PERANCANGAN IMPLEMENTASI SHA-1 *PASSWORD CRACKING* BERBASIS FPGA: STUDI KASUS INSTANSI XYZ

Siti Zaim[1], Nazori AZ[2], Hadi Syahrial[3]

Program Studi Magister Ilmu Komputer, Universitas Budi Luhur
[1]bealfathunnisa@gmail.com, [2]nazori@budiluhur.ac.id, [3]hadisyahrial@gmail.com

ABSTRAK

Password merupakan salah satu metode otentikasi yang digunakan pada banyak aplikasi. Password disimpan dalam bentuk digest yang dihasilkan dari perhitungan menggunakan fungsi hash, salah satunya SHA-1. Instansi XYZ bergerak dalam bidang pengamanan informasi yang membutuhkan kecepatan dalam melakukan komputasi kriptanalisis termasuk password cracking di dalamnya. FPGA merupakan salah satu alternatif teknologi komputasi dengan performansi tinggi. Pada penelitian ini akan dibuat rancangan implementasi SHA-1 password cracking dengan metode brute force attack.Pada Hasil akhir dari penelitian ini berupa rancangan implementasi SHA-1 password cracking berbasis FPGA bagi Instansi XYZ. Dengan menggunakan rancangan ini, Instansi XYZ dapat melakukan password cracking lebih cepat dibandingkan menggunakan personal computer sehingga informasi dapat lebih cepat disampaikan untuk pengambilan keputusan.

Kata kunci: kriptografi, fungsi hash, *password cracking* , SHA-1, dan FPGA

I. PENDAHULUAN

Di era perkembangan teknologi komunikasi dan informasi, keamanan informasi menjadi suatu kebutuhan pokok. Banyak cara yanng dapat dilakukan, salah satunya dengan menggunakan kriptografi. Kriptografi menyediakan layanan keamanan meliputi kerahasiaan, otentikasi, ketersediaan, dan nirpenyangkalan. Salah satu algoritma kriptografi yang sering digunakan adalah SHA-1 (*Secure Hash Algorithm*-1).

SHA-1 merupakan algoritma fungsi hash yang diusulkan oleh *National Institute of Standard and Technology* (NIST) bersama dengan *National Security Agency* (NSA).Salah satu penggunaan algoritma SHA-1 adalah untuk penyimpanan password (*password hashing*). Pada saat ini banyak layanan *software* menyediakan sistem otentikasi yang berdasarkan pada kombinasi identitas pengguna (*username*) dan *password*. *Password* tersebut digenerate oleh pengguna dan kemudian disimpan di lokasi yang aman di dalam sistem. Untuk memastikan *password* tersebut tersimpan dengan aman, maka biasanya digunakan fungsi hash untuk menghitung *digest* dari *password* tersebut dan menyimpannya bersamaan dengan *user credential.*

Sesuai dengan sifat fungsi hash, nilai *digest* tidak mudah dikembalikan, oleh karena itu sangatlah penting bagi seorang lawan untuk dapat mengumpulkan informasi tentang *password* tersebut dan dapat menggambarkan distribusi kemungkinan *password*. Seharusnya *password* yang digunakan seorang user harus memenuhi kriteria sebagai*password* yang kuat(*strong password*). Namun karena keterbatasan kemampuan memori manusia, maka cenderung dipilih *password* yang mudah diingat dan biasanya terkait dengan identitas diri. Hal tersebut akan memungkinkan bagi seorang lawan untuk membangkitkan kandidat *password*, menghitung digest, dan membandingkannya dengan *password* yang disimpan. Skema *password* hashing dibuat sedemikian kompleks sehinga mengurangi kemungkinan serangan yangakan terjadi.

Disisi lain, keamanan dan ancaman terhadap keamanan bagaikan dua sisi mata uang. Perkembangan salah satunya akan mempengaruhi perkembangan yang lain. Pada saat ini banyak teknik/metode yang dapat digunakan untuk memecahkan sistem keamanan baik dengan menggunakan *software* maupun*hardware*. Penggunaan *hardware* dapat meningkatkan efisiensi waktu dan konsumsi power yang lebih rendah dari pada penggunaan *software*[1]. Hukum Moore menyatakan bahwa pertumbuhan kecepatan perhitungan mikroprosesor mengikuti rumusan eksponensial dan kompleksitas sebuah mikroprosesor akan meningkat dua kali lipat tiap 18 bulan sekali. Sesuai dengan hukum Moore, maka sangat diragukan bahwa skema *passwordhashing* termasuk SHA-1masih menyediakan keamanan yang cukup. Apalagi dengan berkembangya perangkat yang didesain khusus (*special purpose hardware*) untuk melakukan komputasi sehingga dapat melakukan komputasi dengan performansi tinggi (*high performance computing*). Salah satu jenis teknologi tersebut adalah FPGA (*Field Programmable Gate Array*). FPGA merupakan *device* yang terdiri dari gerbang-gerbang logic yang *fully programmable*. Penggunaan FPGA lebih hemat biaya dan memiliki performa yang lebih tinggi dibanding *personal computer* dengan nilai biaya yang sama [2].

