RANCANG BANGUN APLIKASI ANTIVIRUS KOMPUTER DENGAN MENGGUNAKAN METODE *SECURE HASH ALGORITHM* 1 (SHA1) DAN *HEURISTIC STRING*

I Gusti Made Panji Indrawinatha[1], Made Sudarma[2], I Made Arsa Suyadnya[3]
[123] Jurusan Teknik Elektro, Fakultas Teknik, Universitas Udayana
Email: panjiindrawinatha@yahoo.com[1], msudarma@unud.ac.id[2], arsa.suyadnya@unud.ac.id[3]

Abstrak

Virus komputer merupakan perangkat lunak berbahaya yang dapat merusak data dan menggandakan diri pada sistem komputer. Untuk mendeteksi dan membersihkan virus dari sistem komputer, maka dibuatlah aplikasi antivirus. Dalam mendeteksi berbagai jenis virus sebuah aplikasi antivirus biasanya menggunakan beberapa metode. Pada penelitian ini akan membahas perancangan sebuah aplikasi antivirus menggunakan metode Secure Hash Algorithm 1 (SHA1) dan heuristic string sebagai metode pendeteksian virus. Dari pengujian yang dilakukan diperoleh hasil dimana saat tidak menggunakan heuristic, antivirus hanya mendeteksi 12 file dari 34 file sample virus atau memiliki tingkat akurasi pendeteksian sebesar 35%. sedangkan saat menggunakan heuristic, antivirus berhasil mendeteksi 31 file dari 34 file sample virus atau memiliki tingkat akurasi pendeteksian sebesar 91%.

***Kata Kunci** : Antivirus, virus, signature, heuristic string, SHA1.*

1. PENDAHULUAN

Pada era globalisasi seperti saat ini, teknologi komputer telah mengalami perkembangan yang sangat pesat. Namun pesatnya perkembangan teknologi komputer juga diikuti dengan tingginya kejahatan *cyber* yang dilakukan oleh pihak-pihak yang ingin merusak kecanggihan teknologi tersebut. salah satu kasus yang sering ditemui adalah berkembangnya virus komputer. Komputer memang rawan terinfeksi virus, apalagi perkembangan virus juga semakin canggih sehingga *user* tidak bisa mendeteksi apakah komputernya terkena virus. Dengan demikian dibutuhkan *software* antivirus sebagai salah satu solusi mencegah penyebaran virus.

Metode pendeteksian virus yang paling sering digunakan oleh pembuat antivirus yaitu metode pencocokan CRC (*Cyclic Redundancy Check*). Hal ini dikarenakan metode CRC sangat sensitif dengan variasi bilangan hexa 8 digit. Namun dalam perkembangannya metode ini mulai ditinggalkan sebab kelemahan sebuah antivirus dengan metode CRC ini yaitu hanya mengandalkan *signature* virus pada *database*, sehingga metode ini tidak cocok apabila diterapkan pada virus yang mampu melakukan teknik *polymorph* atau mengubah karakteristiknya setelah menginfeksi [1].

Selain metode CRC ada juga metode lain yang bisa digunakan yaitu metode pendeteksian *heuristic*. Metode *heuristic* sangat baik dalam mendeteksi virus walaupun virus tersebut melakukan modifikasi terhadap *byte-byte* tertentu pada tubuh virus (*polymorph*) [2].

Berdasarkan hal tersebut di atas maka pada penelitian ini dibuatlah implementasi metode *scanning Secure Hash Algorithm* 1 (SHA1) dan *heuristic string* dalam perancangan sebuah aplikasi antivirus. Metode SHA1 digunakan untuk mengenali virus dengan cara mencocokkan nilai *checksum* SHA1 *file* dengan nilai *checksum* SHA1 virus yang tersimpan dalam *database*. Sedangkan metode *heuristic string* digunakan untuk mengenali virus berdasarkan *string* virus yang terkandung dalam sebuah *file*. Metode *heuristic* sangat berguna untuk mengantisipasi virus-virus yang menyusup dalam sebuah *file* program dan juga memungkinkan aplikasi antivirus untuk mendeteksi varian virus baru.

