Meningkatkan Keamanan *Web Server* Nginx dengan NAXSI sebagai *Web Application Firewall*

Muhammad Innuddin[1],Pahrul Irfan[2], Rifqi Hammad[3],Sahrul[4]
[1]Ilmu Komputer, Fakultas Teknik, Universitas Bumigora
[2]Rekayasa Perangkat Lunak Aplikasi, Fakultas Teknik, Universitas Bumigora
[3]Rekayasa Perangkat Lunak, Fakultas Teknik, Universitas Bumigora
[4]Program Studi Ilmu Komputer, Fakultas Teknik, Universitas Bumigora
[1]inn@universitasbumigora.ac.id,[2]irfan@universitasbumigora.ac.id,
[3]rifqi.hammad@universitasbumigora.ac.id,[4]Sahrulfatih22@gmail.com

ABSTRAK

Teknologi informasi dari generasi sebelumnya hingga saat ini semakin cepat sehingga pemeliharaan dan peningkatan keamanan menjadi faktor penting yang harus diperhatikan dalammerancang dan membangun *WebServer*. Pemeliharaan layanan dengan menerapkan sistem keamanan sangat penting dilakukan untuk mencegah pihak yang tidak berwenang memperoleh informasi penting atau merusak sebuah sistem dengan berbagai jenis serangan berbahaya. Penerapan keamanan bertujuan untuk mencegah serangan yang dilakukan seperti mengubah tampilan *website*, pencurian *password* atau membuat *WebServer* tidak dapat bekerja secara normal. *NAXSI* merupakan *WebApplicationFirewall* (WAF) untuk mengatasi permasalahan keamanan pada *WebServerNginx* dengan melakukan *logging* pada aktivitas *Web* secara *realtime* untuk memantau lalu lintas *HTTP*. Jika terdapat permintaan berbahaya maka akan ditolak dan diarahkan ke halaman *forbidden*. Penelitian ini menggunakan metode *NetworkDevelopmentLifeCycle* (NDLC). Terdapat 5 skenario dilakukan dalam uji cobadiantaranya,*InformationGathering* menggunakan *WhatWeb* dan *DirBuster*, *HTTPAttack*, *XSSAttack* serta *BruteForceLogin* menggunakan *WPScan*. Pengujian performansi penanganan serangan pada *WebServer* menggunakan dua scenario yaitu sebelum diaktifkan dan setelah diaktifkan *NAXSI*. Kesimpulan dari penelitian ini adalah performa *NAXSI* sangat baik dalam melindungi *WebServerNGINX* dari berbagai jenis ancaman serangan berbahaya yang diuji karena *NAXSI* dapat mendeteksi dan mencegah adanya serangan dan dapat menstabilkan nilai *CPUusage*, penggunaan memori serta mampu menormalkan *traffic* jaringan.

Kata Kunci: Jaringan, *Nginx*, *NAXSI*, *Firewall*, *Server*.

ABSTRACT

Information technology from the previous generation to the present is getting faster so that maintenance and increasing security are important factors that must be considered when designing and building a Web Server. Service maintenance by implementing a security system is very important to prevent unauthorized parties from obtaining important information or damaging a system with various types of malicious attacks. Implementing security aims to prevent attacks such as changing the appearance of the website, stealing passwords or making the Web Server unable to work normally. NAXSI is a Web Application Firewall (WAF) to overcome security problems on the Nginx Web Server by logging Web activity in real time to monitor HTTP traffic. If there is a malicious request it will be rejected and redirected to a forbidden page. This research uses the Network Development Life Cycle (NDLC) method. There were 5 scenarios carried out in the trial, including, Information Gathering using WhatWeb and DirBuster, HTTP Attack, XSS Attack and Brute Force login using WPScan. Performance testing for handling attacks on the Web Server uses two scenarios, namely before activating and after activating NAXSI. The conclusion of

this research is that NAXSI's performance is very good in protecting the NGINX Web Server from various types of malicious attack threats that were tested because NAXSI can detect and prevent attacks and can stabilize CPU usage values, memory usage and is able to normalize network traffic.

