untuk melakukan refresh token kepada smart contract setiap 90 detik sekali. Publisher dan subscriber juga dibangun dengan bahasa pemrograman Javascript dan library Aedes JS.

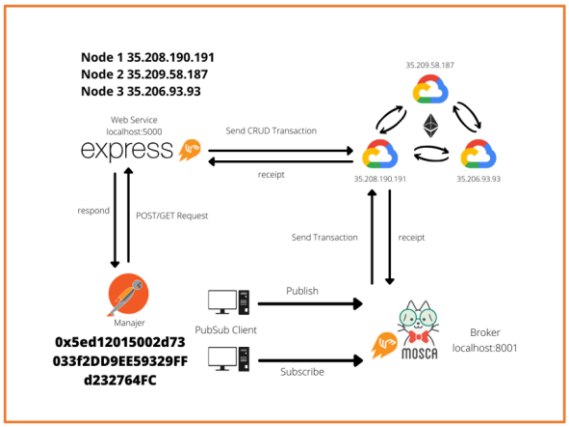
Web service merupakan aplikasi sederhana yang dibuat dengan library Express JS dan Web3. Aplikasi ini hanya bisa digunakan oleh pengguna yang mempunyai alamat wallet manajer. Manajer akan mengirimkan request POST dan GET pada web service layaknya sebuah API. Hal ini agar memudahkan aplikasi Postman dan Jmeter untuk mengakses web service ketika ingin dilakukan pengujian.

5. EVALUASI

5.1. Lingkungan Pengujian

Pada penelitian ini pengujian akan dilakukan pada jaringan private blockchain yang terhubung dengan broker dan web service. Gambar 2 merupakan ilustrasi dari lingkungan pengujian.

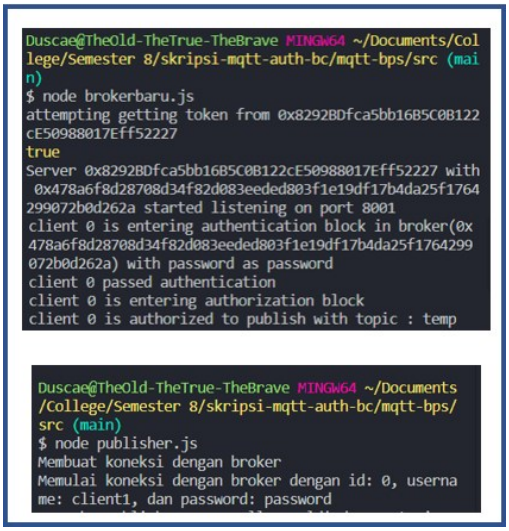
Gambar 2 mengilustrasikan manajer sebagai stakeholder yang menggunakan alamat wallet yang sudah ditentukan sebelumnya. Manajer akan menggunakan web service untuk melakukan pengujian sedangkan broker akan berinteraksi secara langsung dengan private blockchain untuk melakukan pengujian.



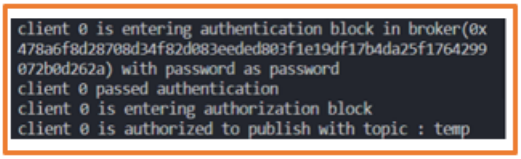
Gambar 2 Lingkungan Pengujian

5.2. Fungsional

Pengujian fungsional sistem dibagi menjadi dua bagian. Bagian pertama merupakan pengujian CRUD data klien dan broker MQTT pada smart contract. Bagian kedua merupakan pengujian autentikasi dan otorisasi klien dan broker MQTT pada smart contract. Smart contract yang dibuat berhasil memenuhi kedua pengujian tersebut. Smart contract berhasil melakukan CRUD data dan aktivitas autentikasi dan otorisasi klien dan broker MQTT yang dapat dilihat pada Gambar 3 dan Gambar 4.



