JIKO (Jurnal Informatika dan Komputer)
Vol. 6, no. 2, September 2022, hal. 185–193
e-ISSN : 2477-3964 — p-ISSN : 2477-4413

ANALISIS VALIDASI FILE UPLOAD MENGGUNAKAN METADATA PNG PADA APLIKASI BERBASIS WEB

Fahmi Anwar [1], **Abdul Fadlil**[2], dan **Imam Riadi**[3]
[1]Program Studi Magister Teknik Informatika, Universitas Ahmad Dahlan, Yogyakarta
[2]Program Studi Teknik Elektro, Universitas Ahmad Dahlan, Yogyakarta
[3]Program Studi Sistem Informasi, Universitas Ahmad Dahlan, Yogyakarta
Email: fahwar95@gmail.com[1], fadlil@ee.uad.ac.id[2], imam.riadi@is.uad.ac.id[3]

Abstrak

Penggunaan Website pada zaman sekarang sedang berkembang pesat, berbagai jenis aplikasi, layanan dari swasta sampai pemerintah banyak menggunakan website sebagai media interaksi antar berbagai platform. Biasanya saat mengunggah berkas pada website, sistem melakukan validasi dengan menyaring jenis berkas objek digital di server side (backend) atau dalam halaman website pada web browser dalam bentuk HTML atau Javascript (frontend) dengan memanfaatkan penyaringan Header atau Multipurpose Internet Mail Extensions (MIME), File Extention dan File Size pada setiap berkas yang diunggah tanpa mengecek isi kontennya sehingga berkas digital object yang corrupt atau yang disisipi selain gambar dapat diproses dalam fitur unggahan. Faktor keamanan sangatlah penting dikarenakan pada zaman yang semakin berkembang banyak sistem yang berupa digital. Ancaman serangan terhadap berbagai jenis sistem yang berbentuk digital atau server juga ikut berkembang, maka diperlukan sebuah penanganan terhadap ancaman atau serangan pada server dengan celah fitur unggahan. Penelitian ini bertujuan untuk meneliti teknik validasi PNG yang tidak hanya mengecek MIME, File Extention dan File Size tetapi memanfaatkan juga Metadata PNG dan nilai warna RGBA (Red, Green, Blue, Alpha) pada setiap pixel dengan Metode Pengembangan menggunakan Extreme Programming (XP). Hasil penelitian dengan sampel 10 gambar berisi metadata dan 10 gambar yang tidak berisi metadata dapat mengecek validasi secara khusus unggahan gambar pada PNG File dengan menyaring secara khusus sesuai keunikan dari file gambar yaitu mempunyai panjang dan lebar pada metadatanya yang harus tersedia (Width, Height, Bits, Mime) serta RGBA (Red, Green, Blue, Alpha) pada setiap piksel sesuai dengan panjang dan lebar sebuah gambar PNG.
Kata Kunci: *Gambar, Keamanan, Metadata, Unggah, Validasi*

Abstract

The use of websites today is growing rapidly, various types of applications, services from the private sector to the government, many use the website as a medium of interaction between various platforms. Usually when uploading files on a website, the system validates by filtering digital object file types on the server side (backend) or on the web page in a web browser in HTML or Javascript (frontend) using the Header or Multipurpose Internet Mail Extensions (MIME) filtering, File Extensions and File Sizes on each file uploaded without checking the contents of the contents so that digital files that are corrupt or inserted objects other than images can be processed in the upload feature. The safety factor is very important because in an age of growing numbers of digital systems. The threat of attacks on various types of systems in the form of digital or server also develops, it is necessary to handle a threat or attack on the server with upload feature loopholes. This study aims to examine PNG validation techniques that not only check MIME, File Extention and File Size but also utilize PNG Metadata and RGBA color values (Red, Green, Blue, Alpha) at each pixel with Development Methods using Extreme Programming (XP). The results of the study with a sample of 10 images containing metadata and 10 images that do not contain metadata can check the validation specifically upload images in PNG File by filtering specifically according to the uniqueness of the image file that has length and width in the metadata that must be available (Width, Height, Bits , Mime) and RGBA (Red, Green, Blue, Alpha) on each pixel according to the length and width of a PNG image.
KeyWords : *Image, Metadata, Security, Upload, Validation*

I. PENDAHULUAN

Website merupakan aplikasi yang dapat diakses oleh berbagai orang di dunia yang biasa disebut dengan pengguna dengan terhubung ke *internet*. Proses unggah berkas adalah salahsatu teknik yang biasanya dibutuhkan secara fungsional oleh aplikasi perusahaan untuk para pengguna [1], yang dapat digunakan untuk dokumen, gambar, unggah data, dan penyimpanan oleh klien [2]. Namun tanpa metode keamanan dan penyaringan yang tepat, pemilihan berkas dan proses validasi selama mengungah dapat memberikan resiko keamanan yang signifikan. *Survey* teknik keamanan aplikasi *web* [3] dengan mengkategorikan tiga sifat penting: integritas negara, validasi *input* dan logika kebenaran yang diperlukan untuk aplikasi keamanan bersama dengan lingkup masa depan penelitian. Argumen yang menyatakan harus ada kolaborasi antara *server* dan klien untuk meningkatkan keamanan dengan memberikan contoh mekanisme untuk mencapai keamanan *end-to-end* [4] seperti pada pemanfaatan teknologi *Web Push Notification* yang megirim pesan ke *web browser* dari *server* [5].