Keywords: *Network, Nginx, NAXSI, Firewalls, Servers.*

1. PENDAHULUAN

Perkemabangan teknologi internet dari waktu ke waktu semakin pesat sehingga menjadikannya sebagai media utama dalam pertukaran informasi. Berdasarkan data BPS hasil Survei tahun 2022 terdapat 66,48% (persen) penduduk Indonesia telah mengakses internet pada tahun 2022 dan 62,10% di tahun 2021[1].

"Internet merupakan jaringan luas dan bersifat publik, oleh karena itu diperlukan suatu usaha untuk menjamin keamanan informasi terhadap data atau layanan yang menggunakan internet"[2]. Web Server berguna untuk melayani dan memfungsikan situs Web, dalam proses pemeliharaan Web Server ada beberapa yang perlu dipertimbangkan, diantaranya sistem keamanan. Menurut [3]"Keamanan jaringan merupakan segala aktifitas pengamanan suatu jaringan dengan bertujuan menjaga privacy, integrity, availability, authentication, access control dan safety terhadap suatu serangan".

Berdasarkan data yang dirilis oleh [4]menyebutkan bahwa kejahatan cyber melalui SQL Injection berada diurutan pertama pada kategori kejahatan cyber menyerang Web Application dengan 27%. Sebanyak 17% kejahatan cyber berupa Path Traversal berada diurutan kedua dan diikuti Cross Site Scripting sebanyak 14%. Berdasarkan data tersebut, pihak web developer dan web administrator harus memperhatikan keamanan website agar website tersebut dapat menjamin keamanan data user, sehingga diperlukan peningkatan keamanan untuk mengurangi resiko kejahatan cyber yakni denganmenerapkan Web Application Firewall.

Menurut Anggraenni(2013)"WAF merupakan metode pengamanan pada aplikasi web yang memiliki beberapa fungsi, diantaranya monitoring trafik, secure directory, pemfilteran string dan proteksi terhadap serangan, seperti SQL Injections, Cross-Site Scripting, dan Unrestricted File Upload". Penelitian ini, membahas penggunaan WAF dengan menggunakan NAXSI sebagai solusi untuk pengamanan Web Server Nginx. NAXSI (Nginx Anti XSS & SQL Injection) adalah modul pihak ketiga unt uk Web Server Nginx. Pengembang menerapkan kekuatan perlindungan dan kesederhanaan pada aturan dan ketahanannya yang tinggi terhadap pengelakan [6].

Menurut Crystallography (2016)"Nginx adalah software open-source yang memiliki kinerja tinggi sebagai server HTTP dan reverse proxy. Hal ini dapat menyebarkan dinamis HTTP konten di jaringan menggunakan FastCGI handler untuk script dan dapat berfungsi sebagai perangkat lunak yang mampu menyeimbangkan beban. Nginx dibangun secara modular dengan demikian mampu mendukung berbagai fitur seperti Load Balancing dan Reverse Proxying, Virtual hosts berbasis nama dan IP, Fast CGI, akses langsung ke cache, SSL, Flash Video Streaming dan sejumlah fitur-fitur standar lainnya. Nginx dapat dijalankan dan tersedia untuk platform Unix, Linux, varian dari BSD, MacOS X, Solaris, dan Microsoft Windows".

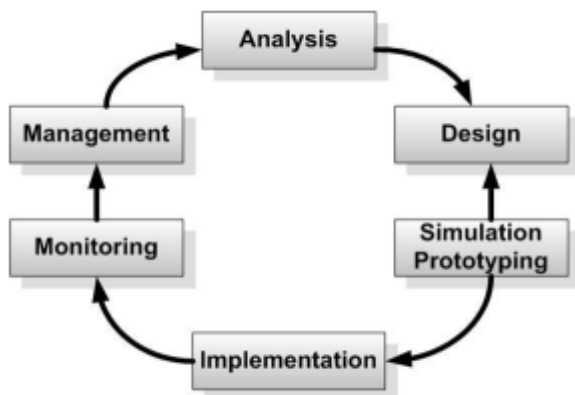
Berdasarkan penelitian yang sebelumnya pernah dilakukan oleh [8] tentang"Analisa WAF (Web Application Firewall) menggunakan NGINX terhadap serangan SQL Injection".Penelitian ini menyatakan bahwa implementasi firewall aplikasi web menggunakan ModSecurity untuk mencegah serangan SQL Injection. Penelitian serupa juga dilakukan [9],