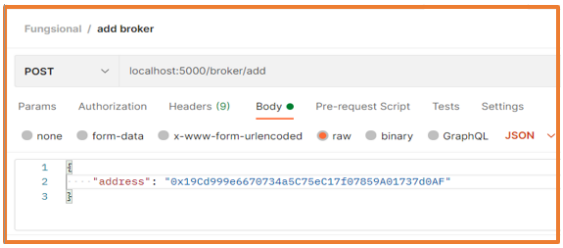
Gambar 3 Autentikasi Klien



Gambar 4 Otorisasi Klien

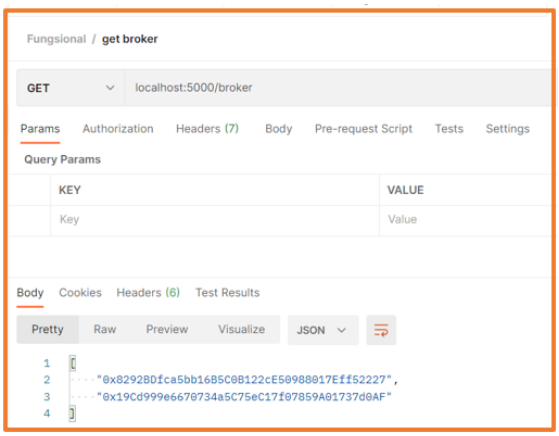
5.3. Konsistensi Data

Hasil dari pengujian konsistensi data menunjukkan bahwa jaringan private blockchain dapat menjaga konsistensi data. Sesuai dengan rancangan pengujian, node kedua akan dimatikan sementara dilanjutkan dengan pengiriman request POST kepada web service untuk mengirimkan transaksi pada node pertama. Setelah itu dilakukan pengecekan data pada masing–masing node setelah node kedua dinyalakan kembali. Oleh karena itu jaringan private blockchain merupakan jaringan yang terdistribusi.



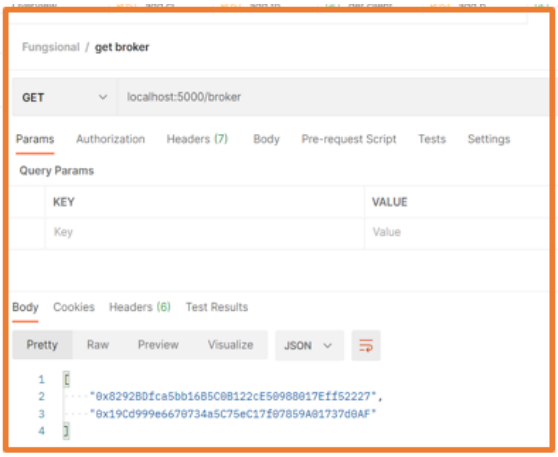
Gambar 5 Penambahan Broker Baru

Hasil dari penambahan broker baru melalui aplikasi POSTMAN dapat dilihat pada Gambar 5.



Gambar 6 Pengecekan Broker Pada Node Ketiga

Hasil dari pengecekan data baru broker pada node kedua menggunakan aplikasi Postman dapat dilihat pada Gambar 6.



Gambar 7 Pengecekan Broker Pada Node Kedua

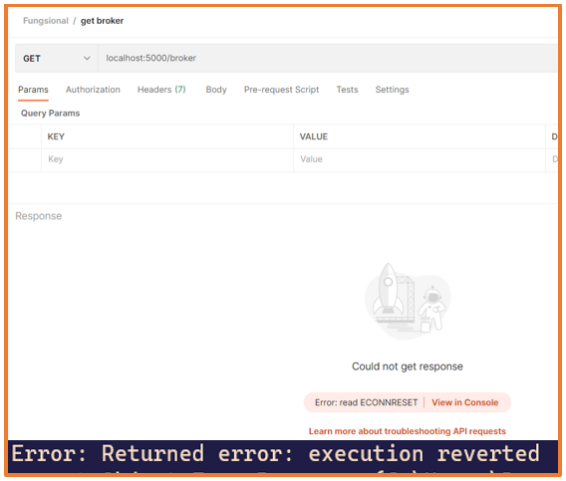
Hasil dari pengecekan data baru broker pada node kedua menggunakan aplikasi Postman dapat dilihat pada Gambar 7.

