
Versioning Catcher Software (VCS) untuk mendeteksi Sensitive Information File Versioning pada Linux Server

Rhenaldo Delfi Nugraha ¹⁾, Dian Widiyanto Chandra ²⁾

^{1,2)} Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana

Jl. Dr. O. Notohamidjojo 1-10, Salatiga 50711, Indonesia

Email : ¹⁾672017011@student.uksw.edu, ²⁾dian.chandra@uksw.edu

Riwayat artikel:

Received: 27-05-2021

Revised: 17-07-2021

Accepted: 18-07-2021

Abstract

Website is a very popular medium for finding information. Building a website requires a lot of component files which will be stored on the server. To minimize website hacking, security is needed from the system side and the user side. Based on OWASP's TOP 10, one of the vulnerabilities that usually appears is the leakage of sensitive information due to the versioning file residing on the server. In order to minimize the leakage of sensitive information from these vulnerabilities, the authors created a Versioning Catcher Software (VCS) program which functions to detect the presence of these versioning files. The purpose of this research is to protect and prevent sensitive information from being leaked on the server. The results of the discussion of this research can be used to minimize the occurrence of server hacks as a result of the versioning file containing sensitive information that is not handled immediately.

Keywords: *server, OWASP, sensitive information, file versioning, information security*

Abstrak

Website merupakan media yang sangat populer untuk menemukan informasi. Membangun sebuah website diperlukan banyak sekali komponen file yang nantinya disimpan di dalam server. Untuk meminimalisir terjadinya peretasan website, diperlukan keamanan dari sisi sistem dan sisi pengguna. Berdasarkan TOP 10 OWASP, salah satu kerentanan yang biasa muncul adalah bocornya informasi sensitif akibat dari file versioning yang berada dalam server. Demi meminimalisir terjadinya kebocoran informasi sensitif dari kerentanan tersebut, maka penulis membuat program Versioning Catcher Software (VCS) yang berfungsi untuk mendeteksi keberadaan file versioning tersebut. Tujuan dilakukannya penelitian ini adalah untuk melindungi dan mencegah bocornya informasi sensitif di server. Hasil pembahasan dari penelitian ini dapat digunakan untuk meminimalisir terjadinya peretasan server akibat dari file versioning berisi informasi sensitif yang tidak segera ditangani.

Kata kunci: *server, OWASP, informasi sensitif, file versioning, keamanan informasi*

Pendahuluan

Server merupakan sebuah sistem komputer yang berfungsi untuk menerima dan merespon suatu permintaan dalam suatu jaringan yang dapat berbentuk *software* maupun *hardware*. Dalam dunia internet khususnya *web server* berguna sebagai layanan untuk menangani permintaan (*request*) dari *client* seperti *e-mail*, DNS, FTP, Proxy dan sebagainya. Selain hal tersebut, server juga dapat digunakan untuk menyimpan data digital seperti HTML, CSS stylesheets, PHP, ASP, Javascript, dokumen, gambar, video, audio dan *file* digital lainnya.

Dalam sebuah *website* yang dinamis, pasti akan membutuhkan *file-file* bahasa pemrograman dalam sisi server (*server-side programming language*) seperti PHP, C#, JSP, Python, Ruby dan lain sebagainya yang disimpan di dalam server dengan tujuan supaya *website* dapat berjalan dengan semestinya. *File* seperti ini merupakan informasi sensitif, sehingga perlu adanya perlindungan agar *file* tersebut tidak terekspos ke publik. Akan tetapi, permasalahan muncul ketika *file* yang mengandung informasi sensitif itu bocor dan membuatnya dapat dijangkau oleh orang-orang yang tidak memiliki wewenang, hal ini mengakibatkan server maupun aplikasi web menjadi rentan untuk terkena serangan siber di internet.

Demi membantu untuk melindungi *website* dari serangan siber maka dibuatlah OWASP (*Open Web Application Security Project*). OWASP merupakan organisasi nirlaba yang memiliki banyak sukarelawan dari berbagai penjuru dunia dengan tujuan untuk meningkatkan keamanan perangkat lunak. Saat ini, banyak sekali faktor yang membuat informasi sensitif terekspos dan mengakibatkan peretasan sebuah *website*, berdasarkan yang tertulis dalam OWASP *Web Security Testing Guide* [1], dijelaskan bahwa ketika *file* yang mengandung informasi sensitif ini terekspos ke publik dapat menimbulkan ancaman seperti orang yang tidak bertanggung jawab dapat mengakses database server, mengakses aplikasi web, melihat konfigurasi *file*, *file* log, melakukan analisis terhadap logika kode program, dan pencurian atau perubahan data penting lainnya yang tersimpan dalam internal server. Tentunya ini bertentangan dengan tiga prinsip keamanan informasi, yaitu *Confidentiality*, *Integrity*, *Availability*.

Salah satu penyebab yang akan menjadi bahasan dalam penelitian ini adalah *file versioning*. *File versioning* merupakan cara untuk menyimpan salinan baru dari sebuah *file* yang akan dimodifikasi, *file versioning* dapat dilakukan secara manual oleh manusia itu sendiri maupun secara otomatis dari sebuah program seperti melakukan perubahan pada nama *file* yang sedang dimodifikasi. Ketika *file* yang mengandung informasi sensitif seperti *source code* dilakukan *versioning* namun tidak ditangani

dengan baik seperti menghapus *file source code versioning* tersebut dan pada akhirnya terekspos ke publik, maka *file* tersebut akan dapat dibaca dari sisi *client* dikarenakan *file* yang telah dilakukan *versioning* tidak akan diolah di sisi server melainkan akan dianggap sebagai *file text* biasa, ini akan menyebabkan alur aplikasi dapat terbaca, memungkinkan dilakukan analisis terhadap *source code* untuk mencari kerentanan dan skenario berbahaya lainnya sehingga ini akan membahayakan server dan aplikasi web itu sendiri.

Seperti pada standar OWASP [1], dalam penelitian ini penulis mengambil contoh kasus bocornya informasi sensitif akibat dari kebiasaan seorang *system administrator* yang menambahkan ekstensi-ekstensi umum di akhir nama *file*, seperti .bak, .old, .original, .backup, .save yang mengakibatkan terjadinya *file versioning*.

Dari permasalahan inilah penulis mencoba untuk mencari solusi dengan membuat sebuah program yang akan dinamai *Versioning Catcher Software (VCS)* menggunakan bahasa pemrograman Python3 yang memiliki fungsi untuk melakukan pendeteksian terhadap *file source code* yang telah dilakukan *versioning*. Teknik yang digunakan agar program ini bekerja adalah dengan melakukan perulangan *request* ke server, tiap *request* akan mengandung ekstensi di akhir nama *file* yang telah disiapkan dalam suatu *wordlist*, server akan menanggapi permintaan tersebut dan memberikan respon berupa kode status HTTP ke setiap *request*, kemudian hasil deteksi akan dilaporkan dalam bentuk *file* teks. Selain itu program ini dapat dijalankan dari sisi eksternal ketika seseorang tidak memiliki akses terhadap internal server maupun sebaliknya.