Beberapa penyebab kerentanan terkait dengan lapisan pada aplikasi berbasis *web* [6]. Penelitian ini juga memberikan ulasan tentang teknik, tahapan, pendekatan dan alat untuk mendeteksi kerentanan. Pentingnya *web server* bersama dengan ancaman yang ditimbulkan oleh peretas [7]. Penanggulangan terhadap ancaman *web server* juga dijelaskan. Tantangan dan masalah yang terjadi selama pengujian keamanan aplikasi *web*. Hal ini dilakukan untuk memberikan masukan kepada penguji dan manajer

sehubungan dengan proyek [8]. Penelitian ini mengambil masalah yang disorot oleh berbagai penulis berdasarkan OWASP top 10 kategorisasi [9].

Beberapa cara di mana fungsi unggahan *file* dapat dieksploitasi seperti tidak ada validasi yang dilakukan pada klien atau *server*, validasi yang diterapkan di sisi klien dapat dilewati menggunakan opsi pengembang, tidak ada validasi yang dilakukan untuk memeriksa konten *file* yang diunggah oleh pengguna akhir, tidak ada validasi yang dilakukan untuk memeriksa ukuran *file* yang diunggah oleh pengguna akhir, ketika validasi didasarkan hanya pada tipe konten, serangan dapat dilakukan dengan memanipulasi tipe konten *file* yang menentukan sifat data, diizinkan menggunakan lebih dari satu jenis ekstensi *file* dan beberapa kondisi dapat menggunakan ekstensi *file* terlarang bersama dengan ekstensi *file* yang tidak diizinkan oleh aplikasi [10]. Para penyerang dapat melakukan XSS yang disimpan menggunakan fitur unggah *file* [11].

Penelitian terdahulu yang meneliti tentang teknik validasi dengan memanfaatkan *Metadata* dan nilai *RGB (Red, Green, Blue)* pada JPEG File menggunakan *Image Processing* [12]. Pada umumnya, teknik validasi hanya menggunakan teknik penyaringan *Multipurpose Internet Mail Extensions (MIME)* atau *Header File*, *File Extension* dan *File Size*. Pada penelitian ini akan memanfaatkan metadata dan isi konten *RGBA (Red, Green, Blue, Alpha)* pada *PNG File* sebagai teknik validasi fitur File Upload pada *website*.

Informasi yang diungkapkan oleh *metadata* mungkin sangat penting bagi penyelidik dalam menemukan perubahan atau manipulasi. Investigasi juga bisa gagal ketika bukti tidak divalidasi, atau harus diperiksa dengan cermat selama investigasi. Faktor-faktor berbeda yang mungkin memengaruhi validitas bukti adalah penggunaan alat, sistem, atau kesalahan aplikasi yang tidak sesuai selama proses pengumpulan bukti, kegagalan melaporkan bukti pembuktian, kesalahan penyajian bukti, kegagalan untuk mengidentifikasi bukti terkait, dan pemalsuan bukti yang mengarah ke penyesatan [13]. *Metadata* dapat menghasilkan bukti yang tidak selalu mudah terlihat dan penyidik harus memastikan pengumpulan data tersebut dan validasinya untuk dipresentasikan di pengadilan. Gambar juga dapat dideteksi keaslian gambar digital dengan memverifikasi menggunakan alat pendeteksi [14].

A. Berkas Gambar

Gambar didefinisikan sebagai susunan angka dan angka tersebut biasanya berarti intensitas cahaya yang berbeda di berbagai bagian gambar [15]. Deskripsi numerik mengambil bentuk kisi di mana masing-masing poin diberi nama ’piksel’. *Pixel* ditampilkan secara *horizontal*, baris demi baris. Dalam skema warna, jumlah *bit* dikenal sebagai kedalaman *bit* dan ini pada dasarnya mengacu pada jumlah *bit* yang ditetapkan untuk setiap piksel [16].

Kedalaman *bit* terkecil dalam skema warna adalah 8, yaitu, 8 *bit* digunakan untuk mewakili warna setiap piksel. Baik gambar skala Monokrom dan skala abu-abu biasanya menggunakan 8 *bit* untuk setiap piksel dan bit tersebut mampu menampilkan hingga 256 warna berbeda atau warna abu-abu. T. Sari, I. Riadi, dan A. Fadlil [17] menjelaskan teknik forensik yang dapat mendeteksi berkas gambar yang direkayasa menggunakan *Error Level Analysis* (ELA) dan teknik ELA (*Error Level Analysis*) pada Forensicallybeta dapat digunakan untuk mendeteksi keaslian suatu citra [18], citra juga digunakan untuk mendeteksi tekstur dengan memanfaatkan *Gray Level Coocurrence Matrix* (GLCM) dengan klasifikasi jarak menggunakan *Euclidean* [19].