5.4. *Auditability*

Hasil dari pengujian auditability menunjukkan bahwa seluruh transaksi yang mengubah state pada smart contract dapat diperiksa atau diaudit. Setiap smart contract menjalankan fungsi yang mengubah sebuah state, fungsi pencatatan log akan dijalankan. Fungsi ini akan menerima parameter berupa timestamp, fungsi yang dijalankan, message, dan akun yang menjalankan fungsi tersebut. Dari parameter tersebut akan dibuat objek baru yang akan dimasukkan pada daftar log pada smart contract. Smart contract melakukan ini dengan memanfaatkan fitur event milik bahasa Solidity. Event akan disisipkan pada setiap fungsi yang ingin dicatat.

5.5. *Confidentiality*

Hasil dari pengujian confidentiality menunjukkan bahwa smart contract sudah bisa menjaga kerahasiaan data. Pengguna yang bukan manajer akan mengirimkan sebuah transaksi kepada smart contract yang berfungsi untuk mengambil data dari smart contract. Transaksi akan dikirimkan dengan web service melalui aplikasi Postman.

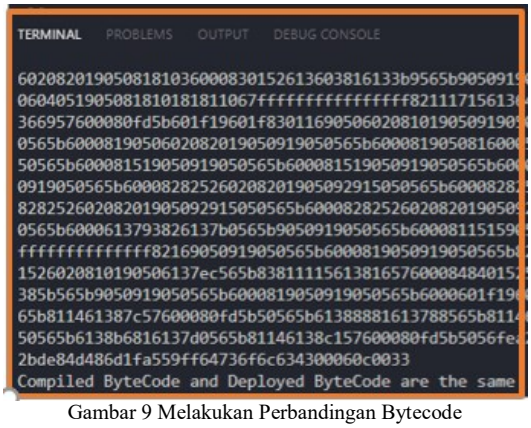


Gambar 8 Eksekusi Transaksi Gagal Karena Tidak Diberi Akses

Dapat dilihat pada Gambar 8 bahwa smart contract berhasil menolak dengan menggagalkan transaksi yang dikirim. Smart contract melakukan ini dengan memanfaatkan fitur modifier milik bahasa Solidity. Modifier akan diberikan pada setiap fungsi yang ingin dibatasi akses dengan parameter beruapa syarat agar fungsi tersebut dapat berjalan. Misalnya pada contoh kali ini parameter nya adalah fungsi baru yang akan mengembalikan true apabila pemanggil berupa manajer.

5.6. Integritas *Smart contract*

Hasil dari pengujian integritas smart contract menunjukkan bahwa sistem dapat memastikan integritas dari smart contract yang digunakan. Hal ini dibuktikan setelah menjalankan script untuk membandingkan bytecode smart contract yang sudah berada pada blockchain dengan bytecode yang didapatkan dari hasil compile mandiri. Integritas smart contract pada jaringan private blockchain sudah terjaga dikarenakan hasil dari perbandingan deployed bytecode dan bytecode compile mandiri merupakan sama.



Gambar 9 Melakukan Perbandingan Bytecode

Dapat dilihat pada Gambar 9 menunjukkan bahwa hasil perbandingan bytecode adalah sama.

5.7. Skalabilitas

Hasil dari pengujian skalabilitas menunjukkan bahwa jaringan *private blockchain* dapat bertahan dari beban yang diberikan. *Web service* berhasil menerima dan meneruskan semua request yang dikirimkan menggunakan aplikasi Jmeter selama 60 detik tanpa ada error. Rangkuman uji skalabilitas dideskripsikan pada Tabel 1.

Tabel 1 Hasil Uji Skalabilitas

No	Request	Waktu(detik)	Error
1	120	60	0%
2	180	60	0%
3	240	60	0%

6. KESIMPULAN

Penerapan MQTT authentication server pada platform Ethereum blockchain berhasil tercapai. Semua komponen sistem dapat diimplementasikan dan diuji sesuai dengan rancangannya.