Instansi XYZ merupakan instansi pemerintah yang bergerak dalam bidang pengamanan informasi yang salah satu misinya adalah menyajikan informasi hasil kriptanalisistermasuk *password cracking* didalamnya. Salah satu jenis algoritma kriptografi yang dikriptanalisis adalah fungsi hash seperti SHA-1 yang salah satu fungsinya digunakan sebagai skema *password hashing*. Instansi XYZ dituntut untuk dapat menyajikan informasi tersebut dengan cepat karena akan dijadikan pertimbangan bagi pimpinan untuk pengambilan keputusan. Dengan semakin kompleksnya *password* yang digunakan pada suatu sistem saat ini, maka dibutuhkan pula sarana komputasi yang memiliki kecepatan tinggi untuk dapat memecahkannya.

Pada saat ini instansi XYZ melakukan *password cracking*menggunakan *personal computer* (PC) dengan *tools*/*software* sehingga waktu komputasi relatif lama.Oleh karena itu, Instansi XYZ memerlukan terobosan baru untuk dapat melakukan kegiatan pemecahan kode lebih cepat sehingga informasi dapat segera disajikan kepada pimpinan.

Pada penelitian ini ditawarkan alternatif solusi berupa rancangan implementasi SHA-1 *password cracking* pada perangkat berbasis FPGA. Dengan diterapkannya alternatif tersebut, diharapkan kegiatan *password cracking* dapat dilakukan dengan lebih cepat sehingga informasi juga dapat segera disampaikan kepada pimpinan untuk selanjutnya digunakan sebagai bahan pertimbangan untuk pengambilan keputusan.

II. LANDASAN TEORI DAN KERANGKA KONSEP PEMIKIRAN

A. *Kriptografi*

Kriptografi dapat didefinisikan sebagai teknik dan metode untuk menjaga kerahasiaan informasi. Kriptografi menyediakan layanan kerahasiaan (*confidentiality*), integritas data (*data integrity*), otentikasi (*otentication*), dan nirpenyangkalan (*nonrepudiation*). Kriptografi memiliki dua proses, yaitu enkripsi dan dekripsi. Enkripsi merupakan proses untuk mengubah teks terang/asli (*plaintext*) menjadi teks sandi (*ciphertext*). Sedangkan dekripsi merupakan kebalikan dari proses enkripsi, yaitu merupakan proses untuk mengubah teks sandi (*ciphertext*) menjadi teks terang (*plaintext*) [3].

Menurut taksonominya, kriptografi dikelompokkan dalam tiga kategori, yaitu:

1) *Symmetric-key primitives*
   Disebut juga sebagai sistem kunci simetris atau kriptografi simetris, yaitu sistem kriptografi yang menggunakan kunci yang sama untuk melakukan enkripsi dan dekripsi.
2) *Asymmetric-key primitives*
   Disebut juga sebagai sistem kunci asimetrik atau kriptografi asimetris, yaitu sistem kriptografi yang menggunakan dua kunci yang berbeda untuk enkripsi dan dekripsi. Kunci publik (*public key*) digunakan untuk enkripsi, dan kunci privat (*prvate key*) untuk dekripsi. Kriptografi asimetris disebut juga sebagai kriptografi kunci publik (*public key cryptography*).
3) *Unkeyed primitives*
   Disebut juga sebagai fungsi hash (*hash function*), yaitu fungsi satu arah (*one-way function*) yang memetakan *plaintext*dengan panjang sembarang menjadi *ciphertext* dengan panjang tetap.

B. *Fungsi Hash*

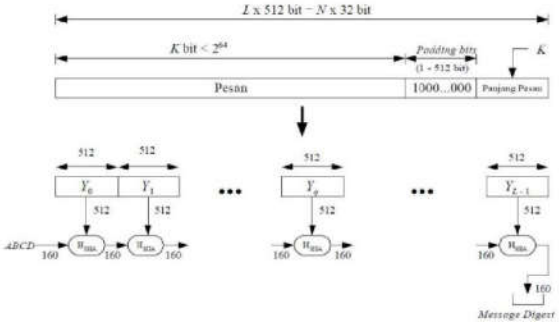
Fungsi hash merupakan fungsi yang memiliki sifat berikut [4]:

1) Kompresi: memetakan *inputx* dengan panjang sembarang, menjadi *outputh(x)* dengan panjang tetap.
2) Mudah dihitung: dengan diberikan fungsi hash *h* dan input *x* maka akan mudah untuk menghitung output *h* (*x*).
3) *Preimage resistance*: dengan diketahui nilai output *h* (*x*), akan sulit untuk menemukan nilai input *x*.
4) *Second preimage resistance*: dengan diketahui nilai output *h(x)* dan input *x* akan sulit untuk menemukan input lain *x'* yang menghasilkan nilai output yang sama.
5) *Collision resistance*: secara komputasi akan sulit untuk menemukan dua input *x* dan *x'* yang memiliki output yang sama.

Fungsi hash banyak dimanfaatkan untuk banyak tujuan, antara lain untuk tabel pencarian, perhitungan *message digest,* fungsi enkripsi, tanda tangan digital, dan penyimpanan *password*.