Warna pada *image file* disimpan dalam format *file 8-bit*, *16-bit*, *24-bit* atau *32-bit* dan untuk memanfaatkan model warna *Red, Green, Blue (RGB)* serta modelwarna *Red, Green, Blue, Alpha (RGBA)* pada *PNG File*. Hampir semua variasi warna untuk piksel gambar *24-bit* berasal dari tiga istilah warna dasar: merah, hijau, dan biru, dan masing-masing warna ini diwakili oleh 8 *bit* [15]. Dengan demikian, dalam piksel apa pun, jumlah warna merah, hijau, dan biru yang berbeda dapat mencapai 256 yang menambahkan hingga lebih dari 16 juta kombinasi yang akhirnya menghasilkan lebih dari 16 juta warna. Format gambar yang paling menonjol, secara eksklusif di *internet*, adalah format pertukaran grafik (GIF), format gabungan kelompok ahli fotografi (JPEG), dan sedikit banyak, format grafik jaringan portabel (PNG) [20].

B. Image Processing

GD adalah pustaka kode sumber terbuka untuk pembuatan gambar secara dinamis oleh pemrogram. *GD* ditulis dalam bahasa C, dan ”pembungkus” tersedia untuk *Perl*, *PHP*, dan bahasa lainnya. *GD* menciptakan gambar PNG, JPEG, GIF, WebP, XPM, BMP, di antara format lainnya. *GD* umumnya digunakan untuk menghasilkan grafik, gambar, thumbnail, dan sebagian besar hal lainnya, dengan cepat. Meskipun tidak terbatas untuk digunakan di *web*, aplikasi *GD* yang paling umum melibatkan pengembangan *web*. *Library* ini awalnya dikembangkan oleh Thomas Boutell dan sekarang dikelola oleh Pierre Joye di bawah payung PHP.net [21].

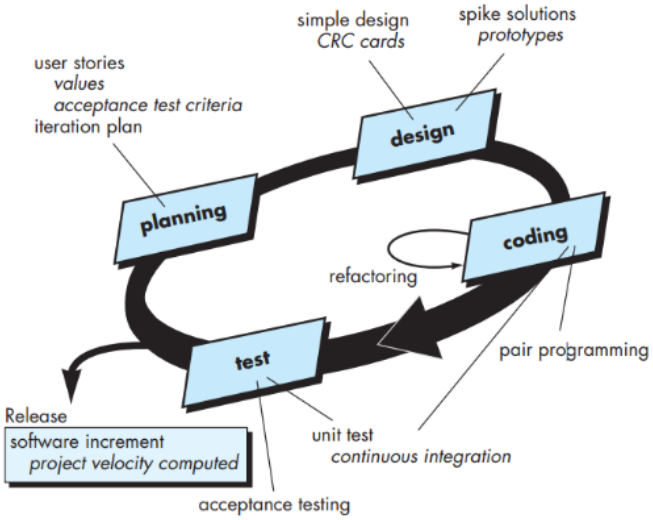
C. Metadata

Metadata dapat memberi penyelidik banyak informasi tentang *file* yang sedang diselidiki. Lebih lanjut, penyelidik forensik dapat menggunakan metadata untuk memperoleh informasi, misalnya, pembuat *file*, tanggal dan waktu pembuatan, berapa kali *file* telah dimodifikasi, termasuk ketika modifikasi terjadi. *Metadata* terkandung dalam *file* media seperti format pertukaran grafik (GIF), JPEG, dan dalam musik: MP3 dan AAC dan format *file* gambar yang ditandai. *Metadata* dapat berguna dalam penyelesaian sengketa hukum, karena dapat digunakan sebagai bukti untuk membuktikan atau membantah bukti lain yang

diajukan dalam kasus pengadilan. Selain itu, metadata dapat disimpan di berbagai situs di dalam *file*. Penyelidik dapat mengumpulkan informasi dari metadata ini yang berkorelasi dengan kepemilikan dan pemilik potensial [13]. *Metadata* merupakan informasi yang dapat digunakan untuk mengidentifikasi sebuah data dasar dari sebuah *object digital* [17].

II. METODE

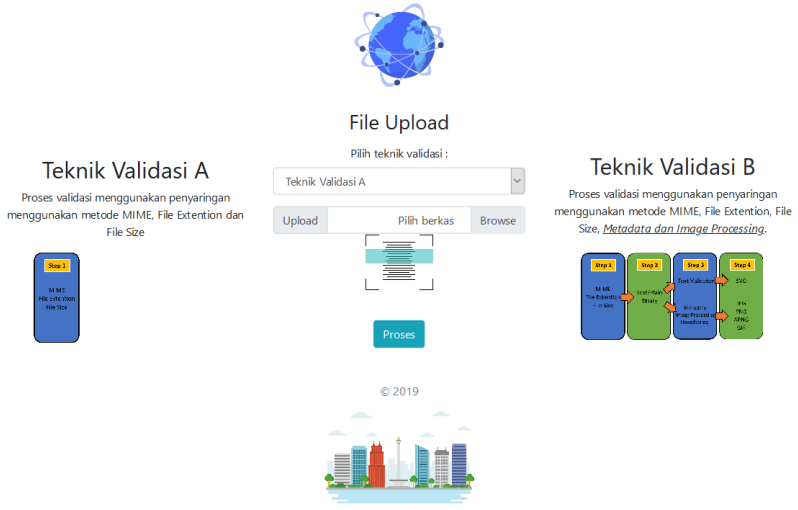
Extreme Programming (XP) merupakan metode yang digunakan dalam membangun aplikasi. *Extreme Programming* (XP) biasanya menggunakan pendekatan pemrograman berorientasi objek (*object oriented programming*) dan sasaran dari metode ini merupakan tim dalam skala kecil hingga menengah juga sesuai jika dihadapkan dengan kebutuhan atau *requirement* yang terjadi perubahan–perubahan dari kebutuhan-kebutuhan yang dinamis atau sangat cepat maupun yang tidak jelas sejak awal [22].



