Studi Literatur: Analisis Performa Algoritma SHA-3 (keccak) dalam Proses Hashing Data

1. **Pendahuluan**

   Keamanan informasi menjadi pilar utama dalam era digital, dengan fungsi hash kriptografis memainkan peran krusial dalam menjamin integritas data, otentikasi, dan tanda tangan digital. Algoritma Secure Hash Algorithm (SHA) yang digunakan secara luas, seperti SHA-1 dan SHA-2, mulai menunjukkan kerentanan terhadap serangan kriptanalisis. Untuk menjawab tantangan keamanan masa depan, National Institute of Standards and Technology (NIST) menyelenggarakan kompetisi yang menghasilkan algoritma Keccak sebagai pemenang, yang kemudian distandarisasi sebagai SHA-3 pada tahun 2015 (NIST, 2015)

   Perkembangan SHA-3, yang didasarkan pada konstruksi sponge dan permutasi, membawa perubahan mendasar dalam desain fungsi hash kriptografis dibandingkan dengan struktur Merkle–Damgård pada SHA-2. Namun, performa SHA-3 dalam proses hashing data masih menjadi tantangan utama**,** terutama di lingkungan dengan sumber daya terbatas seperti perangkat keras FPGA**,** mikrokontroler, atau sistem tertanam IoT/WSN (Jararweh et al., 2012; Kim et al., 2021). Analisis performa mencakup aspek seperti throughput, konsumsi daya, efisiensi area, dan waktu eksekusi**,** yang secara langsung mempengaruhi tingkat adopsi SHA-3 di aplikasi nyata.

   Dalam penyusunan studi literatur ini, penulis memperoleh berbagai referensi jurnal ilmiah yang relevan dengan topik evaluasi performa algoritma SHA-3 (Keccak). Proses pengumpulan literatur dilakukan melalui pencarian sistematis di basis data akademik, antara lain *Google Scholar, ResearchGate, IEEE Xplore,* dan *SpringerLink*, dengan kata kunci seperti *"SHA-3 hardware performance", "Keccak FPGA implementation",* dan *"hash algorithm efficiency"*.

   Berdasarkan hasil penelusuran terhadap jurnal-jurnal tersebut, penulis memperoleh gambaran bahwa kinerja algoritma SHA-3 (Keccak) telah banyak dievaluasi pada berbagai platform perangkat keras, seperti FPGA, AVR, dan ARM. Namun, sebagian besar penelitian terdahulu masih berfokus pada aspek kecepatan dan efisiensi konsumsi daya, tanpa mempertimbangkan optimalisasi integrasi antara keamanan dan performa pada perangkat keras modern seperti sistem IoT dan embedded systems. Hal ini menunjukkan adanya celah penelitian yang dapat dikembangkan lebih lanjut.

   Oleh karena itu, studi literatur ini berfokus pada **"Analisis Performa Implementasi Algoritma SHA-3 (Keccak) pada FPGA untuk Optimasi Throughput dan Efisiensi Area".** Tujuannya adalah untuk meninjau, membandingkan, dan mensintesis hasil-hasil penelitian terdahulu yang mengevaluasi dan mengoptimasi implementasi SHA-3 pada berbagai platform FPGA, serta mengidentifikasi pendekatan desain yang mampu meningkatkan kinerja hashing tanpa mengorbankan efisiensi sumber daya. Kajian ini

diharapkan dapat memberikan kontribusi terhadap pengembangan desain fungsi hash modern yang efisien dan aman untuk digunakan dalam sistem kriptografi masa depan.

2. Konsep dasar Kriptografi dan SHA-3

2.1 Konsep dasar kriptografi

Kriptografi adalah ilmu yang mempelajari teknik untuk mengamankan informasi dengan cara mengubah pesan asli (plaintext) menjadi bentuk yang tidak dapat dibaca (ciphertext) menggunakan algoritma dan kunci tertentu. Tujuan utama kriptografi meliputi: kerahasiaan (confidentiality), integritas (integrity), autentikasi (authentication), dan non-repudiation (Stallings William, 2017)