C. *Skema Algoritma SHA-1*

SHA-1 merupakan fungsi hash yang dirancang oleh *National Institute of Standard and Technology* (NIST) bersama dengan *National Security Agency* (NSA).SHA-1 dapat menerima pesan masukan dengan ukuran maksimum 2^{64} bit (1.147.483.648 *gigabyte*) dan selalu menghasilkan keluaran *message diggest* berukuran 160 bit. Proses pembuatan message diggest ditunjukkan pada Gambar 1 berikut [5].



Gambar 1. Skema SHA-1

Algoritma SHA-1 terdiri dari tahapan berikut:

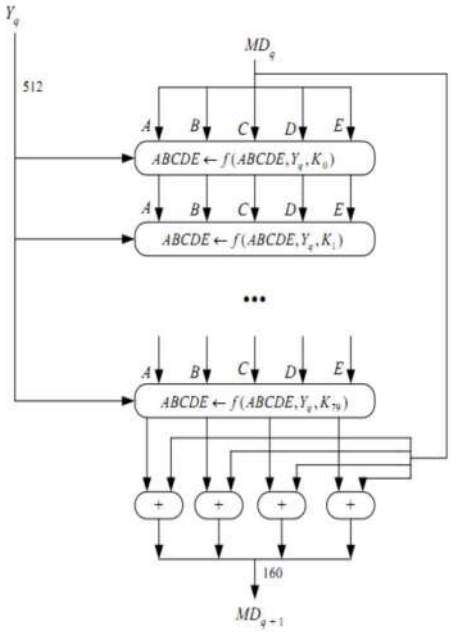
1) Penambahan bit
Pesan *M* dengan panjang *b* bit akan ditambahkan dengan sebuah bit 1 dan diikuti dengan bit 0 hingga panjangnya menjadi kongruen dengan 448 modulo 512.

2) Penambahan ukuran panjang
64 bit yang merepresentasikan panjang pesan akan ditambahkan kedalam pesan yang telah dilakukan penambahan bit sehingga panjang pesan menjadi kelipatan 512 bit.

3) Inisialisasi buffer
SHA-1 membutuhkan lima buah penyangga (*buffer*) yang masing-masih berukuran 32 bit, sehingga totalnya menjadi 160 bit. Kelima *buffer* tersebut menampung hasil antara nilai awal dan nilai akhir.

$$H_0 = \text{x"67452301"}$$
$$H_1 = \text{x"}efcdab89\text{"}$$
$$H_2 = \text{x"}98badcfe\text{"}$$
$$H_3 = \text{x"10325476"}$$
$$H_4 = \text{x"}c3d2e1f0\text{"}$$

4) Pemrosesan pesan
Pesan dibagi kedalam *j* buah blok yang masing-masing panjangnya 512 bit.Setiap blok diproses bersama *buffer* dan menghasilkan output 160 bit.



Gambar 2. Pemrosesan Pesan 512 bit SHA-1

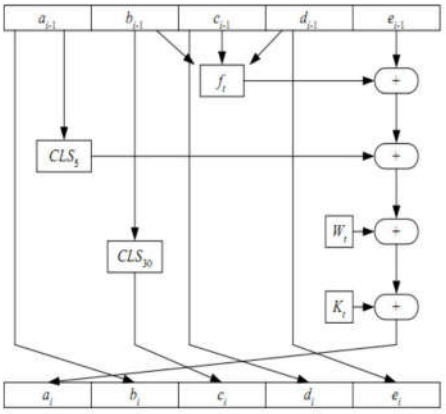
Pada awal proses, didefinisikan lima buah registerberikut:

$$A \leftarrow H_0, B \leftarrow H_1, C \leftarrow H_2, D \leftarrow H_3, E \leftarrow H_4$$

Algoritma memiliki 80 putaran (*round*), dengan masing-masing round *t*, $0 \le t \le 79$ ditambahkan dengan nilai K_t dengan ketentuan berikut:

$$K_t = \begin{cases} \text{x"}5a827999\text{" untuk } 0 \le t \le 19 \\ \text{x"}6ed9eba1\text{" untuk } 20 \le t \le 39 \\ \text{x"}8f1bbcdc\text{" untuk } 40 \le t \le 59 \\ \text{x"}ca62c1d6\text{" untuk } 0 \le t \le 19 \end{cases}$$

Untuk proses satu round SHA-1 ditunjukkan pada Gambar 3 berikut.



Gambar 3. Proses Satu Round SHA-1

Operasi dasar pada gambar tersebut dapat dituliskan dengan persamaan berikut.

$$a, b, c, d, e \leftarrow (CLS_s(a) + f_t(b, c, d) + e + W_t + K_t), a, CLS_{30}(b), c, d$$

dengan CLS_s merupakan pergeseran bit ke kiri sebanyak *s bit*, $W_t$adalah 32 bit word yang diturunkan dari blok 512 bit yang diproses, $f_t$merupakan fungsi *round* ke *t*, dan + merupakan operasi penjumlahan modulo 2^{32}.