Gambar 1: Alur Metode Extreme Programming (XP)

Gambar 1 merupakan tahapan alur pembangunan aplikasi *web* menggunakan metode *Extreme Programming* terdiri dari Perencanaan (*Planning*) yang berisi kumpulan kebutuhan aktifitas dan berbagai proses bisnis yang akan dibuat, Perancangan (*Design*) yang berisi kumpulan dari *Unified Modelling Language* (UML) yang diperlukan, Pengkodean (*Coding*) yang berisi kumpulan kode yang dibuat menggunakan PHP, Pengujian (*Testing*) yang berisi pengujian menggunakan *Black Box Testing* dan Peningkatan Perangkat Lunak (*Software Increment*) yang berisi informasi yang diperlukan untuk pengembangan berikutnya.

III. HASIL



Gambar 2: Tampilan Impementasi Algoritma Teknik Validasi File *Upload* menggunakan *Metadata* dan *Image Processing*

Gambar 2 merupakan implementasi *website* pada fitur Validasi *File Upload* menggunakan *Metadata* dan *Image Processing GD Graphic Library* pada PHP kemudian diproses dengan penyaringan menggunakan *Metadata* sesuai dengan karakteristik *digital object PNG File* yang diunggah dan dibantu oleh *Image Processing* untuk mengambil warna setiap *pixel* pada *digital object* berjenis *image file* dengan karakteristik *RGB (Red, Green, Blue)* pada *JPEG File* [12] dan pada penelitian ini memakai karakteristik *PNG File* seperti *RGBA (Red, Green, Blue, Alpha)* dan bernilai antara 0 sampai 255 setiap *channel* warna pada semua *pixels* yang terdapat pada *digital object PNG File*.

IV. Pembahasan

Penelitian ini menggunakan *Extreme Programming* sebagai metode pengembangan implementasi Teknik Validasi *File Upload* menggunakan *Metadata* dan memanfaatkan *Image Processing GD Graphic Library* dengan beberapa tahapan sebagai berikut :

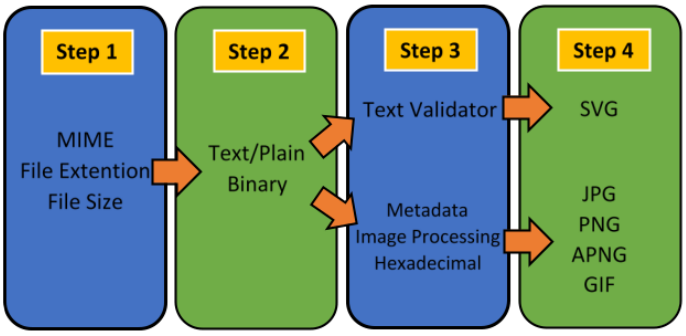
A. Perencanaan (Planning)

Tahap ini berisi perencanaan yang akan dilakukan menggunakan sampel yang ada pada Tabel I dengan 10 sampel berisi *metadata* (*width, height, bits* dan *MIME*) dan 10 sampel tersebut dihapus metadatanya menggunakan *Remove Properties* pada sistem operasi *Windows* dibantu dengan aplikasi *Hex Editor* untuk menghilangkan nilai *metadata* (*width, height* dan *bits*).

Tabel I: Sampel Gambar yang akan digunakan

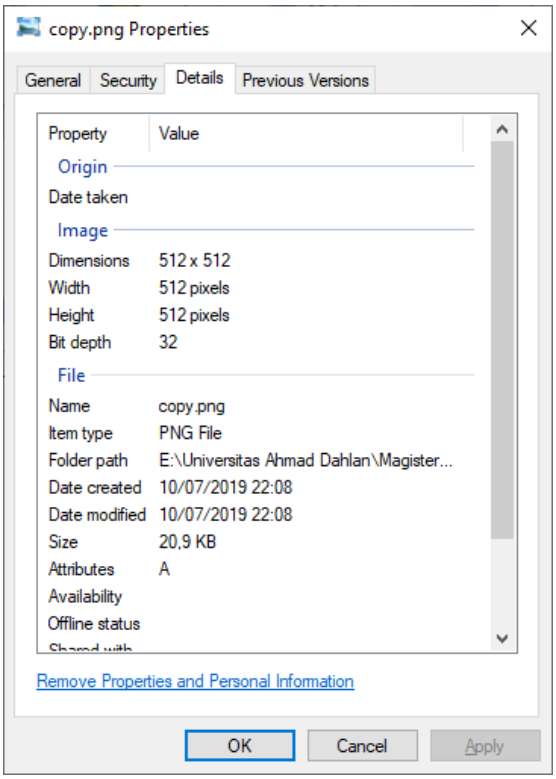
No.	Gambar	Tinggi (Height)	Panjang (Width)	Bits	Warna	Tipe file	Ukuran file
1		512 pixels	512 pixels	32	RGBA	PNG File	28,2 KB
2		512 pixels	512 pixels	32	RGBA	PNG File	29,5 KB
3		512 pixels	512 pixels	32	RGBA	PNG File	20,9 KB
4		512 pixels	512 pixels	32	RGBA	PNG File	20,9 KB
5		512 pixels	512 pixels	32	RGBA	PNG File	58,1 KB
6		512 pixels	512 pixels	32	RGBA	PNG File	24,5 KB
7		512 pixels	512 pixels	32	RGBA	PNG File	39,8 KB
8		512 pixels	512 pixels	32	RGBA	PNG File	55,9 KB
9		512 pixels	512 pixels	32	RGBA	PNG File	57 KB
10		512 pixels	512 pixels	32	RGBA	PNG File	15,5 KB