2. KAJIAN PUSTAKA

2.1 Virus Komputer

Virus komputer secara umum diartikan sebagai perangkat lunak atau program apapun yang mampu mempengaruhi kinerja komputer dan bersifat merugikan bagi para penggunanya karena komputer yang terinfeksi virus tidak akan bisa bekerja secara normal. Virus komputer juga dapat mereplikasi dirinya sendiri serta menyebar dengan cara menginfeksi program atau

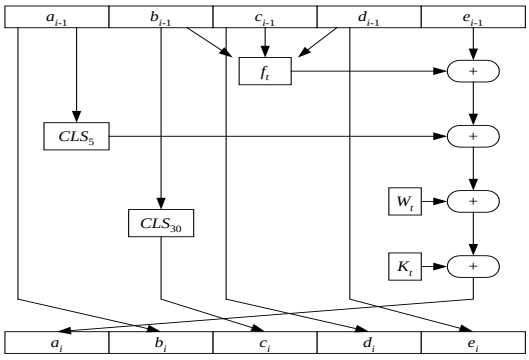
dokumen lain. Sebagian besar virus komputer menyerang sistem operasi Windows dan sisanya menyerang sistem operasi lain seperti FreeBSD dan Sun Operating Syste*m* [3]

2.2 Aplikasi Antivirus

Serangan berbagai jenis virus atau *malware* dapat diantisipasi menggunakan *software* antivirus. *Software* antivirus ini dapat mendeteksi dan membersihkan virus yang menginfeksi sebuah komputer. *Software* antivirus umumnya menggunakan dua teknik dalam mendeteksi berbagai jenis virus atau *malware*, yaitu memeriksa *file* kemudian mencocokkannya dengan kamus virus untuk mengetahui jenis virusnya dan mengidentifikasi perilaku mencurigakan dari program lain di komputer yang menunjukkan bahwa program tersebut terinfeksi. Berbagai antivirus yang banyak beredar biasanya menggunakan kombinasi dari kedua metode ini. Namun, yang paling banyak adalah dengan menggunakan virus *dictionary* atau kamus virus [3].

2.3 *Secure Hash Algorithm* 1 (SHA1)

Secure Hash Algorithm (SHA) dikembangkan oleh *National Institute of Standards and Technology* (NIST) dan *National Security Agency* (NSA) sebagai komponen *Digital Signature Standard* (DSS). *Hash* merupakan algoritma enkripsi untuk mengubah *text* menjadi deretan karakter acak. *Hash* termasuk enkripsi satu arah, dimana hasil dari *hash* tidak dapat dikembalikan kebentuk *text* asli. *Secure Hash Algorithm* 1 merupakan salah satu algoritma *hashing* yang sering digunakan untuk enkripsi data. Hasil dari SHA1 adalah data dengan panjang 20 byte atau 160 bit, biasa ditampilkan dalam bentuk bilangan heksadesimal 40 digit dari masukan dengan ukuran maksimum 2^{64} bit (2.147.483.648 gigabyte) [4]. Operasi dasar SHA ditunjukkan pada Gambar 1.



Gambar 1 Operasi dasar SHA

Gambar 1 dapat dijelaskan sebagai berikut : a, b, c, d, e merupakan lima buah penyangga 32-bit, f_t merupakan fungsi logika, CLS_s merupakan circular left shift sebanyak s bit, K_t merupakan konstanta penambah dan + adalah operasi modulo 2^{32}.

2.4 *Heuristic String*

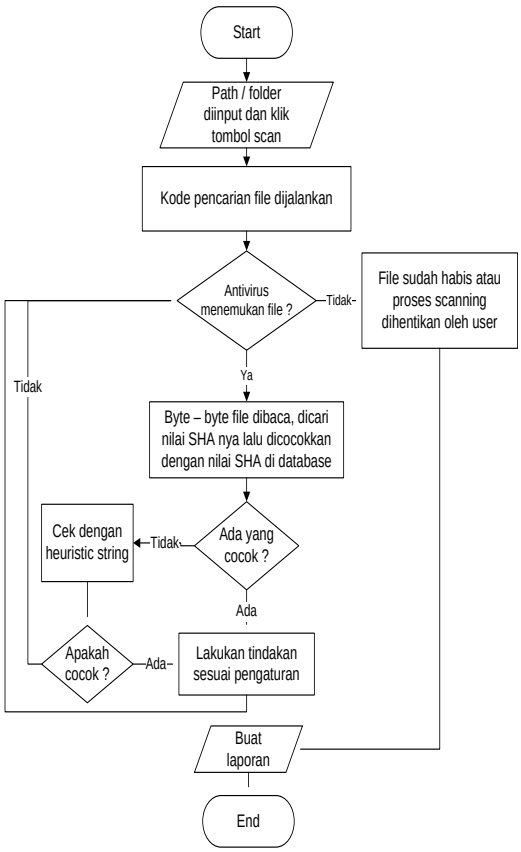
Heuristic String merupakan teknik pendekatan untuk mencurigai bahwa sebuah *file* merupakan sebuah virus atau bukan berdasarkan *string-string* virus pada *file* tersebut. *Heuristic string* sangat berguna untuk mengatisipasi virus-virus yang menyusup dalam sebuah *file* program lain [1].