Untuk $t < 16$, W_t merupakan 32 bit ke *t* dari pesan M_j. Jika $t \ge 16$, W_t dapat dinyatakan dengan

$$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) \lll 1$$

dengan $\lll$ melambangkan *circular left shift*, dan $\oplus$ adalah operasi XOR.

Fungsi f_t untuk setiap round didefinisikan pada Tabel 1 berikut.

Tabel 1.
Fungsi f_t untuk Setiap Round

Round (t)	$\boldsymbol{f_t(b,c,d)}$
0 ... 19	$(b \wedge c) \vee (\sim b \wedge d)$
20 ... 39	$b \oplus c \oplus d$
40 ... 59	$(b \wedge c) \vee (b \wedge d) \vee (c \wedge d)$
60 ... 79	$b \oplus c \oplus d$

dengan $\wedge, \vee, \oplus, \sim$ merupakan operator logika untuk AND, OR, XOR , dan NOT.

Setelah melalui round 80, dilakukan operasi

$H_0 \leftarrow H_0+A$
$H_1 \leftarrow H_1+B$
$H_2 \leftarrow H_2+C$
$H_3 \leftarrow H_3+D$
$H_4 \leftarrow H_4+E$

5) Output
Output yang dihasilkan adalah 160 bit yang terdiri dari H_0, H_1, H_2, H_3, H_4.

D. *Serangan Terhadap Skema Kriptografi*

Serangan terhadap kriptografi, disebut juga sebagai kriptanalisis merupakan seni dan ilmu untuk mengupas informasi terenkripsi dengan menggunakan metode dan teknik matematik. Beberapa jenis serangan yang ada pada kriptografi antara lain [6]:

1) *Exhaustive search attack/brute force attack.*
Pada jenis serangan ini dicoba semua kemungkinan kunci yang digunakan. Oleh karena itu dibutuhkan waktu yang lama dan membutuhkan sumber daya komputasi yang besar.

2) *Birthday attack* [7]
Ide dasar *birthday attack*adalah dari sekelompok orang yang berjumlah 23 orang, akan terdapat 2 orang yang memiliki tanggal lahir sama dengan probabilitas mendekati 50%. Dengan *birthday attack*, penyerang membangkitkan output dari fungsi hash yang diberikan sampai ditemukan dua input yang memiliki output yang sama (*collision*).

3) *Time-memory trade-off attack*[13]Teknik ini didasarkan pada dibutuhkannya waktu dan sumber daya komputasi yang besar untukmelakukan *exhaustive search.*

E. *Password Cracking*

Terdapat sejumlah metode yang dapat digunakan untuk melakukan *password cracking*. Secara umum, metode dibagi menjadi dua, *offline cracking* dan *online cracking*. *Password cracking* dapat dilakukan dengan beberapa metode berikut.

1) Brute force attack
Metode ini memiliki deskripsi yang sama dengan *exhaustive search.*

2) Dictionary attack
Dictionary attack menggunakan *word list* yang berdasarkan asumsi bahwa pengguna menggunakan *password* yang dapat ditemukan di kamus, atau dapat dikelompokkan.

3) Rule-based approach
Dari *word list* yang ada diterapkan *rule* atau aturan tertentu, sehingga isi dari *word list* menambah kemungkinan *password* dalam *word list.*

4) Rainbow table
Rainbow table merupakan *pre-calculated lookup table* dengan penyimpanan yang lebih cerdas dan kurang nyaman saat dilakukan pencarian. Penggunaan *rainbow table* memiliki kemungkinan yang besar untuk berhasil.

F. *Perkembangan Teknologi Komputasi*

Teknologi komputasi berkembang baik dari sisi *hardware* maupun *software*. Tujuan akhir yang ingin dicapai adalah dapat menyelesaikan permasalahan komputasi dengan cepat.

Hardware ada yang didesain untuk menjalankan banyak fungsi yang disebut sebagai *general purpose hardware*, dan *hardware* yang didesain khusus untuk menjalankan fungsi tertentu (*special purpose hardware*). Hardware yang termasuk dalam *special purpose hardware*antara lain:

1) Cell
Cell merupakan *microprocessor* yang dikembangkan oleh Sony, IBM, dan Toshiba.Arsitektur *Cell* menggabungkan *general purpose processor*dengan *streamlined co-processing elements* yang meningkatkan kecepatan aplikasi multimedia dan *vector processing*. Hash-Clash project menggunakan 200 *cell* untuk menemukan *collision* pada algoritma MD5.

2) Graphical Processing Unit (GPU)
GPU memiliki *graphic cards* yang mendukung operasi aritmatika integer dan pemrograman umum *application programming interface* (API's) sehingga dapat digunakan untuk melakukan perhitungan algoritma kriptografi dan kriptanalisis.