Tabel I berisi 10 gambar dengan parameter yang sama dengan tinggi 512 piksel dan lebar 512 piksel dengan nilai Bits 32 dan memiliki warna RGBA atau terdapat warna *channel alpha*. Kumpulan gambar pada Tabel 1 akan diproses ke dalam tahapan-tahapan seperti pada Gambar 2.



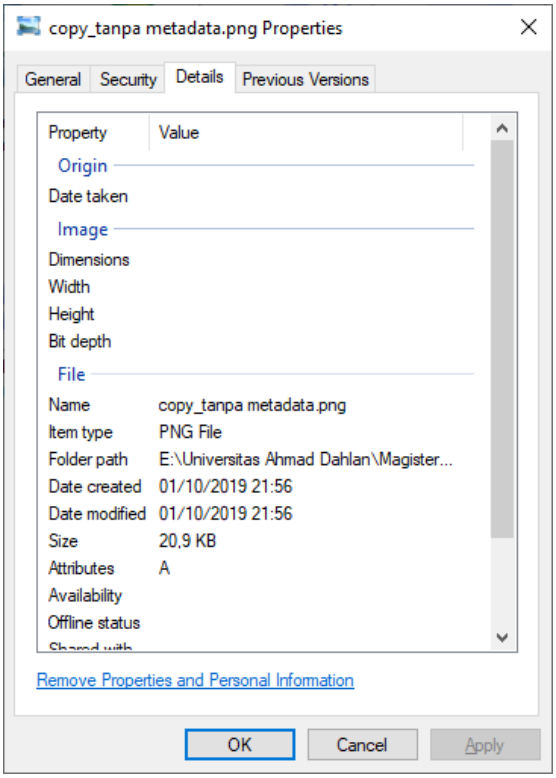
Gambar 3: Tahapan Validasi menggunakan *Metadata* dan *Image Processing*

Gambar 3 merupakan tahapan-tahapan dalam melakukan proses validasi *digital object* yang biasanya hanya menggunakan *Step* 1 saja sedangkan dalam penelitian ini menggunakan 4 tahapan. Setelah melewati *Step* 1 akan di pisahkan berdasarkan 2 kategori *text/plain* dan *binary*. Pada *Step* 2 akan dipisahkan menjadi dua kategori *text/plain* dan *binary*, *step* 3 di proses berdasarkan hasil dari *step* 2 untuk *MIME* bertipe *text/plain* menggunakan metode *Text Validator* sedangkan untuk *MIME* bertipe *binary* menggunakan metode pemanfaatan *Metadata*, *Image Processing* sebagai media pengolahan citra dan *Hexadecimal* untuk mengambil nilai *binary*. *Object digital* yang berjenis *text/plain* akan diproses menggunakan *Text Validator* sesuai dengan jenis *MIME* dan *File Extension* pada *Step* 1 sedangkan *Binary* akan diproses menggunakan *Metadata*, *Image Processing* dan *Hexadecimal* setelah itu akan menghasilkan klasifikasi jenis berkas yang diunggah.



Gambar 4: Properties Berkas PNG berisi *Metadata*

Gambar 4 berisi informasi *metadata* yang utuh dan Gambar 5 tidak berisi informasi *metadata* dan hanya berisi informasi *MIME* saja atau *Start Of Image* seperti pada awalan heksadesimal pada Tabel II yaitu marker **89 50 4E 47 0D 0A**. Gambar 5 berisi metadata kategori *File* memiliki *tag Name* bernilai copy.png, *tag Item type* bernilai PNG File, *tag Folder path* bernilai lokasi berkas, *tag Date created* bernilai 10/07/2019 22:08, *tag Date modified* bernilai 10/07/2019 22:08, *tag Size* bernilai 20,9 KB dan *tag Attributes* bernilai A.



Gambar 5: Properties Berkas PNG yang tidak berisi *Metadata*

Tabel II: Heksadesimal Metadata berisi *MIME* (*Header File*) pada Berkas Gambar PNG

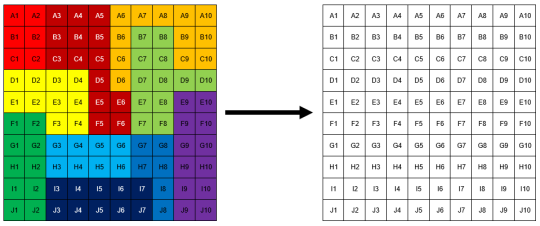
00000000h:	89 50 4E 47 0D 0A	1A 0A 00 00 00 0D 49 48 44 52 ;%ₒPNG.....IHDR
00000010h:	00 00 02 00 00 00 02 00 08 06 00 00 00 F4 78 D4	;.....ôxÔ
00000020h:	FA 00 00 00 04 73 42 49 54 08 08 08 08 7C 08 64	;ú....sBIT....|.d
00000030h:	88 00 00 00 09 70 48 59 73 00 00 0B 13 00 00 0B	ˆ....pHYs.....