3. METODE PENELITIAN

3.1 Perancangan Aplikasi Antivirus

1. *Flowchart* Antivirus

Flowchart dari antivirus yang dibangun, ditunjukkan pada Gambar 2.



Gambar 2 *Flowchart* Antivirus

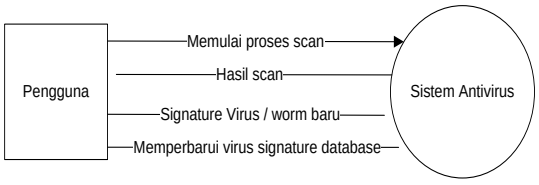
Gambar 2 dapat dijelaskan sebagai berikut :

- Pada awalnya, antivirus dijalankan oleh pengguna, lalu pengguna memasukkan *path* atau alamat *folder* yang akan diperiksa.

- Selanjutnya antivirus melakukan pencarian *file* satu per satu pada *path* yang diberikan. Apabila menemukan sebuah *file*, *byte-byte file* tersebut dibaca untuk didapatkan *checksum* SHA1nya kemudian dicocokkan dengan *signature* virus pada *database*.
- Apabila ada yang cocok, akan dilakukan langkah-langkah, misalnya, menghapus *file* tersebut atau sekedar ditampilkan informasi *file* yang ditemui pada *listview*. Lalu proses selanjutnya adalah kembali melakukan pencarian *file* yang lain.
- Apabila nilai *checksum file* tidak ada yang cocok dengan virus *signature* yang ada dalam *database*, antivirus akan menggunakan teknik *heuristic string* untuk melakukan pendeteksian pada *file* yang sedang diproses. Jika berdasarkan *heuristic file* tersebut adalah virus, maka akan dilakukan tindakan sesuai pengaturan, atau hanya ditampilkan pada *listview*.
- Apabila berdasarkan *heuristic string* tetap tidak menemukan kecurigaan, proses akan meloncat ke pencarian *file* selanjutnya. Jika *file* yang sedang diproses sudah habis atau belum habis, tetapi sengaja dihentikan oleh pengguna ketika proses *scanning*, proses langsung menuju ke tahap terakhir.

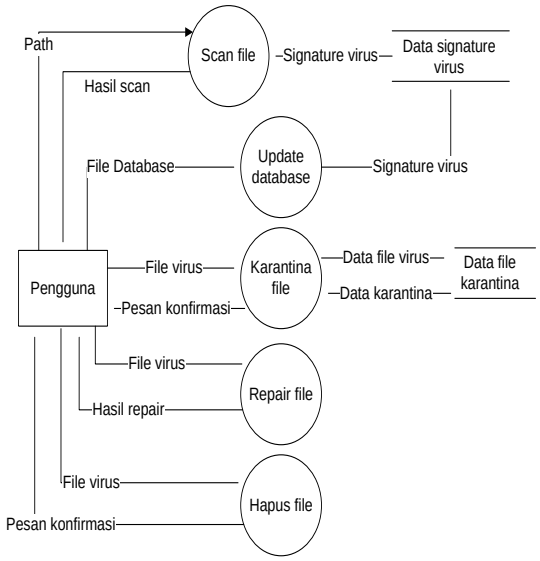
2. *Data Flow Diagram*

Dalam merancang sebuah sistem, langkah awal yang perlu dilakukan adalah membuat *contex diagram*. *Context diagram* dari sistem antivirus yang dibangun dapat dilihat pada Gambar 3.