3) Field Programmable Gate Array (FPGA)
FPGA merupakan *integrated circuit* (IC) yang didesain untuk dapat dikonfigurasi oleh pengguna. FPGA terdiri dari komponen logic yang *reprogrammable* (*logic block*), dan *reconfigurable interconnect* yang memungkinkan komponen berkomunikasi. *Logic block* dapat dikonfigurasiuntuk menjalankan fungsi kombinasional yang kompleks, atau gerbang logic sederhana seperti OR, AND, dan XOR sehingga dapat digunakan untuk menerapkan algoritma kriptografi.

III.TINJAUAN STUDI

Berikut ringkasan penelitian sebelumnya yang terkait dengan penggunaan FPGA dalam kriptografi dan kriptanalisis.

1) Penelitian mengenai implementasi SHA pada FPGA oleh Hanamakumar, dkk. Diperoleh hasil bahwa implementasi secara *hybrid* yaitu gabungan *software* dan *hardware* 150 kali lebih cepat daripada implementasi *software* saja [8].

2) Kimmo Jarvinen,2004, mengimplementasikan SHA-1 pada FPGA Xilinx Virtex-II XC2V2000-6. Implementasi membutuhkan 1275 slice, beroperasi pada frekuensi 117,6 MHz memberikan throughput 734 Mbps. Implementasi ini merupakan implementasi yang paling cepat dibanding dengan implementasi sebelumnya [9].

3) Guneysu, 2009. Melakukan implementasi algoritma kriptografi dan kriptanalisis pada FPGA. Pertama, Guneysu menggunakan core fungsi aritmatik yang tersedia di FPGA modern untuk implementasi algoritma kriptografi. Kedua, Guneysu melakukan cryptanalysis dengan menggunakan *cluster* FPGA. Algoritma yang diserang adalah *Data Encryption Standard* (DES) *based system* yaitu token dan ANSI X9.9. Attack menggunakan *cluster* yang terdiri dari 120 FPGA yang dinamakan COPACOBANA. COBACOBANA mampu memecahkan DES *based system*dalam waktu rata-rata 6,4 hari [10].

4) Kumar, Sandeep, dkk melakukan cryptanalysis algoritma DES, ECC, faktorisasi dan RSA dengan menggunakan perangkat berbasis FPGA, COPACOBANA yang terdiri dari 120 FPGA dengan biaya US$ 10.000. Algoritma DES dapat dipecahkan dalam waktu 9 hari dengan kecepatan rata-rata 48.10^9 kunci/detik. Sementara ECC dan RSA belum dapat dipecahkan [11].

5) David Hulton, 2009 dari picocomputing melakukan *password* recovery dengan menggunakan perangkat berbasis FPGA Xilinx Virtex5. Perangkat bekerja dengan frekuensi 100 Mhz dan kecepatan rata-rata10^8 kunci/detik [12].

IV. TINJAUAN OBYEK PENELITIAN

A. *Obyek Penelitian*

Dalam penelitian ini, yang menjadi lokus adalah Instansi XYZ. Instansi XYZ adalah instansi pemerintah yang bergerak dalam bidang pengamanan informasi.

B. *Perangkat Keras*

Dalam penelitian ini digunakan perangkat keras berikut:

1) Komputerdengan spesifikasi:
   a) Prosesor : Intel Core i7 @ 2,1 GHz
   b) Memori : 8 GB
   c) *Hard disk* : 250 GB
   d) Sistem operasi: Linux Backtrack5 R3

2) FPGA SPARTAN 6Developtment Kit dengan spesifikasi:
   a) Jenis : Xilinx Spartan 6 LX150 (XC6LX150)
   b) Memori : Memori 512 MB DDR3
   c) Data rate : 800 Mbps

C. *Perangkat Lunak*

Berikut perangkat lunak yang digunakan dalam penelitian ini.

1) Xilinx ISE [2]
   Xilinx ISE (*integrated software environtment*) mengendalikan semua aspek pengembangan. Xilinx ISE yang digunakan adalah versi 14.3.

2) **ISE *Simulator* (ISim)**
   ISim memungkinkan untuk menjalankan simulasi *behavioral* dan *timing* dari desain dalam *software* ISE *Project Navigator* dengan cepat.

3) **John The Ripper**
   John The Ripper (JTR) merupakan *tool* untuk melakukan *password cracking* yang dapat digunakan sistem operasi Unix, Windows, DOS, BeOS, dan Open VMS [16].

V. PERANCANGAN SISTEM DAN ANALISIS

A. *Perancangan Sistem*

Proses perancangan merupakan tahap awal dalam membangun sebuah sistem. Sistem dirancang dengan tahapan berikut:

1) Permodelan sistem
   Permodelan sistem menggunakan *Data Flow Diagram* (DFD) untukmenggambarkan aliran data yang berjalan dalam sebuah sistem dari satu proses ke proses yang lain. DFD dimulai dari pembuatan *context* diagram yang merupakan level paling atas hingga paling bawah yang tidak bisa diuraikan lagi.
2) Identifikasi komponen datapath
   Pada tahap ini dilakukan identifikasi terhadap komponen-komponen *datapath* yang akan digunakan dalam implementasi SHA-1 *password cracking* pada FPGA.
3) Pembuatan *datapath*
   Diagram datapath yang akan menggambarkan lebih jelas mengenai operasi yang dilakukan pada modul SHA-1 *password cracking*. *Datapath* terdiri dari lebih dari satu komponen dan disusun berdasarkan operasi yang terdapat dalam algoritma SHA-1 *password cracking*.
   Setiap *datapath* komponen yang telah dibuat pada tahap pembuatan *datapath* komponen menjalankan masing-masing menjalankan fungsi tertentu dalam struktur SHA-1 *password cracking*. Pada tahap perancangan arsitektur lengkap, semua *datapath* tersebut diiintegrasikan menjadi satu kesatuan untuk mendapatkan fungsi SHA-1 *passsword cracking*.
4) *Timing diagram*
   Timing diagram menggambarkan jalannya algoritma berdasarkan parameter waktu.