Tabel II berisi data heksadesimal *MIME* atau jenis *File Format* pada Berkas yang bisa disebut pula dengan segments beserta *fields marker* tersebut merupakan awalan nilai pada heksadesimal atau *Start Of Image* bernilai **89 50 4E 47 0D 0A 1A 0A** diakhiri sampai panjang **IHDR** bernilai *00 00 00 0D* dan **IHDR** bernilai *49 48 44 52* atau informasi *metadata* yang berisi nilai (*Width, Height dan Bits*) seperti pada Tabel III.

Tabel III: Heksadesimal *Metadata* berisi (*Width, Height* dan *Bits*) pada Berkas Gambar PNG

00000000h:	89 50 4E 47 0D 0A 1A 0A	00 00 00 0D	49 48 44 52 ;%ₒPNG.....IHDR
00000010h:	00 00 02 00 00 00 02 00 08 06 00 00 00 F4 78 D4		;.....ôxÔ
00000020h:	FA 00 00 00 04 73 42 49 54 08 08 08 08 7C 08 64		;ú....sBIT....|.d
00000030h:	88 00 00 00 09 70 48 59 73 00 00 0B 13 00 00 0B		ˆ....pHYs.....

Metadata yang ada dalam berkas PNG, kemudian di hilangkan dan hanya menyisakan *tag* yang wajib ada, salahsatunya ukuran dimensi gambar. Kemudian diambil warna gambar sesuai dengan dimensi yang ada pada metadata dimasukan kedalam canvas gambar baru secara rekursif seperti pada Gambar 6.



Gambar 6: Ilustrasi Pengambilan Warna pada Gambar ke *Canvas* Baru menggunakan *GD Graphic Library*

Gambar 6 merupakan ilustrasi pengambilan warna pada berkas gambar yang diunggah dan membuat *canvas* sesuai dengan ukuran panjang dan lebar *pixel* pada gambar yang diunggah, kemudian memasukan nilai *RGBA (Red, Green, Blue, Alpha)* pada setiap *pixel* yang diambil kedalam *canvas* yang telah dibuat secara rekursif sehingga informasi selain warna tidak disimpan di *server* menggunakan *Image Processing* seperti *GD Graphic Library* [12].

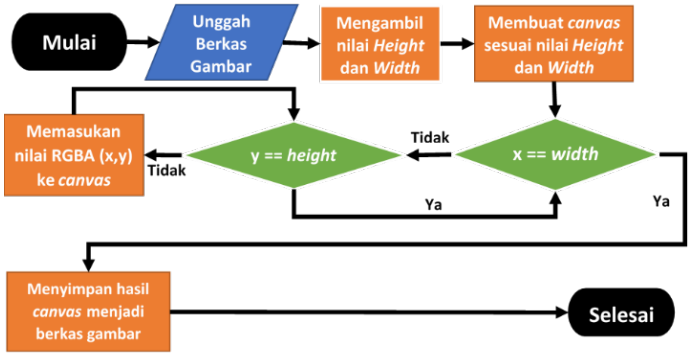
Gambar 7: Ilustrasi Aliran Berkas Gambar JPEG dideteksi berstatus Corrupted atau OK

Gambar 6 merupakan ilustrasi aliran berkas gambar JPEG yang dimasukan kedalam proses analisa pada Gambar 7 dengan menghasilkan keterangan *Corrupted* atau *OK*, jika *OK* maka berkas gambar akan diproses dengan mengambil warna setiap *pixel* sesuai dengan dimensi gambar yang telah diketahui pada *metadata* dengan implementasi menggunakan algoritma pada algoritma 1.

Algoritma 1:
Algoritma Pengecekan Dan Pemindahan Warna Gambar Pada Canvas Gambar Baru Menggunakan GD Graphic liblary

```
$input = "files/".$_GET['file'].".png";
$output = "hasil/".$_GET['file'].".png";
list($width, $height) =  getimagesize($input);
$gd = imagecreatetruecolor($width, $height);
$png = imagecreatefrompng($input);
for ($y = 0; $y < $height; $y++) {
for ($x = 0; $x < $width; $x++) {
    $rgba = imagecolorat($png, $x, $y);
    imagesetpixel($gd, $x,$y, $rgba);
}
}
imageresolution($gd, 72);
imagepng($gd, "hasil/".$_GET['file'].".png",100);
imagedestroy($gd);
```

B. Perancangan (Design)



Gambar 8: Alur Pengambilan Warna pada Gambar ke Canvas Baru menggunakan GD Graphic Library

Gambar 8 merupakan *Flowchart* Pengambilan warna pada gambar ke *canvas* baru menggunakan *GD Graphic Library* pada PHP dengan mengambil informasi panjang dan lebar *pixel* pada gambar yang diunggah kemudian membuat canvas baru dengan ukuran panjang dan lebar yang telah diambil dari metadata, alur ini akan diimplementasikan ke dalam tahap Pengkodean (*Coding*) dengan menggunakana pemrograman PHP sebagai *backend* dan HTML, CSS & JS sebagai *frontend*.