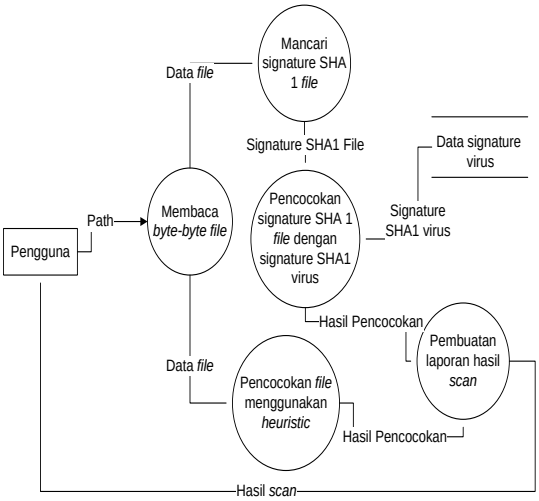
Gambar 3 *Context Diagram*

Dari *Context Diagram*, penggambaran proses-proses dalam sistem bisa lebih diperluas lagi dengan pembuatan *Data Flow Diagram level* 0. Aplikasi antivirus ini memiliki lima proses utama yaitu proses *scan file*, *update database*, *repair file*, karantina *file*, dan hapus *file*. *Data Flow Diagram level* 0 antivirus dapat dilihat pada gambar 4.



Gambar 4 Data *Flow Diagram Level 0* Sistem Antivirus

Data flow diagram level 1 merupakan pengembangan dari *data flow diagram level* 0. Gambar 5 merupakan *data flow diagram level* 1 proses *scan file* aplikasi antivirus.

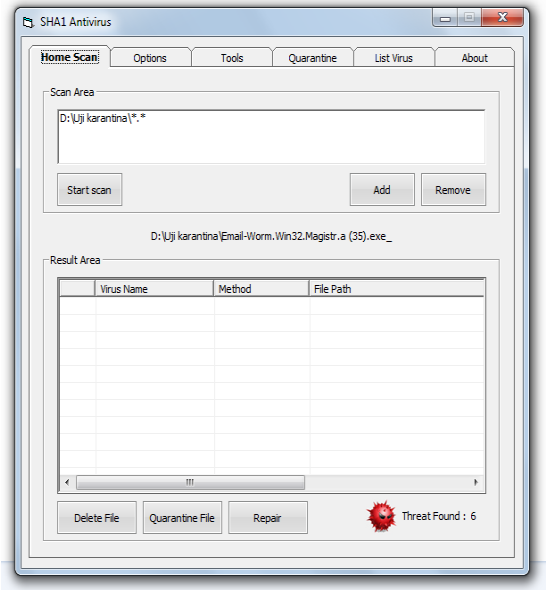


Gambar 5 *Data Flow Diagram Level* 1 *Proses Scan File*

4. HASIL DAN PEMBAHASAN

4.1 Implementasi Antarmuka Aplikasi

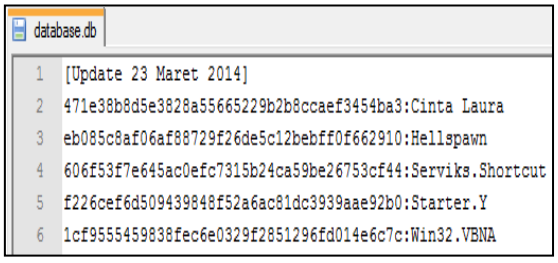
Aplikasi antivirus telah berhasil dibuat dengan antarmuka halaman utama seperti Gambar 6. Melalui halaman utama pengguna dapat memasukkan *path* atau alamat *folder* dan melakukan *scanning* sesuai *path* yang diberikan. Beberapa *menu* yang tersedia pada aplikasi antivirus antara lain *Menu Home Scan* atau halaman utama, *Menu Options*, *Menu Tools*, *Menu Quarantine*, *Menu List Virus*, dan *Menu About*.



Gambar 6 Halaman Utama

4.2 Implementasi Virus *Signature*

Database dari aplikasi antivirus yang dibangun merupakan *file* terpisah dari program utama antivirus. Di dalam *database* tersebut tersimpan virus *signature* yang berupa 40 digit bilangan hexadesimal dan kemudian diikuti oleh nama virusnya. Berikut ini adalah *format database* yang digunakan pada sistem antivirus yang dibangun



Gambar 9 *Format Database* Antivirus

4.3 Pengujian Pendeteksian Virus

Pengujian pendeteksian virus bertujuan untuk mengetahui tingkat akurasi pendeteksian antivirus dengan metode *secure hash algorithm* 1 (SHA1) dan *heuristic string*. Pengujian ini dilakukan dengan cara melakukan *scanning* pada sebuah *folder* yang di dalamnya terdapat 34 *file sample* virus. proses *scanning* dilakukan dua kali dimana pada proses *scanning* pertama pendeteksian *heuristic* dalam keadaan tidak aktif dan pada proses *scanning* kedua pendeteksian *heuristic* dalam keadaan aktif. Untuk jenis virus yang dijadikan *sample* pada pengujian ini dapat dilihat pada Tabel 1.