Kajian Pustaka

Dalam suatu penelitian diperlukan referensi penelitian-penelitian terdahulu yang berkaitan dengan penelitian tersebut, penelitian sebelumnya berguna untuk menunjang penelitian ini dibuat. Berikut adalah penelitian terdahulu sebagai referensi penelitian ini dibuat:

Pada penelitian sebelumnya oleh Akbar & Neforawati (2017) [2], mengutarakan kendala yang biasa dialami oleh tim ketika sedang mengembangkan suatu aplikasi *website* adalah ketika menggabungkan *file source code* yang telah dilakukan perubahan oleh setiap anggota tim akan sering menimbulkan error karena terdapat perubahan *file source code* yang sama. Untuk mengurangi terjadinya kesalahan ini, maka seorang developer diharuskan menggunakan *tool* yang berfungsi untuk mengelola, memonitor dan mengatur *source code* dengan teknologi *versioning*, dalam penelitian ini menggunakan Git dan Phinx.

Mider, Garlicki dan Mincewicz (2019) [3], menjelaskan bagaimana seseorang dapat mengoleksi data penting dari internet dengan menggunakan mesin pencari seperti Google. Dengan membuat suatu *query* seseorang akan bisa mendapatkan informasi secara spesifik, seringkali hal ini digunakan untuk mendapatkan informasi sensitif yang ada di internet, seperti menemukan data konfigurasi untuk mengakses server, kredensial *username & password*, *personally identifiable information* (PII) dengan memanfaatkan OSINT.

Hassan, Bhuiyan, Biswas (2016) [4], menjelaskan tentang betapa bahayanya ketika orang yang tidak memiliki wewenang dapat mengakses *file* berisi informasi sensitif, dengan kasus penelitian memanfaatkan kerentanan *Local File Disclosure* (LFD) pada aplikasi *website* edukasi di Bangladesh, yang memungkinkan seorang penyerang dapat mendapatkan *file* penting berisi informasi sensitif dari server seperti informasi nama *database*, *username*, *password*, *port* yang dapat digunakan untuk menyerang *web server* secara spesifik. Ketika *file* bermuatan informasi sensitif tersebut terekspos, maka dapat hal ini akan membahayakan server itu sendiri.

Dalam penelitian Haq (2018) [5], menyebutkan bahwa “*Google Dork* merupakan aktivitas legal jika digunakan untuk keperluan yang baik, disamping belum ada aturan suatu negara yang melarangnya, pihak Google sampai saat ini belum mengeliminasi operator yang dapat digunakan dalam implementasinya.” Yang berarti bahwa penggunaan *dork* pada mesin pencari seperti Google dapat disalahgunakan untuk mendapatkan informasi apapun yang tersebar di internet, seperti informasi sensitif.

Cernica, Popescu, țigănoaia (2019) [6], menjelaskan bahwa melindungi *website* dari berbagai ancaman merupakan suatu tugas yang penting. Dalam penelitian ini memiliki contoh kasus analisis *data leak* yang disebabkan oleh *backup plugin* yang terdapat pada *content management system* Wordpress. Bocornya informasi sensitif seperti *file backup* dapat membuat *website* menjadi dapat diretas, mengingat bahwa Wordpress merupakan *content management system* yang digunakan oleh jutaan orang di dunia.

Bocornya informasi sensitif dapat menimbulkan berbagai bahaya baik itu pada sistem maupun manusia. Informasi sensitif dapat berupa data finansial, catatan kesehatan, lokasi geografis, informasi kontak, status sistem, rahasia bisnis, konfigurasi dan status dari suatu jaringan, metadata dan sebagainya. [7]

Toffalini, Abbà, Carra, Balzarotti (2016) [8], menjelaskan seorang penyerang dapat memanfaatkan berbagai macam teknologi untuk mendapatkan informasi sensitif

seperti dengan membuat kueri pada mesin pencari untuk melakukan *fingerprint* pada target di internet. Dalam penelitian tersebut dijelaskan bagaimana menangani *fingerprint* yang dilakukan menggunakan *Google Dork*.

Kerentanan informasi sensitif dapat muncul dalam berbagai cara, namun Portswigger telah mengkategorikan tiga hal seperti tidak menghapus suatu konten internal dan mengakibatkan muncul di publik, konfigurasi pada website yang tidak benar, kecacatan pada desain dan *behavior* dari aplikasi. [9]

Google menggunakan *software* bernama "*web crawlers*" yang berguna untuk menjelajahi situs agar dapat terindeks dalam mesin pencari Google. Google penelusuran memiliki tiga tahapan yang menjelaskan bagaimana mesin pencari tersebut bekerja untuk pengguna yaitu dengan proses *crawling*, proses *indexing* dan proses *servicing search results*. Hal ini dapat mendasari bagaimana seorang penyerang dapat memanfaatkan bagaimana cara kerja Google untuk mendapatkan informasi sensitif yang ada di internet. [10]

Terdapat suatu program pendeteksi *file backup* karya rakjong (2019) [11] memiliki konsep yang sama dengan *Versioning Catcher Software* (VCS) untuk melakukan pendeteksian pada suatu *website*. Dimana *file backup* juga merupakan informasi sensitif yang berbahaya apabila *file* tersebut berada dalam server *production* dan tidak ditangani dengan baik.

Carlton, Levy, Ramim (2019) [12], menyebutkan bahwa kesalahan pengguna yang memiliki kemampuan *cybersecurity* yang buruk dapat mengakibatkan ancaman dunia maya terhadap suatu organisasi hingga 95 persen. Hal ini akan berdampak juga terhadap finansial dan kekayaan intelektual yang substansial.

OWASP membuat daftar 10 tipe kerentanan pada web aplikasi yang sering terjadi di internet, dalam sepuluh tipe kerentanan tersebut terdapat *Sensitive Data Exposure* dan *Security Misconfiguration*. Kedua tipe kerentanan tersebut memiliki keterkaitan atas terjadinya kasus peretasan pada sebuah website. *Security misconfiguration* merupakan isu yang sering terjadi dan apabila tidak segera diperbaiki akan menimbulkan masalah baru yaitu pencurian informasi sensitif (*sensitive data exposure*). [13]

Cortes [14], menjelaskan bahwa konfigurasi server yang tidak tepat akan mengakibatkan informasi sensitif menjadi terekspos ke publik. Dengan memanfaatkan fitur *dork* dari mesin pencari Google, penyerang akan dapat dengan mudah mendapatkan informasi sensitif tersebut. Informasi sensitif ini akan digunakan untuk

mengeksploitasi target. Metode ini seringkali digunakan oleh peretas untuk menemukan informasi sensitif pada atarget yang rentan.