2.2 Fungsi Hash dan Proses Hashing

Fungsi hash adalah algoritma matematis satu arah yang mengonversi data input (pesan) dengan ukuran acak menjadi output tetap (hash atau digest), dinyatakan sebagai **h = H(M)**, di mana M adalah pesan (data input dengan ukuran acak) dan h adalah niai hash (output dengan ukuran tetap). Proses hashing melibatkan langkah-langkah seperti padding, pemrosesan blok, dan kompresi untuk memastikan integritas data. SHA-3, sebagai standar terbaru, menggunakan sponge construction yang terdiri dari fase absorb (menyerap input) dan squeeze (menghasilkan output), berbeda dari struktur Merkle-Damgård pada SHA-2 (NIST, 2015). Ini membuat SHA-3 lebih fleksibel dan tahan terhadap serangan seperti length extension.

2.3 Konsep Kunci Algoritma SHA-3 (Keccak)

Standar SHA-3 mendefinisikan keluarga fungsi hash dan *Extendable-Output Functions* (XOF) yang beroperasi pada data biner (Jurnal NIST/FIPS PUB 202).

- **Algoritma Dasar:** SHA-3 didasarkan pada algoritma Keccak, yang menggunakan Permutasi Keccak-p sebagai inti pemrosesannya. Permutasi ini bekerja melalui serangkaian *step mappings* $(\boldsymbol{\theta}, \boldsymbol{\rho}, \boldsymbol{\pi}, \boldsymbol{\iota}, \boldsymbol{\chi})$ pada *state array* berukuran 5 x 5 x w bit, dimana :
  - Angka 5 × 5 mewakili 25 “lane” (jalur) yang membentuk susunan dua dimensi (baris dan kolom),
  - Dan *w* menunjukkan lebar tiap lane dalam bit**,** yang menentukan ukuran total *state* sebesar **b = 5 × 5 × w bit** (Bertoni et al., n.d.).
- **Konstruksi:** Arsitektur SHA-3 menggunakan *Sponge Construction* (Konstruksi Spons), yang terdiri dari dua fase:
  1) *Absorb:* Data input “diserap” ke dalam *state* internal secara bertahap, diikuti dengan permutasi Keccak-p di setiap blok data.
  2) *Squeeze:* Setelah semua data diabsorpsi, keluaran hash “diperas” dari *state* internal hingga mencapai panjang output yang diinginkan.

Model ini dikontrol oleh dua parameter utama, yaitu **rate (r)** dan **capacity (c)**, di mana *rate* memengaruhi kecepatan pemrosesan data dan *capacity* menentukan tingkat keamanan terhadap serangan kolisi maupun pra-citra. Desain *sponge* ini terbukti lebih tahan terhadap serangan *length-extension* dan *collision* dibandingkan dengan struktur *Merkle–Damgård* yang digunakan pada SHA-2 (NIST, 2015).

- **Varian Standar:** Standar meresmikan fungsi hash tetap (SHA3-224, SHA3-256, SHA3-384, SHA3-512) dan fungsi output yang dapat diperpanjang (SHAKE128, SHAKE256), yang menawarkan fleksibilitas untuk berbagai aplikasi kriptografi modern.

2.4 Aspek Performa Algoritma SHA-3

SHA-3 didasarkan pada Keccak, dengan varian seperti SHA3-256, SHA3-512, dan SHAKE (Extendable Output Function). Performa diukur melalui metrik seperti:

- **Throughput:** Mengukur kecepatan pemrosesan data per satuan waktu. Nilai throughput yang tinggi menunjukkan kemampuan algoritma untuk menghasilkan hash dalam waktu singkat.
- **Area (CLB/Slices):** Menggambarkan jumlah sumber daya logika yang digunakan pada perangkat keras seperti FPGA. Area yang lebih kecil berarti desain lebih hemat ruang dan biaya.
- **Clock Frequency:** Menunjukkan kecepatan operasi sistem. Frekuensi yang tinggi umumnya meningkatkan throughput, tetapi bisa berdampak pada konsumsi daya.
- **Power Consumption:** Mengukur energi yang digunakan selama proses hashing. Ini menjadi faktor penting untuk aplikasi portabel atau perangkat IoT yang bergantung pada baterai.
- **Latency/Delay:** Waktu yang dibutuhkan untuk memproses satu blok data input hingga menghasilkan output hash (Jararweh et al., 2012; Kaps et al., 2011)