Tabel 1 *Sample* Virus Pada Pengujian Pendeteksian Antivirus SHA1 Dan *Heuristic String*

No	Nama virus	Jumlah *file sample*
1	Cinta Laura	4
2	Worm.VBS.Evan	1
3	Win32.Bater.a	1
4	Win32.Borzella	1
5	Win32.Codered.D	1
6	Win32.Fizzer	1
7	Win32.Gift	1
8	Win32.Happy	1
9	Win32.Lara	1
10	Win32.Alman	5
11	Win32.Sality	17

Perbandingan hasil *scanning* saat menggunakan *heuristic* dan tanpa menggunakan *heuristic* dapat dilihat pada Tabel 2

Tabel 2 Hasil Pengujian Antivirus Tanpa Metode *Heuristic* dan Saat Menggunakan Metode *Heuristic*

Status Pemeriksaan *Heuristic*	Jumlah *File* yang Terdeteksi	Jumlah *Sample Virus/Worm* yang Ada	Akurasi Pendeteksian
Tidak aktif	12	34	12/34*100 = 35%
Aktif	31	34	31/34*100 = 91%

Pada Tabel 3 berikut ini merupakan jenis virus yang berhasil dideteksi pada saat melakukan *scanning* tanpa *heuristic*

Tabel 3 Jenis Virus Yang Terdeteksi Oleh Program Antivirus Saat Tidak Menggunakan *Heuristic*

No	Nama virus	Jumlah *file sample*	Jumlah *file* terdeteksi
1	Cinta Laura	4	4
2	Worm.VBS.Evan	1	1
3	Win32.Bater.a	1	1
4	Win32.Borzella	1	1
5	Win32.Codered.D	1	1
6	Win32.Fizzer	1	1
7	Win32.Gift	1	1
8	Win32.Happy	1	1
9	Win32.Lara	1	1
10	Win32.Alman	5	0
11	Win32.Sality	17	0

Pada proses *scanning* yang hanya mengandalkan *checksum* SHA1 dan tanpa menggunakan *heuristic*, aplikasi antivirus tidak mampu mendeteksi virus Win32.Alman dan Win32.Sality. Hal ini dikarenakan virus Win32.Alman dan Win32.Sality tidak terdapat SHA1 *checksum*nya pada *database*. Pada Tabel 4 berikut ini merupakan jenis virus yang berhasil dideteksi pada saat melakukan *scanning* menggunakan *heuristic*.

Tabel 4 Jenis Virus Yang Terdeteksi Oleh Program Antivirus Saat Menggunakan *Heuristic*

No	Nama virus	Jumlah *file sample*	Jumlah *file* terdeteksi
1	Cinta Laura	4	4
2	Worm.VBS.Evan	1	1
3	Win32.Bater.a	1	1
4	Win32.Borzella	1	1
5	Win32.Codered.D	1	1
6	Win32.Fizzer	1	1
7	Win32.Gift	1	1
8	Win32.Happy	1	1
9	Win32.Lara	1	1
10	Win32.Alman	5	5
11	Win32.Sality	17	14

Pada proses *scanning* menggunakan *checksum* SHA1 dan *heuristic*, aplikasi antivirus berhasil mendeteksi virus Win32.Alman dan Win32.Sality. Hal ini menunjukkan walaupun *checksum* SHA1 dari virus Win32.Alman dan Win32.Sality tidak terdapat dalam *database*, dengan metode *heuristic* kedua jenis virus tersebut mampu dideteksi dengan cukup baik.

4.4 Perbandingan *Scanning* SHA1 AV Dengan Beberapa Antivirus Internasional

Adapun perbandingan hasil *scanning* antara aplikasi antivirus yang dibangun (SHA1 AV) dengan beberapa antivirus internasional adalah sebagai berikut.