5) Pembuatan *controller*
Untuk dapat mengatur jalannya keseluruhan proses pada SHA-1 *password cracking,* diperlukan adanya komponen yang bertindak sebagai *controller.* Komponen ini dapat dibangun menggunakan menggunakan *state machine*. Pada tahap ini akan dinuat *state diagram* yang akan menggambarkan alur algoritma SHA-1 *password cracking* berdasarkan statenya.

6) Perancangan interface
Interfacing dilakukan pada sisi *host* dan pada FPGA. Pada sisi host interfacing menggunakan pemrograman C, sedangkan pada FPGA menggunakan VHDL.

B. Implementasi Sistem

Tahap implementasi dibagi lagi menjadi beberapa tahapan kecil dengan mengadopsi dari tahapan implementasi pada *Software* Xilinx ISE.

1) *Design entry*
Pada tahap ini dilakukan proses *coding.Coding* dilakukan dengan menggunakan bahasa pemrograman VHDL. *Source code* dibuat untuk setiap komponen, kemudian diintegrasikan menjadi satu rangkaian. Entitas yang digunakan bukan hanya entitas utama, tapi juga entitas yang digunakan untuk interfacing menggunakan API.

2) *Design synthesis*
Proses *synthesis* menghasilkan skematik seperti yang telah digunakan pada *design entry.* Berikut penggunaan *resource*modul SHA-1 *password cracking* yang telah dibuat.

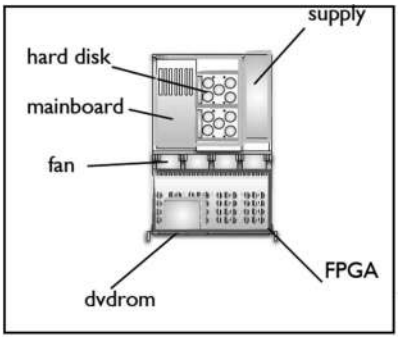
Tabel 2. Penggunaan Resource Modul SHA-1 Cracking.

Resource	Tersedia	Terpakai	Prosentase (%)
Slice registers	184304	9947	5 %
Slice LUT:	92152	7545	8 %
- Logic	92152	6993	7 %
- Memory	21680	552	2 %
LUT Flip Flop:			
- Unused Flip Flop	11345	1398	12 %
- Unused LUT	11345 11347	3800 6147	33 % 54 %
- Fully used LUT-Flip Flop		319	
- Unique control set			
IOB	338	38	11 %
Block RAM/FIFO	268	26	9 %
BUFG/BUFGCTRLs	16	6	37 %
Minimum period : 9, 246 ns (Maximum Frequency: 108,175 MHz) Maximum input required time before clock: 1, 801 ns Maximum output required after clock: 3,597 ns			

3) *Implement design*
Pada tahap ini dilakukan konfigurasi pin input/output untuk sistem SHA-1 *password cracking.* Konfigurasi pin input/output disimpan dalam file dengan ekstensi *.ucf.

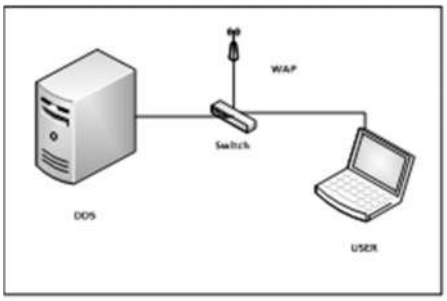
4) *Programming* perangkat
Pada tahap ini akan dibangkitkan sebuah file yang berekstensi *.bit. File berekstensi *.bit tersebut dibangkitkan setelalah dilakukan *implement design* dengan ISE *design suite*. File inilah yang nantinya akan di *download* kedalam *Board* FPGA.

5) Implementasi pada perangkat
Tahap selanjutnya yang dilakukan adalah implementasi pada perangkat DDS (*Design and Developtment System*) berbasis FPGA. DDS yang digunakan memiliki arsitektur berikut.



Gambar 4. Arsitektur DDS

DDS dapat diakses oleh *host*melalui jaringan dengan konfigurasi berikut.



Gambar 5. Konfigurasi Jaringan

C. Pengujian Sistem

Terdapat dua tahap pengujian sistem, yaiu *test bench* dan *test vector. Test bench*dilakukan setelah tahap *design synthesis*. Test bench dilakukan untuk mengetahui apakah pemrograman yang telah dibuat pada setiap modul telahbekerja dengan benar. *Test bench* dijalankan menggunakan Isim (ISE *Simulator*).