C. Pengkodean (Coding)

Algoritma 2:
Algoritma Mengecek Ukuran Berkas Gambar mengambil nilai pada Metadata

```
<?php
if(getimagesize($image) === false){
    echo "File is corrupted";
  }else{
    echo "File is ok";
  }
?>
```

Algoritma 2 berisi algoritma pemrograman PHP yang berfungsi untuk mengambil nilai dari ukuran panjang dan lebar berkas gambar yang telah diunggah oleh pengguna, jika berisi maka proses penyimpanan akan dilakukan sedangkan jika tidak berisi nilai maka akan menampilkan pesan kesalahan. Contoh informasi yang ada pada *getimagesize()* adalah sebagai berikut :

```
Array ( [0] => 512 [1] => 512 [2] => 3 [3] => width="512"
height="512" [bits] => 32 [mime] => image/png )
```

Fungsi *getimagesize()* menghasilkan 6 elemen *array* seperti elemen [0] bernilai 512, elemen [1] bernilai 512, elemen [2] bernilai 3, elemen [3] bernilai *width=”512” height=”512”*, elemen [*bits*] bernilai 32, elemen *mime* bernilan *image/png*.

D. Pengujian (Testing)

Pengujian pada penelitian ini menggunakan 10 sampel yang memiliki metadata dan 10 sampel yang tidak memiliki metadata kemudian di unggah sesuai dengan skenario uji yang telah ditentukan pada Tabel IV yang berisi hasil deteksi hanya menggunakan *MIME* semuanya lolos valdasi sedangkan pada Tabel V hasil deteksi menggunakan *MIME* dan *metadata* hanya berkas gambar yang dapat dibuka yang lolos validasi sesuai skenario pada Tabel VI.

Tabel IV: Hasil deteksi hanya menggunakan MIME Check

	Corrupt Detected	***Not Corrupt Detected***	**JUMLAH**
Corrupted File	0	10	10
Not Corrupted FIle	0	10	10
JUMLAH	0	10	

Tabel IV merupakan tabel yang berisi hasil deteksi dengan menggunakan metode penyaringan hanya menggunakan *MIME Check* sedangkan Tabel V menggunakan metode penyaringan menggunakan *Metadata* (*Width, Height, Bits, Mime*).

Tabel V: Hasil deteksi menggunakan MIME dan Metadata

	Corrupt Detected	***Not Corrupt Detected***	**JUMLAH**
Corrupted File	10	0	10
Not Corrupted FIle	0	10	10
JUMLAH	0	10	

Pengujian selanjutnya menggunakan pengujian skenario uji pada Algoritma yang diterapkan dengan alur skenario uji, kemudian dibuat kesimpulan berdasarkan hasil yang diharapkan seperti pada Tabel VI.

Tabel VI: Skenario Uji pada Algoritma yang diterapkan

Skenario Uji	**Hasil yang diharapkan**	**Kesimpulan**
Upload gambar normal	Sukses Mengunggah	[✓] Berhasil [ ] Tidak Berhasil
Upload gambar hapus beberapa metadata kecuali (*Width, Height, Bits, Mime*)	Sukses Mengunggah	[✓] Berhasil [ ] Tidak Berhasil
Upload gambar tanpa metadata	Gagal Mengunggah	[✓] Berhasil [ ] Tidak Berhasil

Pengujian Skenario Uji dengan hasil pada Tabel VI, semua skenario uji telah melakukan ujicoba dengan membandingkan ekspetasi dan kesimpulan dari hasil yang diharapkan telah berhasil.

E. Peningkatan Perangkat Lunak (Software Increment)

Penelitian ini dapat diterapkan dalam Algoritma *Upload File PNG* setelah validasi lainnya misalnya seperti validasi *MIME Type* dan Ukuran *File* untuk mengecek validasi khusus *upload file JPEG* dengan menyaring secara khusus sesuai keunikan dari *file* gambar yaitu mempunyai panjang dan lebar pada metadatanya yang harus tersedia (*Width, Height, Bits, MIME, RGBA*) jika tidak sesuai *Metadata* maka berkas PNG akan dianggap *corrupt*.

V. Simpulan dan Saran

Penelitian ini menghasilkan aplikasi yang dapat memberikan keamanan dalam proses pengunggahan berkas pada aplikasi berbasis *web* khususnya berjenis *PNG File*. Metode *Extreme Programming* (XP) digunakan untuk mengembangkan atau membangun perangkat lunak karena banyaknya perubahan secara cepat atau dinamis dalam proses pembuatan. Pembuatan standarisasi berupa petunjuk teknis penggunaan digunakan agar aplikasi dapat berjalan sesuai dengan fungsinya, teknik validasi keamanan penanganan *file upload* berjenis *PNG File* menggunakan *Metadata* dan *MIME* lebih baik dibanding hanya menggunakan *MIME*, *File Extention*, *File Size* saja yang dapat menyaring berkas gambar PNG hingga dapat memilah berkas yang berstatus corrupt atau tidaknya dengan dapat mengecek validasi secara khusus unggahan gambar pada *PNG File* dengan menyaring secara khusus sesuai keunikan dari *file* gambar yaitu mempunyai panjang dan lebar pada metadatanya yang harus tersedia (*Width, Height, Bits, Mime*) serta *RGBA (Red, Green, Blue, Alpha)* pada setiap piksel sesuai dengan panjang dan lebar sebuah gambar PNG.