1. Perbandingan Untuk Jenis Virus Yang Terdeteksi Baik Oleh Antivirus Internasional

Pada Tabel 6 menunjukkan perbandingan *scanning* antivirus yang dibangun (SHA1 AV) dengan beberapa antivirus internasional antara lain Avira, Avast, dan AVG. Adapun *sample* virus yang digunakan adalah jenis virus yang mampu dideteksi dengan baik oleh antivirus internasional dan untuk lebih jelas mengenai *sample* virus yang digunakan dapat dilihat pada Tabel 5.

Tabel 5 Jenis virus yang mampu dideteksi dengan baik oleh antivirus internasional

No	Nama Virus	Jumlah *File Sample*
1	Win32.Alman	5
2	Win32.Sality	18
3	VBS.SSIWG.a	1
4	Worm.VBS.Solow	1
5	VBS.Homepage	1
6	VBS.VBSWG	1
7	Win32.Bagle.y	1

Beberapa parameter yang menjadi acuan dalam perbandingan *scanning* antivirus ini diantaranya adalah jumlah *file sample*, *scanning time (m:s)*, *virus detected*, virus *quarantine/delete*, *error quarantine/delete* dan *detected ratio*(%).

Tabel 6 Perbandingan *Scanning* Antivirus Untuk Jenis Virus Yang Terdeteksi Baik Oleh Antivirus Internasional

No	Perbandingan	Antivirus			
		Avira	Avast	AVG	SHA1 AV
1	*File Sample*	28	28	28	28
2	*Scanning Time (m:s)*	00:05	00:04	00:05	00:14
3	*File Detected*	28	28	28	22
4	*File Quarantine/Delete*	28	28	28	22
5	*Error Quarantine/Delete*	0	0	0	0
6	*Detected Ratio (%)*	100%	100%	100%	78%
$Ratio\ (\%) = \left(\frac{file\ detected}{file\ sample}\right).100$					

Pada Tabel 6 dapat dilihat perbandingan hasil *scanning* antivirus, pada parameter pertama semua antivirus diuji dengan *file sample* sebanyak 28 *file* virus. Pada parameter kedua yaitu *scanning time*, Avira melakukan proses *scanning* dengan waktu 8 detik, Avast dengan waktu 4 detik, AVG dengan waktu 5 detik, dan SHA1 AV dengan waktu 14 detik. Pada parameter ketiga yaitu *file detected*, Avira mampu mendeteksi semua *file sample*, Avast mampu mendeteksi semua *file sample*, AVG mampu mendeteksi semua *file sample*, dan SHA1 AV hanya mampu mendeteksi 22 *file sample* yang mana 22 *file sample* tersebut adalah 5 *file* virus Win32.Alman, 14 *file* virus Win32.Sality, 1 file virus VBS.SSIWG.a, 1 *file* virus VBS.VBSWG, dan 1 *file* virus Win32.Bagle.y. Pada parameter keempat dan kelima, semua antivirus dapat mengkarantina /menghapus semua *file* virus yang dideteksinya dan tidak mengalami *error* saat mengkarantina/menghapus *file* virus yang tersebut. Pada parameter keenam yaitu *detected ratio*, Avira memiliki *ratio* 100%, Avast memiliki *ratio* 100%, AVG memiliki *ratio* 100%, dan SHA1 AV memiliki *ratio* 78%.

2. Perbandingan Untuk Jenis Virus Yang Terdeteksi Dengan Baik Oleh Antivirus SHA1 AV

Pada Tabel 8 menunjukkan perbandingan *scanning* antivirus yang dibangun (SHA1 AV) dengan beberapa antivirus internasional. Adapun *sample* virus yang digunakan adalah jenis virus yang mampu dideteksi dengan baik oleh SHA1 AV

dan untuk lebih jelas mengenai *sample* virus yang digunakan dapat dilihat pada Tabel 7.

Tabel 7 Jenis virus yang mampu dideteksi dengan baik oleh SHA1 AV

No	Nama Virus	Jumlah *File Sample*
1	Win32.Cosmu	6
2	Wsar.A	5
3	Win32.Bater.A	2
4	Win32.Fizzer	1
5	Win32.Nimda.A	2
6	Win32.Plage.A0	2
7	Win32.Navidad.B0	2
8	Win32.Mumu.B	1

Tabel 8 Perbandingan *Scanning* Antivirus Untuk Jenis Virus Yang Terdeteksi Baik Oleh SHA1 AV