Hasil test *vector* FPGA ditunjukkan pada Gambar 6 berikut.

Gambar 6. Hasil Tes *Vector*FPGA

Hasil dari John The Ripper ditunjukkan pada Gambar 7 berikut.

Gambar 7. Hasil JTR

Dari *test vector* diperoleh hasil yang sama antara sistem SHA-1 *password cracking* yang menggunakan FPGA dibandingkan JTR.

D. Analisis Hasil Implementasi

1) Analisis Resource

Jenis dan besarnya resource yang digunakan untuk implementasi sistem dapat dilihat dari *Designsynthesis*. Secara keseluruhan penggunaan *resource*yang ada cukup efisien. Namun secara detail untuk tiap sub modul terdapat penggunaan modul yang sangat besar yaitu 351%. Hal tersebut dikarenakan penggunaan *pipelining*. Penggunaan *pipelining* akan membutuhkan lebih banyak ruang pada modul. Namun disisi lain penggunaan pipelining dapat meningkatkan *clock speed*.

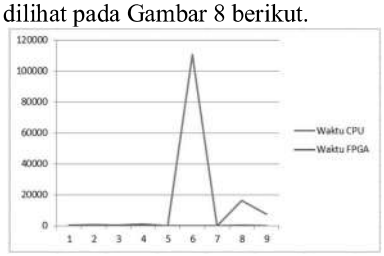
2) Analisis Performa

Analisis performa dilakukan dengan menghitung *throughput* yang digunakan oleh modul SHA-1 *password cracking* untuk memproses hash target yang diberikan sehingga menemukan *plaintext*. Throughput dapat didefinisikan sebagai banyaknya data yang dapat dikirim dalam waktu tertentu. Throughput dapat dihitung dengan menggunakan rumus

$$\boldsymbol{throughput} = \frac{\boldsymbol{jumlah\ bit}}{\boldsymbol{cycle} \times \boldsymbol{Time\ period}}$$

Desain yang dibuat memberikan *throughput*216, 356 Mbps.

Untuk melihat performa secara real, dilakukan perbandingan waktu yang dibutuhkan untuk melakukan SHA-1 *password cracking* dengan menggunakan FPGA dibandingkan dengan menggunakan PC. Sebagai sampel dibangkitkan nilai hash yang merupakan hasil perhitungan *password* yang terdiri dari alfabet, numerik dan kombinasi keduanya. Hasil Perbandingan keduanya dapat dilihat pada Gambar 8 berikut.



Gambar 8. Perbandingan Waktu PC dan FPGA

Dari hasil pengujian dapat disimpulkan bahwa SHA-1 *password cracking* dengan menggunakan PC dengan John The Ripper (JTR).

E. Implikasi Penelitian

1) Aspek Sistem

Makin berkembangnya teknologi komputasi terutama *Field Programmable Gate Array* (FPGA) membuat perangkat tersebut relatif mudah didapatkan dengan harga yang terjangkau di pasaran. Dengan harga yang tidak terlalu mahal, dan jauh lebih murah dibandingkan dengan harga *personal computer* dengan spesifikasi prosesor yang sangat tinggi saat ini, FPGA memiliki performa yang jauh lebih tinggi dalam melakukan komputasi yang kompleks

2) Aspek Organisasi

Mengingat kebutuhan Instansi XYZ yang membutuhkan kecepatan dalam melakukan *password cracking*, hasil penelitian tesis ini dapat menjadi alternatif solusi untuk dapat diterapkan di Instansi XYZuntuk melakukan *password cracking* dengan lebih cepat dan mendukung pencapaian misi organisasi.

3) Aspek Penelitian Lanjutan

Dapat dilakukan penelitian lanjutan antara lain untuk meningkatkan efisiensi agar mengurangi kebutuhan *resource* yang ada,*password cracking* untuk *salted* SHA-1, dan implementasi metode lain seperti menggunakan *dictionary attack*agar *password cracking* dapat dilakukan lebih cepat.

F. Rencana Implementasi

Implementasi dilakukan melalui beberapa tahapan berikut.

1) Melakukan perbaikan efisiensi pemrograman dan penyesuaian desain

Perancangan yang telah dibuat masih menggunakan *resource* yang cukup besar. Untuk itu perlu dilakukan perbaikan dari sisi implementasi pemrograman agar *resource*yang digunakan lebih efisien. Selain itu, perancangan yang telah dibuat akan diimplementasikan pada cluster. Oleh karena itu diperlukan penyesuaian desain agar sistem dapat bekerja dengnan maskimal.