Sebagian besar *file* dalam *web server* ditangani langsung oleh server itu sendiri, namun tidak jarang juga ditemukan suatu *file* yang terlupakan atau tidak direferensikan. *File* ini dapat digunakan untuk mendapatkan informasi penting tentang infrastruktur atau kredensial. Melakukan pengeditan di tempat atau tindakan administratif lainnya pada *production web server* mungkin secara tidak sengaja meninggalkan salinan cadangan, baik yang dibuat secara otomatis oleh *editor* saat mengedit *file*, atau oleh *administrator* yang sedang membuat zip satu set *file* untuk membuat cadangan [1]. Maka dari itu, *file versioning* yang bermuatan informasi sensitif harus segera ditangani.

Penelitian terdahulu tersebut menjadi penunjang untuk penelitian ini dibuat, dikarenakan ketika terdapat kasus dimana seorang *developer* atau *sysadmin* lupa untuk menghapus *file source code* yang telah dilakukan *versioning* baik itu secara manual maupun otomatis maka akan menimbulkan suatu kerentanan yaitu *source code disclosure*, karena *file source code* yang telah dilakukan *versioning* dengan menambahkan ekstensi di akhir nama *file* oleh *developer* maupun *sysadmin*, hal ini akan merubah juga *filename extension*, sehingga *file versioning* pada *source code* tersebut tidak akan diolah oleh sisi server melainkan akan dianggap sebagai *file text* biasa atau *non-web files* dan dapat dilihat atau diunduh oleh siapapun yang mengunjungi *file* atau *source code* tersebut dan tentunya hal ini akan menjadi ancaman yang berbahaya bagi server.

Dalam penelitian ini akan lebih ditekankan pada proses pendeteksian dengan contoh kasus seorang *system administrator* yang melakukan *file versioning* dengan menambahkan ekstensi-ekstensi umum pada akhir nama *file*. Sehingga teknik yang digunakan oleh program dalam mendeteksi adanya *file versioning* adalah dengan menyiapkan sebuah *wordlist* berisi ekstensi umum, lalu melakukan perulangan *request file* yang memiliki akhiran ekstensi umum *file versioning* tersebut ke *website* target, server akan menanggapi permintaan tersebut dan memberikan respon berupa kode status HTTP dan hasil deteksi akan disimpan dalam sebuah *file* teks. Program dibuat menggunakan bahasa pemrograman Python (Python3). Python merupakan bahasa pemrograman tingkat tinggi yang sering digunakan untuk berbagai hal seperti pengembangan web, aplikasi *mobile*, permainan video, AI, *machine learning*. Dalam pembuatannya, program ini menggunakan *module* utama yaitu *module requests* yang memungkinkan untuk membuat sebuah permintaan HTTP menggunakan Python.

Metode Penelitian

Dalam pelaksanaan penelitian, penulis memilih Penelitian dan Pengembangan sebagai metode penelitian, dikarenakan penulis akan membuat sebuah program bernama VCS (*Versioning Catcher Software*) Information untuk membantu memecahkan masalah *file versioning* yang masih tersimpan di server *production*. Langkah-langkah dalam perancangan adalah sebagai berikut:



Gambar 1 Alur perancangan penelitian

Penjelasan dari gambar 1 adalah sebagai berikut:

- a) **Tahap Identifikasi Masalah:** Merupakan tahap awal dari penelitian dengan melakukan identifikasi terhadap masalah yang menjadi topik penelitian di lapangan secara aktual.
- b) **Tahap Perancangan dan Pembuatan Program:** Pada tahap kedua ini peneliti akan merancang dan membuat program menggunakan bahasa pemrograman Python3.
- c) **Tahap Implementasi dan Pengujian:** Pada tahap ketiga, peneliti akan mengimplementasikan dan melakukan pengujian program yang telah dibuat ke lab lokal dan *website* yang telah disiapkan.
- d) **Tahap Penulisan Laporan Penelitian:** Merupakan tahap akhir dimana peneliti akan mendokumentasikan hasil penelitian dengan menuliskan laporan penelitian.

Alat-alat yang digunakan dalam melakukan penelitian adalah sebagai berikut:

a) Raspberry Pi 3B+

Merupakan sebuah komputer yang memiliki ukuran sebesar kartu kredit dan dapat bekerja seperti komputer pada umumnya. Raspi dipilih karena harganya yang relatif murah dan sangat cocok jika digunakan untuk membuat lab simulasi. Dalam hal ini, Raspberry Pi akan digunakan sebagai server mini target.

b) Sistem Operasi Raspbian Lite

Raspbian merupakan salah satu dari banyak varian sistem operasi yang dapat berjalan di perangkat Raspberry Pi. Raspbian Lite ini memiliki ukuran yang relatif kecil sehingga tidak akan memakan banyak sumber daya terutama dalam penggunaan penyimpanan. Penulis memilih varian Raspbian Lite dikarenakan kebutuhan lab yang hanya akan menjalankan beberapa layanan saja.

c) Apache *Web server*

Untuk mengelola *website* yang akan dibangun, maka diperlukan *software server* yang sesuai. Apache merupakan satu dari sekian banyak *web server* yang ada, karena *website* yang dibangun merupakan web sederhana, maka Apache adalah *web server* yang cocok untuk digunakan.

d) Kabel UTP

Unshielded Twisted Pair merupakan salah satu tipe kabel yang digunakan untuk membuat server dengan *client* dapat terkoneksi menjadi satu jaringan, sehingga keduanya dapat saling berkomunikasi.

e) Laptop

Dalam sebuah simulasi lab yang dibangun dibutuhkan perangkat komputer yang nantinya akan digunakan untuk menjalankan program VCS dari sisi *client*.

f) WSL

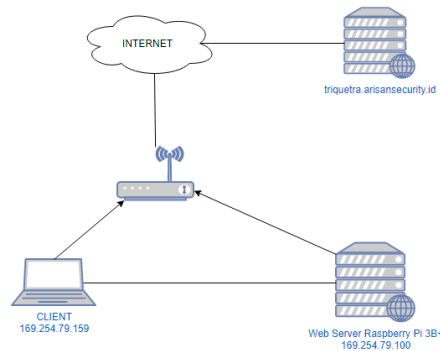
Windows Subsystem for Linux (WSL) merupakan sebuah *software* yang memungkinkan seseorang dapat menjalankan perintah-perintah Linux dalam sistem operasi Windows. WSL ini akan digunakan pada Laptop *client* guna untuk menjalankan program yang penulis buat.