Implementasi SHA-3 dapat dilakukan di hardware (FPGA) untuk performa tinggi atau software (mikrokontroler) untuk efisiensi daya. Tantangan utama adalah trade-off antara keamanan (tinggi pada SHA-3) dan performa (lebih lambat dibanding SHA-2 di beberapa platform) (Chandran & Manuel, 2016)

3. Tinjauan penelitian terdahulu

Analisis performa SHA-3 (Keccak) dan kandidat finalisnya (*Blake, Groestl, JH, Skein*) telah dilakukan secara ekstensif pada berbagai platform, yang menunjukkan *trade-off* kritis antara kecepatan (*throughput* dan *frequency*) dan efisiensi sumber daya (*area* dan *power*).

Peneliti & Tahun	Metode/ Algoritma	Tujuan penelitian	Hasil & temuan	Kelemahan/ keterbatasan

(Jararweh et al., 2012)	Evaluasi hardware kandidat SHA-3 (Blake, Groestl, JH, Keccak, Skein) pada FPGA Virtex-5/6/7 menggunakan VHDL, dengan metrik clock frequency, area, throughput, dan daya.	Menentukan kandidat terbaik berdasarkan performa hardware untuk standar baru.	Keccak menunjukkan performa terbaik secara konsisten dalam hal Clock Frequency dan Throughput tertinggi di semua platform FPGA. JH dan Keccak memiliki frekuensi tertinggi, dan semua kandidat lebih baik dari SHA-2 dari sisi *throughput*. Blake dan Groestl memiliki hasil terburuk pada efisiensi daya dan luas area.	Tidak menguji platform non-FPGA serta fokus kandidat final saja.
(Kim et al., 2021)	Optimasi chaining pada SHA-3 (Keccak) di AVR 8-bit, menggabungkan proses θ, ρ, π untuk mengurangi akses memori.	Meningkatkan efisiensi SHA-3 di mikrokontroler berdaya rendah untuk IoT/WSN.	Peningkatan performa signifikan, hingga 26,1% dibandingkan implementasi terbaik sebelumnya, dengan mengurangi akses memori internal (dari 7 menjadi 5 kali per putaran). Metode ini efektif pada sistem tertanam (embedded systems) dan tidak mengorbankan keamanan algoritma.	Tidak membandingkan dengan hardware tapi fokus software embedded.
(Chandran &	Modifikasi SHA-3 pada FPGA Spartan-6 menggunakan	Menganalisis efisiensi area, delay, dan	MUX mengurangi area (slice registers dari 82k ke 4k), tapi delay	Trade-off area vs kecepatan tidak dioptimalkan,

Manuel, 2016)	Verilog, dengan step-by-step dan MUX-based architecture, ditambah error-tolerant scheme.	toleransi kesalahan.	lebih tinggi, step-by-step lebih cepat.	fokus FPGA saja.
(NIST, 2015)	Spesifikasi standar SHA-3 berdasarkan Keccak-p permutation dan sponge construction, dengan analisis keamanan dan fleksibilitas.	Menetapkan standar SHA-3 untuk keamanan informasi, melengkapi SHA-2	SHA-3 tahan collision hingga 256-bit, SHAKE fleksibel untuk output variabel, efisien di hardware/software .	Tidak fokus performa spesifik, lebih pada spesifikasi standar.
(Kaps et al., 2011)	Implementasi ringan kandidat SHA-3 (Blake, Groestl, JH, Keccak, Skein) pada FPGA (Spartan-3/6, Virtex-5, Cyclone-II) dengan batas area 400-600 slices.	Membandingkan performa lightweight SHA-3 candidates di FPGA Spartan-3/6 dan Altera Cyclone-II.	BLAKE-256 unggul dalam metrik Throughput per Area pada FPGA Xilinx. Grøstl unggul pada platform Altera. Skein adalah yang terburuk dalam kategori *lightweight* karena kompleksitas kontrol/datapath.	Tidak membahas konsumsi daya atau optimasi untuk pesan pendek hanya ber fokus hardware saja.