Pustaka

[1] H. Chen, L. J. Zhang, B. Hu, S. Z. Long, and L. H. Luo, "On Developing and Deploying Large-File Upload Services of Personal Cloud Storage," Proc. - 2015 IEEE Int. Conf. Serv. Comput. SCC 2015, pp. 371–378, 2015.
[2] R. Umar, A. Hadi, P. Widiandana, F. Anwar, M. Jundullah, and A. Ikrom, "Perancangan Database Point of Sales Apotek Dengan Menerapkan Model Data Relasional," Query J. Sist. Inf., vol. 03, no. 02, pp. 33–41, 2019.
[3] X. Li and Y. Xue, "A survey on web application security," Nashville, TN USA, 2011.
[4] U. Erlingsson, B. Livshits, and Y. Xie, "End-to-end Web Application Security," Proc. 11th Work. Hot Top. Oper. Syst. (HotOS'07), San Diego, CA, pp. 2–7, 2007.
[5] A. Rahmatulloh, A. N. Rachman, and F. Anwar, "Implementasi Web Push Notification pada Sistem Informasi Manajemen Arsip Menggunakan PUSHJS," J. Teknol. Inf. dan Ilmu Komput., vol. 6, no. 3, pp. 327–334, 2019.
[6] R. Sajjad, H. Mamoona, G. Zartasha, A. Ansar, and J. Hasan, "Systematic Review of Web Application Security Vulnerabilities Detection Methods," J. Comput. Commun., vol. 3, pp. 28–40, 2015.
[7] S. B. Almi, "Web Server Security and Survey on Web Application Security," Int. J. Recent Innov. Trends Comput. Commun., vol. 2, no. 1, pp. 114–119, 2014.
[8] A. Jaiswal, G. Raj, and D. Singh, "Security Testing of Web Applications: Issues and Challenges," Int. J. Comput. Appl., vol. 88, no. 3, pp. 26–32, 2014.
[9] OWASP, "OWASP Top 10 - The Ten Most Critical Web Application Security Risks", OWASP Top 10, 2013. [Online]. Available: https://www.owasp.org/images/f/f8/OWASP_Top_10_-_2013.pdf. [Accessed: 20-Jun-2019].
[10] K. Pooj and S. Patil, "Understanding File Upload Security for Web Applications", Int. J. Eng. Trends Technol., vol. 42, no. 7, pp. 342–347, 2016.
[11] Y. W, I. Riadi, and A. Yudhana, "Analisis Deteksi Vulnerability pada Webserver Open Jurnal System menggunakan OWASP Scanner", JURTI, vol. 2, no. 1, pp. 1–8, 2018.
[12] F. Anwar, A. Fadlil, and I. Riadi, "ANALISA KEAMANAN IMAGE JPEG FILE UPLOAD MENGGUNAKAN METADATA DAN GD GRAPHIC LIBRARY PADA APLIKASI BERBASIS WEB," in Seminar Nasional Teknologi Fakultas Teknik Universitas Krisnadwipayana, 2019, pp. 479–487.
[13] F. Alanazi and A. Jones, "The Value of Metadata in Digital Forensics," Proc. - 2015 Eur. Intell. Secur. Informatics Conf. EISIC 2015, vol. 8, no. 2011, p. 182, 2016.
[14] I. Riadi, A. Fadlil, and T. Sari, "Image Forensic for detecting Splicing Image with Distance Function," Int. J. Comput. Appl., vol. 169, no. 5, pp. 6–10, 2017.
[15] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," Computer (Long. Beach. Calif)., vol. 31, no. 2, pp. 26–34, 1998.
[16] R. Poornima and R. J. Iswarya, "An Overview Of Image Steganography," Int. J. Comput. Sci. Eng. Surv., vol. 4, no. 1, pp. 23–31, 2013.
[17] T. Sari, I. Riadi, and A. Fadlil, "Forensik Citra untuk Deteksi Rekayasa File Menggunakan Error Level Analysis," Annu. Res. Semin. 2016, vol. 2, no. 1, pp. 133–138, 2016.
[18] W. Y. Sulistyo, I. Riadi, and A. Yudhana, "Analisis Deteksi Keaslian Citra Menggunakan Teknik," vol. 2018, no. November, pp. 154–159, 2018.
[19] S. Saifudin and A. Fadlil, "Sistem Identifikasi Citra Kayu Berdasarkan Tekstur Menggunakan Gray Level Coocurrence Matrix (GLCM) Dengan Klasifikasi Jarak Euclidean," Sinergi, vol. 19, no. 3, p. 181, 2015.
[20] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," Signal Processing, vol. 90, no. 3, pp. 727–752, 2010.
[21] T. Boutell and P. Joye, "About." [Online]. Available: https://libgd.github.io/pages/about.html. [Accessed: 14-May-2019].
[22] R. S. Pressman and B. R. Maxim, Software Engineering: a practitioner's approach, 8th ed. New York: McGraw-Hill Education, 2014.