2) Implementasi pada *cluster* FPGA
Setelah dilakukan perbaikan dan penyesuaian desain pada rancangan yang telah dibuat, tahap selanjutnya adalah implementasi pada *cluster* FPGA yang dimiliki Instansi XYZ.
3) Melakukan pengujian pada *cluster* FPGA
Pengujian dilakukan untuk mengetahui apakah rancangan yang telah diimplementasikan pada *cluster* FPGA telah bekerja dengan benar dan sesuai dengan fungsinya. Setelah dipastikan bahwa sistem telah bekerja dengan benar maka selanjutnya dapat dilakukan uji coba oleh Instansi XYZ.
4) Ujicoba oleh Instansi XYZ
Uji coba diperlukan oleh Instansi XYZ untuk mengetahui apakah sistem yang telah dibuat telah sesuai dengan kebutuhan Instansi XYZ. Selain itu uji coba juga dilakukan untuk mengetahui kekurangan dari sistem yang telah dibuat.
5) Review hasil uji coba
Setelah dilakukan uji coba oleh Instansi XYZ, selanjutnya dilakukan review terhadap hasil uji coba yang telah dilakukan. Pada tahap ini dihimpun data yang terkait dengan kekurangan sistem baik secara teknis seperti *bug* dan *erroe*maupun kekurangan secara operasional.
6) Memperbaiki *bug* dan *error* berdasarkan laporan yang diterima
Setelah dilakukan identifikasi kekurangan sistem yang telah dibuat, selanjutnya dilakukan perbaikan terhadap *bug* dan *error* berdasarkan laporan yang diterima dari hasil uji coba yang dilakukan oleh Instansi XYZ.
7) Operasional SHA-1 *password cracking* dengan FPGA oleh Instansi XYZ
Setelah dilakukan perbaikan terhadap kekurangan yang ada pada sistem yang telah dibuat, dan dipastikan sistem telah bekerja dengan benar maka sistem telah siap dioperasionalkan oleh Instansi XYZ untuk melakukan SHA-1 *password cracking.*

VI. KESIMPULAN

SHA-1 banyak digunakan secara luas, salah satunya untuk melakukan penyimpanan *password* (*password hashing*).*Password* merupakan salah satu bentuk otentikasi untuk dapat mengakses sistem. Perkembangan teknologi komputasi seperti *Field Programmable Gate Array* (FPGA) membuat komputasi untuk melakukan *password cracking*semakin cepat. Pada penelitian ini telah dilakukan perancangan implementasi SHA-1 *password cracking* dengan metode *brute force attack* pada perangkat berbasis FPGA. Hasil pengujian menunjukkan bahwa rancanganyang telah dibuat telah bekerja dengan benar dan sesuai fungsinya sehingga dapat dilakukan untuk melakukan *SHA-1 password cracking*.

Dari penelitian yang telah dilakukan, diperoleh hasil bahwa untuk menghitung password yang kompleks, FPGA akan lebih cepat dibandingkan dengan menggunakan *personal computer* dengan tools John The Ripper (JTR). Dengan demikian maka kesimpulan yang diperoleh untuk menjawab pertanyaan penelitian pada perumusan masalah adalah dengan menggunakan FPGA, maka Instansi XYZ dapat melakukan SHA-1 *password cracking*untuk *password* yang kompleks dengan lebih cepat sehingga informasi dapat lebih cepat disampaikan kepada pimpinan guna pengambilan keputusan.

DAFTAR PUSTAKA

[1] Alkalbani, Abdullah Said, Mantoro, Teddy, Osman, Abu., Comparison between RSA Hardware and Software Implementation for WSNs Security Schemes. *Proceeding 3rd International Conference on ICT4M* , (2010): 84-89.

[2] Chu, Pong P., "*FPGA Prototyping by VHDL Examples: Xilinx Spartan -3 Version*", New Jersey: John Wiley & Sons, Inc., 2008.

[3] _____."*FIPS PUB 180 – 4: Secure Hash Standard (SHS)*". Gaithersburg: FIPS Publication. 2012.

[4] Fischer, Thorsten. "Everyday Password Cracking". *IRM Research White Paper*, IRM PLC, (Desember, 2007): 16 pages.

[5] Guneysu, Tim, Erhan."Cryptography and Cryptanalysis on Reconfigurable Devices." Ph.D, diss,. Ruhr-University Bochum, 2009.

[6] Hanamakumar, Veerla, Syamsundar. "Cost-Efficient SHA Hardware Accelerator". *IJAEST,* vol.no.5, (2011): 37-52.

[7] [Hulton 2009] Hulton, David, Hur, Mark. "Using FPGA Cluster for Fast Password Recovery". *A Pico Computing Data Security White Paper.*(November, 2009): 4 pages.

[8] Jarvinen, Kimmo. "Design and Implementation of a SHA-1 Module on FPGAs". 2004.

[9] Kumar, asndeep, Paar, Christof. "Breaking Ciphet with COPACOBANA – A Cost-Optimized Parallel Code Breaker". *IACR Proceeding.* 2006.

[10] Kusbandono, Arif., "Implementasi Algoritma AES-128 padaMikrokontroler 8051." Bachelor Thesis, Institut Teknologi Bandung,Bandung, 2004.

[11] Menezes, Alfred J., Oorschot, Paul C. , "*Handbook of Applied Cryptography*", Waterloo: CRC Press, 1996.

[12] Marechal, Simon. "Advances in password cracking" . *J Comput Virol*, vol.4, (2008):73-81