Hasil dan Pembahasan

Perancangan

Sesuai dengan alur perancangan metode penelitian, sebelum melakukan tahap pembuatan, implementasi dan pengujian program akan dilakukan perancangan terlebih dahulu. Pada tahap ini, penulis mempersiapkan lab untuk implementasi dan pengujian

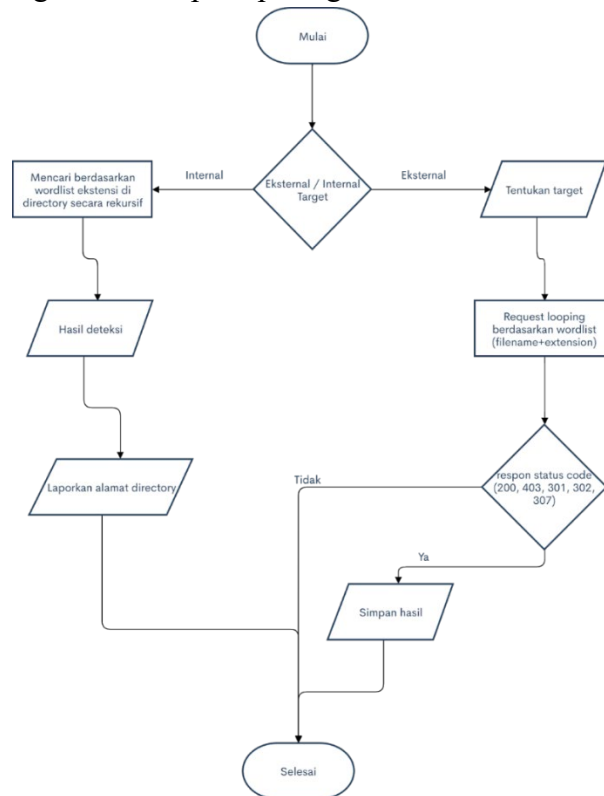
yang berisi komputer *client*, server mini Raspberry Pi, *access point* yang terhubung dengan internet, kemudian untuk pengujian pada target yang aktif akan menggunakan *subdomain website* Arisan Security. Desain lab yang digunakan dalam penelitian ini disajikan pada gambar 2.



Gambar 2 Desain lab pada penelitian

Pembuatan program

Dalam memecahkan masalah, program *Versioning Catcher Software* (VCS) memiliki tahapan diagram alur seperti pada gambar 3 berikut.



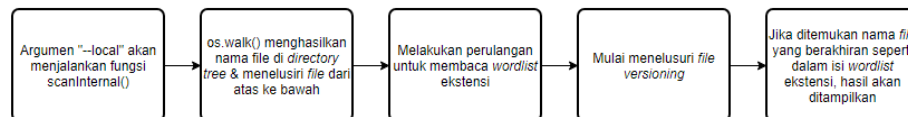
Gambar 3 Diagram alur program *Versioning Catcher Software*

Gambar 3 menunjukkan bagaimana program *Versioning Catcher Software* (VCS) bekerja. Dimulai dari penentuan target itu internal atau eksternal, apabila target merupakan server internal maka program akan menjalankan fungsi `scanInternal()`, fungsi ini akan mencari *file versioning* pada *current directory* dimana program ini dijalankan, proses pendeteksian didasarkan pada *wordlist* ekstensi umum yang telah disiapkan, apabila terdapat *file* yang memiliki akhiran ekstensi yang ada di dalam *wordlist*, maka program akan menampilkan lokasi dimana *file* tersebut berada. Berikut gambar 4 menunjukkan kode fungsi `scanInternal` yang digunakan.

```
def scanInternal():  
    for dirpath, dirs, files in os.walk("."):   
        for filename in files:  
            fname = os.path.join(dirpath, filename)  
            with open("extension.txt", "r") as ext:  
                extension = ext.read().split()  
            if fname.endswith(tuple(extension)):  
                print(fname)
```

Gambar 4 `scanInternal` function

Berdasarkan gambar 4, fungsi `scanInternal` berguna mencari *file versioning* dengan skenario ketika seseorang memiliki akses masuk ke server dan berada dalam server secara internal. Dengan memanfaatkan metode `walk()` dalam modul `os`, program akan mendeteksi *file versioning* secara rekursif dalam direktori dimana program dijalankan.



Gambar 5 *Flow* teknik mendeteksi pada fungsi `scanInternal`

Gambar 5 merupakan *flow* dari fungsi `scanInternal`, dengan menambahkan argument "--local", program akan mengeksekusi fungsi `scanInternal()` untuk menelusuri *file versioning* dalam direktori secara rekursif. Program akan melakukan perulangan dengan membaca *wordlist file* berisi ekstensi umum yang dipersiapkan dan menggabungkannya dengan *wordlist* berisi nama *file*. Program akan mulai menelusuri *file versioning* pada ada direktori dimana program ini dijalankan dan apabila terdapat nama *file* yang sama dengan gabungan *wordlist*, maka lokasi direktori *file versioning* tersebut akan ditampilkan.

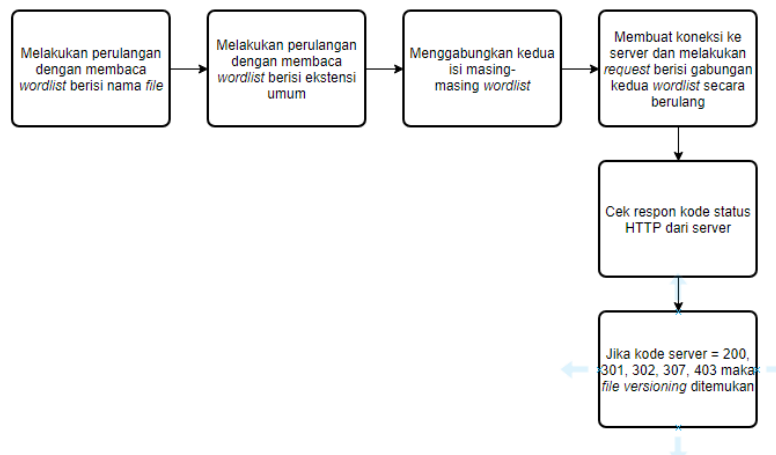
```

def runexec(target):
    startBanner()
    founded = []
    with open('filename.txt','r') as f:
        first = f.read().split()
    for one in first:
        with open('extension.txt', 'r') as s:
            second = s.read().split()
            for two in second:
                concat = one + "." + two
                web = requests.get(str(target) + str(concat))
                statusCode = str(web.status_code)
                print(stylize("Request to " + str(target) + str(concat), normal))
                if statusCode == "200" or statusCode == "403" or statusCode == "301" or statusCode == "302" or statusCode == "307":
                    print(stylize("[+] Found: " + target + str(concat) + "\t[" + statusCode + "]" + "\n", ketemu))
                    founded.append(concat)
                else:
                    print(stylize("[-] Not Found \t[" + statusCode + "]", fail))
    print("\n[!] Finished: " + str(datetime.datetime.now()) + "\n", normal)
    print("\nFound:\n" + str(target) + str(founded), normal)
    f = open("output.txt","w")
    f.write(str(founded))
    f.close()
    pwd = os.getcwd()
    print("Output saved in: " + pwd + "/output.txt", normal)

```

Gambar 6 runexec() function

Seperti yang ditunjukkan pada gambar 6, program ini juga dapat dijalankan untuk target eksternal dengan menjalankan fungsi runexec(target) dimana “target” adalah hasil masukan yang diberikan oleh *user*.