No	Perbandingan	Antivirus			
		Avira	Avast	AVG	SHA1 AV
1	*File Sample*	21	21	21	21
2	*Scanning Time (m:s)*	00:05	00:03	00:04	00:13
3	*File Detected*	20	21	20	21
4	*File Quarantine/Delete*	20	21	20	21
5	*Error Quarantine/Delete*	0	0	0	0
6	*Detected Ratio (%)*	98%	100%	98%	100%
Ratio (%) = $\left(\frac{file\ detected}{file\ sample}\right)$.100					

Pada Tabel 8 dapat dilihat perbandingan hasil *scanning* antivirus, pada parameter pertama semua antivirus diuji dengan *file sample* sebanyak 21 *file* virus. Pada parameter kedua yaitu *scanning time*, Avira melakukan proses *scanning* dengan waktu 5 detik, Avast dengan waktu 3 detik, AVG dengan waktu 4 detik, dan SHA1 AV dengan waktu 13 detik. Pada parameter ketiga yaitu *file detected*, Avira tidak mendeteksi 1 *file* virus yaitu Win32.Mumu.B, Avast mendeteksi semua *file sample*, AVG tidak mendeteksi 1 *file* virus yaitu Win32.Mumu.B, dan SHA1 AV mendeteksi semua *file sample*. Pada parameter keempat dan kelima, semua antivirus dapat mengkarantina /menghapus semua *file* virus yang dideteksinya dan tidak mengalami *error* saat mengkarantina/menghapus *file* virus yang tersebut. Pada parameter keenam yaitu *detected ratio*, Avira memiliki *ratio* 98%, Avast memiliki *ratio* 100%, AVG memiliki *ratio* 98%, dan SHA1 AV memiliki *ratio* 100%.

4.5 Perbandingan Hasil *Scanning* Antivirus Lokal Smadav dengan Antivirus SHA1 AV

Adapun hasil perbandingan hasil *scanning* antara aplikasi antivirus yang dibangun (SHA1 AV) dengan antivirus lokal Smadav adalah sebagai berikut.

1. Perbandingan Untuk Jenis Virus Yang Terdeteksi Dengan Baik Oleh Antivirus Smadav

Pada Tabel 10 menunjukkan perbandingan *scanning* antivirus yang dibangun (SHA1 AV) dengan antivirus lokal smadav. Adapun *sample virus* yang digunakan adalah jenis virus yang mampu dideteksi dengan baik oleh antivirus Smadav dan untuk lebih jelas mengenai *sample* virus yang digunakan dapat dilihat pada Tabel 9.

Tabel 9 Jenis virus yang mampu dideteksi dengan baik oleh Antivirus Lokal Smadav

No	Nama Virus	Jumlah *File Sample*
1	VB.Stub.B	1
2	Brontok.A	1
3	Serviks.Shortcut	4
4	MSO.Shortcut	5
5	Win32.Gift	1
6	VBS.Yuyun.B	1
7	Hellspawn	1
8	Getraw.C	1
9	FakeFile.A	17
10	Trojan.Win32.Generic!BT	1

Tabel 10 Perbandingan *Scanning* Antivirus Smadav dan SHA1 AV Untuk Jenis Virus Yang Terdeteksi Baik Oleh Antivirus Smadav

No	Perbandingan	Antivirus	
		Smadav	SHA1 AV
1	*File Sample*	33	33
2	*Scanning Time (m:s)*	00:03	00:16
3	*File Detected*	33	22
4	*File Quarantine/Delete*	33	22
5	*Error Quarantine/Delete*	0	0
6	*Detected Ratio (%)*	100%	66%
Ratio (%) = $\left(\frac{file\ detected}{file\ sample}\right)$.100			

Pada Tabel 10 dapat dilihat perbandingan hasil *scanning* antivirus, pada parameter pertama semua antivirus diuji dengan *file sample* sebanyak 33 *file* virus. Pada parameter kedua yaitu *scanning time*, Smadav melakukan proses *scanning* dengan waktu 3 detik dan SHA1 AV dengan waktu 16 detik. Pada parameter ketiga yaitu *file detected*, Smadav mampu mendeteksi semua *file sample*, dan SHA1 AV hanya mampu mendeteksi 22 *file sample* yaitu 16 *file* FakeFile.A, 4 *file* Serviks Shortcut, 1 *file* Hellspawn, dan 1 *file* Win32.Gift. Pada parameter keempat dan kelima, semua antivirus dapat mengkarantina /menghapus semua *file* virus yang dideteksinya dan tidak mengalami *error* saat mengkarantina /menghapus *file* virus yang tersebut. Pada parameter keenam yaitu *detected ratio*, Smadav memiliki *ratio* 100%, dan SHA1 AV memiliki *ratio* 66%.