4. Analisis & Sintesis

Dari tinjauan penelitian tersebut, beberapa pola dan tren dapat diidentifikasi:

- **Tren Implementasi Hardware vs Software**: Penelitian hardware (FPGA) menekankan efisiensi area dan throughput (Jararweh et al., 2012), sementara software (mikrokontroler) fokus pada optimasi daya dan memori (Kim et al., 2021). SHA-3 standar (2015) memberikan dasar untuk keduanya, dengan sponge construction yang mendukung paralelisasi.
- **Perbandingan Kandidat dan Trade-off Performa**: Keccak (SHA-3) konsisten unggul dalam throughput dan efisiensi daya, tapi Blake/Groestl lebih baik di area terbatas (Kaps et al., 2011). Modifikasi seperti MUX (2013) mengurangi area tapi meningkatkan delay, menunjukkan trade-off antara kecepatan dan resource.

- **Celah Penelitian (Research Gap)**: Kurangnya integrasi hardware-software hibrida; tidak ada perbandingan dengan SHA-2 di platform embedded; fokus pada kandidat lama, belum eksplorasi SHA-3 di aplikasi modern seperti blockchain atau AI-driven hashing.

5. Arah dan Peluang Penelitian

Berdasarkan analisis, peluang penelitian masa depan meliputi:

1. **Integrasi Hardware-Software Hibrida**: Menggabungkan optimasi chaining (2014) dengan implementasi FPGA (2011) untuk sistem IoT yang efisien.
2. **Perbandingan dengan Algoritma Modern**: Evaluasi SHA-3 vs SHA-2 di platform quantum-resistant, dengan metrik daya dan throughput.
3. **Optimasi untuk Aplikasi Spesifik**: Penelitian hashing SHA-3 di big data atau edge computing, mengatasi celah performa untuk pesan besar.

6. Kesimpulan

SHA-3 merupakan algoritma hash yang kuat dan terbukti superior dalam hal keamanan dan performa *throughput* pada *hardware* kelas atas. Meskipun demikian, adopsi di lingkungan terbatas (sistem tertanam atau *low-cost* FPGA) memerlukan strategi implementasi spesifik. Implementasi *hardware* harus menyeimbangkan kebutuhan throughput dengan efisiensi area, di mana kandidat lain (seperti Blake) mungkin lebih unggul dalam rasio *throughput-per-area* yang ketat. Sementara itu, untuk implementasi *software* pada mikrokontroler, optimasi kode yang mengurangi akses memori adalah kunci untuk mencapai efisiensi yang kompetitif dengan standar lama.

Referensi

Bertoni, G., Daemen, J., Peeters, M., & Assche, G. Van. (n.d.). *Building power analysis resistant implementations of Keccak*. http://conferenze.dei.polimi.it/FDTC10/index.html

Chandran, N. R., & Manuel, E. M. (2016). Performance Analysis of Modified SHA-3. *Procedia Technology*, *24*, 904–910. https://doi.org/10.1016/j.protcy.2016.05.168

Jararweh, Y., Tawalbeh, L., Tawalbeh, H., & Moh'd, A. (2012). Hardware Performance Evaluation of SHA-3 Candidate Algorithms. *Journal of Information Security*, *03*(02), 69–76. https://doi.org/10.4236/jis.2012.32008

Kaps, J. P., Yalla, P., Surapathi, K. K., Habib, B., Vadlamudi, S., Gurung, S., & Pham, J. (2011). Lightweight implementations of SHA-3 candidates on FPGAs. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and*

Lecture Notes in Bioinformatics), *7107 LNCS*, 270–289. https://doi.org/10.1007/978-3-642-25578-6_20

Kim, Y. B., Youn, T. Y., & Seo, S. C. (2021). Chaining optimization methodology: A new sha-3 implementation on low-end microcontrollers. *Sustainability (Switzerland)*, *13*(8). https://doi.org/10.3390/su13084324

NIST. (2015). *SHA-3 STANDARD: PERMUTATION-BASED HASH AND EXTENDABLE OUTPUT FUNCTIONS*. https://doi.org/10.6028/NIST.FIPS.202

Stallings William. (2017). *Cryptography_and_Network_Security_Principles and Practice*.