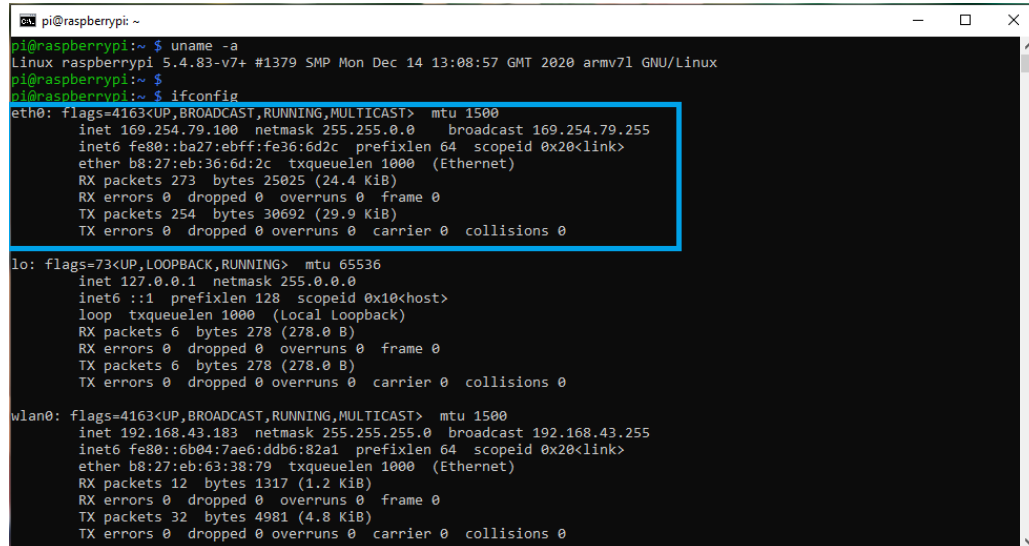


Gambar 7 Flow fungsi runexec()

Proses deteksi *file versioning* dalam fungsi runexec() ditunjukkan pada *flow* gambar 7. Program bekerja dengan melakukan perulangan membaca isi *wordlist* ekstensi dan *wordlist* nama *file* yang telah disiapkan kemudian kedua *wordlist* digabungkan. Program akan mengirimkan *request* HTTP secara berulang-ulang dengan membawa gabungan *wordlist* nama *file* dengan *wordlist* ekstensi ke *website* target, proses ini akan berlangsung terus-menerus hingga isi dari *wordlist* tersebut habis. Dalam setiap permintaan, server akan memberikan balasan berupa kode status HTTP, kode balasan ini akan berfungsi sebagai acuan untuk menentukan apakah *website* target memiliki *file versioning* atau tidak. Kode HTTP yang digunakan dalam menentukan adanya *file versioning* dalam program ini adalah 200, 301, 302, 307, 403. Hasil dari pendeteksian akan disimpan dalam bentuk *file* teks.

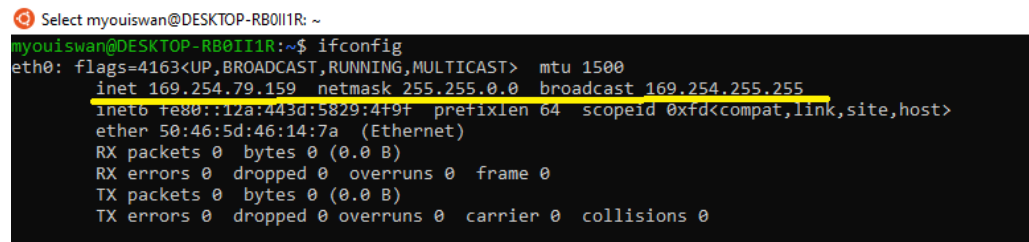
Implementasi

Pengimplementasian *Versioning Catcher Software* (VCS) dilakukan di lab implementasi dan pengujian yang telah penulis bangun yang ditunjukkan pada gambar 2. Dalam penggunaan alamat IP disetiap perangkat, penulis menggunakan sistem *Automatic Private IP Addressing* (APIPA) yang memiliki rentang IP dari 169.254.0.1 hingga 169.254.255.254 dan memiliki subnet mask 255.255.0.0.



```
pi@raspberrypi: ~  
pi@raspberrypi:~$ uname -a  
Linux raspberrypi 5.4.83-v7+ #1379 SMP Mon Dec 14 13:08:57 GMT 2020 armv7l GNU/Linux  
pi@raspberrypi:~$  
pi@raspberrypi:~$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 169.254.79.100 netmask 255.255.0.0 broadcast 169.254.79.255  
    inet6 fe80::ba27:ebff:fe36:6d2c prefixlen 64 scopeid 0x20<link>  
    ether b8:27:eb:36:6d:2c txqueuelen 1000 (Ethernet)  
    RX packets 273 bytes 25025 (24.4 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 254 bytes 30692 (29.9 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 6 bytes 278 (278.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 6 bytes 278 (278.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.43.183 netmask 255.255.255.0 broadcast 192.168.43.255  
    inet6 fe80::6b04:7ae6:ddb:82a1 prefixlen 64 scopeid 0x20<link>  
    ether b8:27:eb:63:38:79 txqueuelen 1000 (Ethernet)  
    RX packets 12 bytes 1317 (1.2 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 32 bytes 4981 (4.8 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Gambar 8 Alamat IP *private* Raspberry Pi



```
Select myouiswan@DESKTOP-RB0II1R: ~  
myouiswan@DESKTOP-RB0II1R:~$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 169.254.79.159 netmask 255.255.0.0 broadcast 169.254.255.255  
    inet6 fe80::12a:443d:5829:4f9f prefixlen 64 scopeid 0xfd<compat,link,site,host>  
    ether 50:46:5d:46:14:7a (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 0 (0.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Gambar 9 Alamat IP *private* laptop (WSL)

Gambar 8 merupakan alamat IP dari server mini Raspberry Pi, yaitu 169.254.79.100 dengan subnet mask 255.255.0.0. Sedangkan gambar 9 merupakan alamat IP dari laptop yang berfungsi sebagai *client* 169.254.79.159 dengan subnet mask 255.255.0.0.