2. Perbandingan Untuk Jenis Virus Yang Terdeteksi Dengan Baik Oleh Antivirus Smadav

Pada Tabel 11 menunjukkan perbandingan *scanning* antivirus yang dibangun (SHA1 AV) dengan antivirus lokal smadav. Adapun *sample virus* yang digunakan adalah jenis virus yang mampu dideteksi dengan baik oleh antivirus Smadav dan untuk lebih jelas mengenai *sample* virus yang digunakan dapat dilihat pada Tabel 7.

Tabel 11 Perbandingan *Scanning* Antivirus Smadav dan SHA1 AV Untuk Jenis Virus Yang Terdeteksi Baik Oleh SHA1 AV

No	Perbandingan	Antivirus	
		Smadav	SHA1 AV
1	*File Sample*	21	21
2	*Scanning Time (m:s)*	00:03	00:12
3	*File Detected*	15	21
4	*File Quarantine/Delete*	15	21
5	*Error Quarantine/Delete*	0	0
6	*Detected Ratio (%)*	71%	100%
$Ratio\ (\%) = \left(\frac{file\ detected}{file\ sample}\right).100$			

Pada Tabel 11 dapat dilihat perbandingan hasil *scanning* antivirus, pada parameter pertama semua antivirus diuji dengan *file sample* sebanyak 21 *file* virus. Pada parameter kedua yaitu *scanning time*, Smadav melakukan proses *scanning* dengan waktu 4 detik dan SHA1 AV dengan waktu 12 detik. Pada parameter ketiga yaitu *file detected*, SHA1 AV mampu mendeteksi semua *file sample,* Smadav hanya mampu mendeteksi 15 *file sample* yaitu 6 *file* virus Win32.Cosmu, 5 *file* virus Wsar.A, 2 *file* virus Win32.Plage.A0, dan 2 *file* virus Win32.Nimda.A. Pada parameter keempat yaitu virus *quarantine/delete*, semua antivirus dapat mengkarantina/menghapus semua *file* virus yang dideteksinya. Pada parameter kelima yaitu *error quarantine/delete* semua antivirus tidak mengalami *error* saat mengkarantina/menghapus *file* virus yang dideteksinya. Pada parameter keenam yaitu *detected ratio*, Smadav memiliki *ratio* 40%, dan SHA1 AV memiliki *ratio* 100%.

5. KESIMPULAN

Setelah dilakukan pengujian dan analisis terhadap antivirus yang dikembangkan, maka dapat ditarik simpulan sebagai berikut.

1. Dari pengujian pendeteksian antivirus didapatkan hasil dimana metode deteksi *checksum* SHA1 yang dibarengi dengan *heuristic string* memiliki akurasi pendeteksian yang lebih baik dari pada hanya menggunakan metode *checksum* SHA1 tanpa *heuristic string*.
2. Metode *heuristic string* memungkinkan sebuah antivirus mampu mendeteksi jenis virus yang belum tercatat dalam *database* antivirus.
3. Dari perbandingan hasil *scanning* dengan beberapa produk antivirus, aplikasi antivirus yang dibangun (SHA1 AV) dan antivirus lokal Smadav terlihat saling melengkapi dalam hal jenis virus yang mampu dideteksi oleh masing-masing antivirus. Sedangkan antivirus internasional mampu mendeteksi virus dengan sangat baik meskipun tidak dipasang bersama dengan antivirus lain.

6. DAFTAR PUSTAKA

[1] Hirin, A. M. 2010. *Cara Praktis Membuat Antivirus Komputer*. Jakarta: Mediakita

[2] Pradana, D. 2012. Rancang Bangun Software Antivirus dengan menggunakan Metode Pendeteksian Heuristic. Vol. 6, No 3, September. Bandar Lampung.

[3] Wahana Komputer. 2011. *Mudah Basmi Virus, Spam dan Malware dengan Free Antivirus Online*. Yogyakarta: Andi

[4] Dewanagn. R. R. 2015. Implementation of Secure Hash Algorithm Using JAVA Programming. Vol. 02, No 2, November. Bhilai, India..