Aspek keamanan yang direncanakan ada pada sistem berhasil diimplementasikan. Aspek authenticity berhasil diterapkan ketika klien dapat divalidasi oleh broker. Aspek confidentiality berhasil diterapkan ketika smart contract berhasil menolak akses dari alamat wallet yang bukan manajer atau broker. Aspek integrity berhasil diterapkan ketika integritas smart contract dapat dipastikan dengan melakukan perbandingan bytecode. Aspek auditability dapat diterapkan ketika smart contract menyimpan log transaksi yang merubah state. Selain itu sistem yang dibuat juga merupakan sistem yang terdistribusi berdasarkan pengujian konsistensi data.

Kinerja dari private blockchain dalam menangani beban berhasil diukur setelah melakukan pengujian skalabilitas. Dapat disimpulkan dari

pengujian skalabilitas, jaringan private blockchain dapat menerima 3 request transaksi per 1 detik. Hal ini terjadi ketika web service dapat menerima POST request dari 120 thread dan waktu ramp up 60 detik yang dikirimkan oleh aplikasi Jmeter.

DAFTAR PUSTAKA

AMRON, K., YAHYA, W., & MOHAMMAD, H. R. 2018. Implementasi Metode Failover pada Broker Protokol MQTT Dengan ActiveMQ. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*(10).

ANA, R., MARTIN, C., CHEN, J., SOLER, E., & DIAZ, M. 2018. On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems*.

BUCCAFURRI, F., DE ANGELIS, V., & NARDONE, R. 2020. Securing MQTT by Blockchain-Based OTP Authentication. *Sensors MDPI*.

DAVID, A. 2016. Managing Iot Data On Hyperledger Blockchain. *UNLV, University Libraries*.

DEVETSIKIOTIS, M., & CHRISTIDIS, K. 2016. Blockchains and Smart Contracts for the Internet of Things. *Special Section On The Plethora Of Research In Internet Of Things (IoT), 4*, 2292-2303.

FAT, J., CANDRA, H., & WILLIAM. (2019). Sekuritisasi Data Sensor Pada Aplikasi Internet of Things (IoT) Dengan Menggunakan Blockchain Ethereum Di Jaringan Testnet. *TESLA, 21*(1).

FATWA FAUZI, A. 2021. Otentikasi pada Internet-of-Things berbasis MQTT Menggunakan One-Time-Password pada Kasus IoT Home Gateway. *e-Proceeding of Engineering : Vol.8, No.1*, 809.

GONG, L., M. ALGHAZZAWI, D., & CHENG, L. 2021. BCoT Sentry: A Blockchain-Based Identity Authentication Framework for IoT Devices. *MDPI*.

HU, S., CHO, S., & KIM, S. 2017. Managing IoT Devices using Blockchain Platform . *International Conference on Advanced Communication Technology (ICACT)* .

LATVINA, N. 2017. Security in the Internet of Things: A Survey on Application Layers. *International Conference on Control Systems and Computer Science*.

PANARELLO, A., TAPAS, N., MERLINO, G., LONGO, F., & PULIAFITO, A. 2018. Blockchain and IoT Integration: A Systematic Survey. *Multidisciplinary Digital Publishing Institute*.

RAMACHANDRAN, G. S., WRIGHT, K.-L., & KRISHNAMACHARI, B. 2018. Trinity: A Distributed Publish/Subscribe Broker with Blockchain-based Immutability. *IEEE*.

S. ANDY, R., & HANINDHITO, B. 2017. Attack scenarios and security analysis of mqtt communication protocol in iot system. *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*.

WILDAN, H., BHAWIYUGA, A., & BASUKI, A. 2018. Rancang Bangun IOT Cloud Platform Berbasis Protokol Komunikasi MQTT. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*(2), 479-485.

WOOD, G. 2021, August 12. *Ethereum: A Secure Decentralised Generalised Transaction Ledge*. Retrieved from Gavin Wood: https://gavwood.com/paper.pdf

Halaman ini sengaja dikosongkan