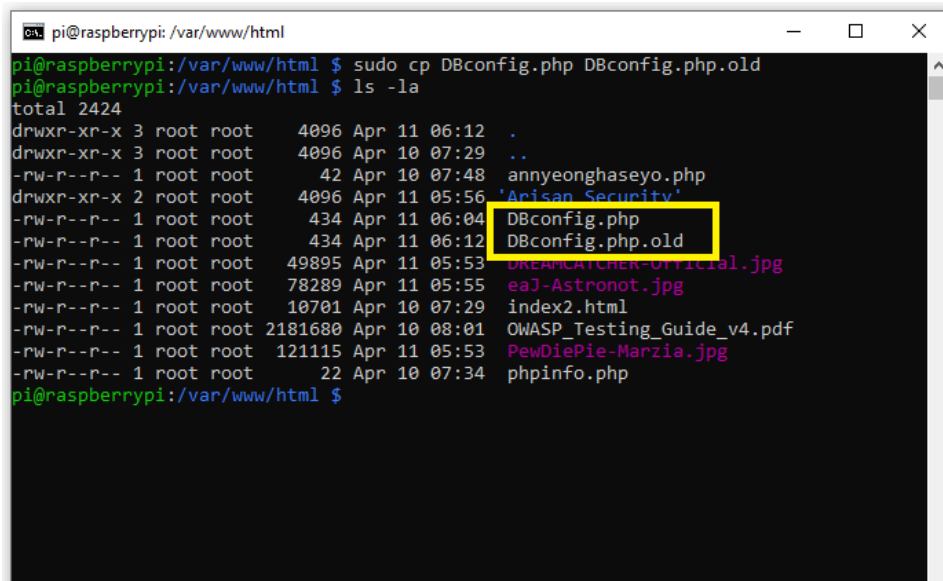
Index of /

	Name	Last modified	Size	Description
	Arisan Security/	2021-04-11 05:56	-	
	DBconfig.php	2021-04-11 06:04	434	
	DREAMCATCHER-Official.jpg	2021-04-11 05:53	49K	
	OWASP_Testing_Guide_v4.pdf	2021-04-10 08:01	2.1M	
	PewDiePie-Marzia.jpg	2021-04-11 05:53	118K	
	anyeonghaseyo.php	2021-04-10 07:48	42	
	eaJ-Astronot.jpg	2021-04-11 05:55	76K	
	index2.html	2021-04-10 07:29	10K	
	phpinfo.php	2021-04-10 07:34	22	

Apache/2.4.38 (Raspbian) Server at 169.254.79.100 Port 80

Gambar 10 Tangkapan layar isi Server Raspberry Pi dari luar

Gambar 10 merupakan tampilan isi file dari *web server* ketika mengakses IP lokal server menggunakan browser di sisi *client*. Isi dari dalam *web server* tersebut adalah *file* PDF, gambar, *file* bahasa pemrograman, folder dan hasil dari *file versioning* bahasa pemrograman.



```











pi@raspberrypi: /var/www/html
pi@raspberrypi: /var/www/html $ sudo cp DBconfig.php DBconfig.php.old
pi@raspberrypi: /var/www/html $ ls -la
total 2424
drwxr-xr-x 3 root root 4096 Apr 11 06:12 .
drwxr-xr-x 3 root root 4096 Apr 10 07:29 ..
-rw-r--r-- 1 root root 42 Apr 10 07:48 anyeonghaseyo.php
drwxr-xr-x 2 root root 4096 Apr 11 05:56 'Arisan_Security/'
-rw-r--r-- 1 root root 434 Apr 11 06:04 DBconfig.php
-rw-r--r-- 1 root root 434 Apr 11 06:12 DBconfig.php.old
-rw-r--r-- 1 root root 49895 Apr 11 05:53 DREAMCATCHER-Official.jpg
-rw-r--r-- 1 root root 78289 Apr 11 05:55 eaJ-Astronot.jpg
-rw-r--r-- 1 root root 10701 Apr 10 07:29 index2.html
-rw-r--r-- 1 root root 2181680 Apr 10 08:01 OWASP_Testing_Guide_v4.pdf
-rw-r--r-- 1 root root 121115 Apr 11 05:53 PewDiePie-Marzia.jpg
-rw-r--r-- 1 root root 22 Apr 10 07:34 phpinfo.php
pi@raspberrypi: /var/www/html $

```

Gambar 11 Source code file versioning

Dalam skenario penelitian ini, gambar 11 menunjukkan hasil terjadinya *file versioning* dengan menambahkan ekstensi umum tambahan `.old` di akhir nama *file*. *File* awal bernama “DBconfig.php” berubah menjadi “DBconfig.php.old”. Kebiasaan *file*

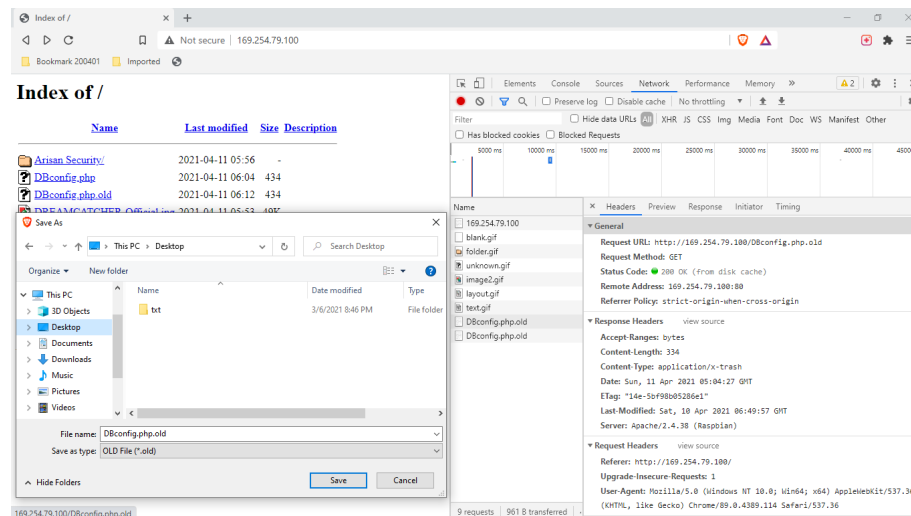
versioning semacam ini akan memunculkan ancaman yang berbahaya bagi server karena dapat memungkinkan terjadinya *sensitive information disclosure*. Ketika hal ini terjadi, seorang *threat actor* dapat dengan mudah mendapatkan *file* tersebut lalu membaca atau menganalisa isi dari *file* dan memanfaatkannya untuk mencoba menerobos masuk ke dalam sistem.

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Arisan Security/	2021-04-11 05:56	-	
 DBconfig.php	2021-04-11 06:04	434	
 DBconfig.php.old	2021-04-11 06:12	434	
 DREAMCATCHER-Official.jpg	2021-04-11 05:53	49K	
 OWASP_Testing_Guide_v4.pdf	2021-04-10 08:01	2.1M	
 PewDiePie-Marzia.jpg	2021-04-11 05:53	118K	
 annyeonghaseyo.php	2021-04-10 07:48	42	
 eaJ-Astronot.jpg	2021-04-11 05:55	76K	
 index2.html	2021-04-10 07:29	10K	
 phpinfo.php	2021-04-10 07:34	22	

Apache/2.4.38 (Raspbian) Server at 169.254.79.100 Port 80

Gambar 12 File versioning DBconfig.php.old

Gambar 12 menunjukkan tampilan hasil *file versioning* dari sisi *client*. Terdapat *file* awal bernama “DBconfig.php” dan “DBconfig.php.old” yang telah dilakukan *file versioning* dengan menambahkan ekstensi umum di akhir nama *file*.



Gambar 13 Source code file versioning dapat diakses oleh remote user

Gambar 16 merupakan proses program saat dijalankan pada server Raspberry Pi. Pada tahap ini, program akan menjalankan fungsi `runexec(target)` berdasarkan target yang telah dimasukkan oleh *user*. Program akan mulai melakukan *request* terhadap server dengan menggabungkan kedua *wordlist* nama *file* dan ekstensi satu per satu hingga akhir isi dari kedua *wordlist*. Teknik yang digunakan program ini untuk menentukan kemungkinan ada atau tidaknya *file versioning* dalam server adalah dengan melihat respon kode status HTTP yang diberikan oleh server. Kode status HTTP yang digunakan ketika terdapat kemungkinan adanya *file versioning* di server adalah 200, 403, 301, 302 dan 307.

```
myouiswan@DESKTOP-RB0I1R: ~/TA/revisi
[-] Not Found [404]
Request to http://169.254.79.100/connection.php.old
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.zip
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.tar.gz
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.bzip
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.rar
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.gzip
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.7z
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.bak
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.backup
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.old
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.original
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.save
[-] Not Found [404]
Request to http://169.254.79.100/DBconfig.php.old
[*] Found: http://169.254.79.100/DBconfig.php.old [200]

[!] Finished: 2021-04-11 12:49:44.785660

Found:
http://169.254.79.100/['DBconfig.php.old']
Output saved in: /home/myouiswan/TA/revisi/output.txt
myouiswan@DESKTOP-RB0I1R:~/TA/revisi$
```

Gambar 17 Hasil deteksi *file versioning* di server Raspberry Pi

Gambar 17 menunjukkan hasil proses mendeteksi *file versioning* pada server Raspberry Pi. Berdasarkan *request* `DBconfig.php.old` yang dikirim, program menerima balasan kode status HTTP berupa 200. Sesuai dengan teknik pendeteksian yang ditunjukkan pada gambar 6 dan gambar 7, program *Versioning Catcher Software* (VCS) berhasil mendeteksi adanya kemungkinan *file versioning* yang ada di server. Kemudian hasil dari deteksi akan disimpan dalam sebuah *file* teks.

Berdasarkan lab implementasi dan pengujian yang telah dibangun, *Versioning Catcher Software* (VCS) akan diujikan ke sebuah *subdomain website* dari komunitas *cybersecurity* bernama Arisan Security yang berada di kota Salatiga. Dikarenakan *subdomain* merupakan milik penulis personal, tentunya tindakan pengujian dilakukan secara legal dan aman.

Name	Size	Last Modified	Type
secret.php	132 bytes	Today, 1:40 PM	text/x-generic
secret.php.bak	132 bytes	Today, 1:42 PM	text/x-generic

Gambar 18 File versioning pada subdomain triquetra.arisansecurity.id

Gambar 18 menunjukkan isi dari server subdomain tersebut. Dalam pengujian ini, subdomain triquetra.arisansecurity.id memiliki file bernama “secret.php.bak” dimana file ini merupakan hasil dari file versioning yang dilakukan dengan menambahkan ekstensi diakhir nama file. File versioning ini memiliki kode permission 600, dimana ketika seseorang mencoba mengakses file versioning tersebut, maka akan didapatkan respon berupa 403 forbidden oleh server. Meskipun tidak bisa didapatkan secara langsung, seorang threat actor dapat menggunakan berbagai macam attack vector yang ada untuk mendapatkan file versioning. Hasil dari proses deteksi file versioning ditunjukkan pada gambar 19.

```
Request to http://triquetra.arisansecurity.id/secret.php.bak
[+] Found: http://triquetra.arisansecurity.id/secret.php.bak [403]

Request to http://triquetra.arisansecurity.id/secret.php.backup
[-] Not Found [404]
Request to http://triquetra.arisansecurity.id/secret.php.old
[-] Not Found [404]
Request to http://triquetra.arisansecurity.id/secret.php.original
[-] Not Found [404]
Request to http://triquetra.arisansecurity.id/secret.php.save
[-] Not Found [404]
Request to http://triquetra.arisansecurity.id/secret.php.php.old
[-] Not Found [404]

[!] Finished: 2021-04-11 14:10:57.960940

Found:
http://triquetra.arisansecurity.id/['secret.php.bak']
Output saved in: /home/myouiswan/TA/revisi/output.txt
myouiswan@DESKTOP-RB0II1R:~/TA/revisi$
```

Gambar 19 Hasil deteksi file versioning pada subdomain

Pada gambar 19, dapat dipastikan bahwa hasil request secret.php.bak yang dilakukan oleh program ini memiliki balasan kode status 403 dari server. Kode 403 menandakan bahwa terdapat file yang bersifat rahasia dan hanya orang yang memiliki wewenang saja yang dapat mengaksesnya. Program Versioning Catcher Software (VCS) berhasil mendeteksi adanya kemungkinan file versioning pada target eksternal subdomain triquetra.arisansecurity.id.

Pengujian program dengan target internal dilakukan pada server Raspberry Pi. Dalam skenario ini memiliki contoh kasus seorang system administrator ingin mencari file versioning yang ada dalam directory server, dengan kebiasaan sysadmin yang

memberikan ekstensi di akhir nama *file* seperti pada contoh kasus target eksternal untuk melakukan *versioning*.

```
pi@raspberrypi: ~/revisi
pi@raspberrypi:~/revisi $ ls -la secretFolder/
total 8
drwxr-xr-x 2 pi pi 4096 Apr 11 08:19 .
drwxr-xr-x 4 pi pi 4096 Apr 11 08:21 ..
-rw-r--r-- 1 pi pi   0 Apr 11 08:19 configuration.backup
pi@raspberrypi:~/revisi $
```

Gambar 20 Tangkapan layar dari dalam server

Gambar 20 merupakan contoh skenario *file versioning* dari dalam server Raspberry Pi. Terdapat *file* bernama “configuration.backup”, *file* ini akan digunakan untuk melakukan pengujian program pada target internal.

```
pi@raspberrypi: ~/revisi
pi@raspberrypi:~/revisi $ python3 vcs.py -l

### ###          ###          ##
# #              # #          #
# #              # #          #
# #              ## ##      ## ## ## ## ##
# #              # # # # # # # # # # # #
# # ##### #      ### # # # # # # # # #
# #              # # # # # # # # # #
# #              ### ##### ## ## ## ## ## ##

Tugas Akhir \('o')/

Try with -h or --help to get some help

[!] Start Time: 2021-04-11 08:22:14.295705
./secretFolder/configuration.backup
[!] Finished: 2021-04-11 08:22:14.297265
pi@raspberrypi:~/revisi $
```

Gambar 21 Hasil deteksi *file versioning* secara internal

Gambar 21 menunjukkan hasil deteksi keberadaan *file versioning* pada target internal. Dengan menjalankan fungsi `scanInternal()`, program akan mulai mencari *file versioning* pada *current directory* dimana program dijalankan berdasarkan *wordlist* yang telah disiapkan, kemudian ketika *file* tersebut berhasil ditemukan, program akan menampilkan lokasi *directory* dimana *file versioning* berada.

Berdasarkan hasil implementasi dan pengujian yang dilakukan, maka dapat disimpulkan bahwa program *Versioning Catcher Software* (VCS) dapat mendeteksi keberadaan *file versioning* yang ada dalam server. Hasil pengujian menunjukkan

bahwa program dapat membantu untuk meminimalisir, melindungi dan mencegah *file versioning* yang bermuatan informasi sensitive bocor ke publik.

Simpulan

Dari hasil penelitian dan pengujian yang telah dilakukan, maka dapat disimpulkan bahwa *Versioning Catcher Software (VCS)* dapat membantu untuk meminimalisir terjadinya *sensitive information disclosure* pada *server production* yang diakibatkan oleh kebiasaan *sysadmin* menambahkan ekstensi umum pada akhir nama *file* sehingga mengakibatkan terjadinya *file versioning* kemudian informasi sensitif yang ada menjadi terekspos. Terdapat beberapa kekurangan dalam program ini, yaitu program ini hanya mengandalkan *wordlist* yang berisi ekstensi tambahan di akhir *file* untuk mendeteksi *file versioning* dan respon kode status HTTP dari server untuk menentukan pendeteksian, oleh karena itu perlu adanya pengecekan ulang dari sisi manusia untuk memastikan hasil deteksi tidak *false positive*. Selain itu, program akan membutuhkan waktu yang lama ketika jumlah *wordlist* ekstensi semakin banyak.

Daftar Pustaka

- [1] M. Meucci, P. Luptak, and M. Morana, *OWASP Testing Guide 4.0*. Accessed: Dec. 03, 2020. [Online]. Available: <https://owasp.org/www-project-web-security-testing-guide/stable/>
- [2] A. F. Akbar and I. Neforawati, "Penggunaan Teknologi Versioning Git dan Phinx untuk Pengembangan Aplikasi E-Commerce Berbasis Website," *E-Jurnalcom J. Has. Ris.*, 2017, Accessed: Dec. 03, 2020. [Online]. Available: <https://www.e-jurnal.com/2017/04/penggunaan-teknologi-versioning-git-dan.html>
- [3] D. Milder, J. Garlicki, and W. Mincewicz, "The Internet Data Collection with the Google Hacking Tool – White, Grey or Black Open-Source Intelligence?," *Agencja Bezpieczeństwa Wewnętrznego*, vol. 11, no. 20, pp. 280–300, Apr. 2019.
- [4] M. M. Hassan, T. Bhuiyan, and S. Biswas, "An Investigation of Educational Web Applications in Bangladesh: A Case Study on Local File Disclosure Vulnerability," *Researchgate*, 2016, Accessed: Dec. 27, 2020. [Online]. Available: https://www.researchgate.net/publication/311806770_An_Investigation_of_Educational_Web_Applications_in_Bangladesh_A_Case_Study_on_Local_File_Disclosure_Vulnerability
- [5] M. Z. U. Haq, "GOOGLE DORK, SEBUAH PENDEKATAN LANJUTAN PEMANFAATAN MESIN PENCARI SEBAGAI PENUNJANG LITERASI INFORMASI," *Unilib J. Perpust.*, vol. 8, no. 1, pp. 29–32, Oct. 2018, doi: 10.20885/unilib.vol8.iss1.art3.
- [6] I. Cernica, N. Popescu, and B. țigănoaia, "Security Evaluation of Wordpress Backup Plugins," *Int. Conf. Control Syst. Comput. Sci. CSCS*, pp. 312–316, May 2019, doi: 10.1109/CSCS.2019.00056.
- [7] CWE Content Team, "CWE-538: Insertion of Sensitive Information into Externally-Accessible File or Directory." Accessed: May 27, 2021. [Online]. Available: <https://cwe.mitre.org/data/definitions/538.html>

-
- [8] F. Toffalini, M. Abbà, D. Carra, and D. Balzarotti, “Google Dorks: Analysis, Creation, and New Defenses,” *Lect. Notes Comput. Sci.*, Jul. 2016, doi: 10.1007/978-3-319-40667-1_13.
- [9] PortSwigger, “Information Disclosure Vulnerabilities,” *Information Disclosure*. <https://portswigger.net/web-security/information-disclosure> (accessed Nov. 18, 2020).
- [10] “How Google Search Works for Beginners | Google Search Central,” *How Search Works (for beginners)*. <https://developers.google.com/search/docs/basics/how-search-works> (accessed May 27, 2021).
- [11] Rakjong, “Backup-scan,” *Backup-scan*. <https://github.com/rakjong/Backup-scan> (accessed Jul. 13, 2021).
- [12] M. Carlton, Y. Levy, and M. Ramim, “Mitigating cyber attacks through the measurement of non-IT professionals’ cybersecurity skills,” *Inf. Comput. Secur.*, vol. 27, no. 1, pp. 101–121, Jan. 2019, doi: 10.1108/ICS-11-2016-0088.
- [13] “Top 10 Web Application Security Risks,” *OWASP Top 10 Web Application Security Risks*. <https://owasp.org/www-project-top-ten/> (accessed Jun. 30, 2021).
- [14] B. Rodrigo Barbosa Cortes, “Utilizando Google Hacking para encontrar vulnerabilidades em sites,” *Rev. Exército Bras.*, vol. v. 7 n. 2 (2017): O Comunicante, Sep. 2018, Accessed: Jul. 01, 2021. [Online]. Available: <http://www.ebrevistas.eb.mil.br/OC/article/view/1724>