menjelaskan bahwa dengan "menerapkan Sistem Keamanan pada WEB Menggunakan Application Firewall dengan dengan module dan rule ModSecurity dapat melindungi web server dari serangan SQL Injection, Cross Site Scripting (XSS), dan Command Excecution". Penelitian sejenis juga dilakukan oleh [10] dengan judul "Implementasi dan Analisis Open Source RaptorWAF Pada Aplikasi Web Berdasarkan Standar PTES"menyimpulkan bahwa penelitian ini berhasil memblokir serangan SQL Injection, Cross-Site Scripting dan Local File.

Berdasarkan pembahasan di atas, mendorong peneliti untuk menganalisa penerapan pengamanan NAXSI untuk menangani serangan pada Web Server Nginx. Manfaat penelitian ini adalah untuk menambah wawasan berupa pen getahuan tentang pengamanan serangan terhadap Web Server.

2. METODE PENELITIAN

Penelitian ini menggunakan metodologi NDLC (Network Development Life Cycle). Menurut [9] "NDLC adalahmodel yang mendefinisikan proses perancangan atau pengembangan system jaringan computer. Adapun tahapan pada NDLC yaitu tahap analysis, tahap design, tahap simulation prototype, tahap implementation, tahap monitoring dan tahap management". Dari keseluruhan tahapan tersebut penelitihanya menggunakan 4 tahapan antara lain tahap Analysis, tahap Design, tahap Simulation Prototyping dan tahap Implementation. Adapun tahapan tersebut dapat dilihat pada gambar tersebut.



Gambar 1. *NetworkDevelopmentLifeCycle* (NDLC)

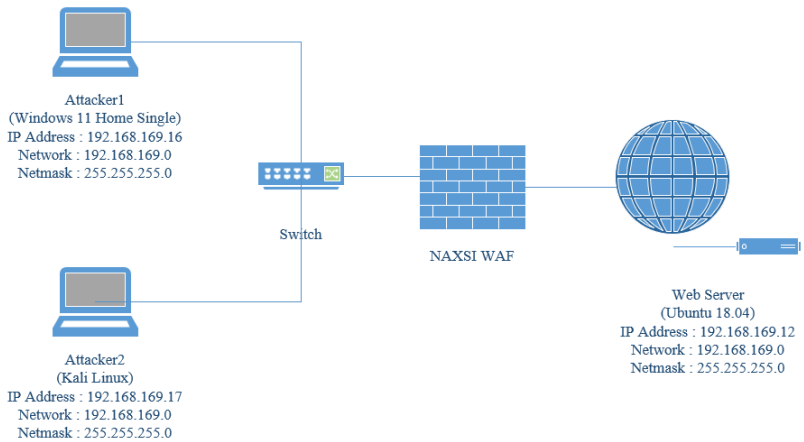
2.1. Tahap Analisis

Tahap ini peneliti mengumpulkan data-data yang terkait atau relevan dengan penelitian yang dibahas dengan beberapa metode diantaranya studi literatur dengan mereview dan mempelajari beberapa jurnal yang terkait Web Application Firewall, NAXSI serta Web Server dan untuk melengkapai beberapa data peneliti juga mencari data di internet terkait materi yang dibahas setelah peneliti melakukan pengumpulan data dari berbagai sumber berikutnya peneliti menganalisa data yang benar-benar dibutuhkan terkait penelitian yang dibahas baik dari segi kemanan dan serangan pada Web Servernya. Analisis data kebutuhan penelitian adalah tahapan yang sangat menentukan keberlangsungan dalam meneliti karena jika sampai salah dalam memilih data maka output dari penelitian akan gagal.

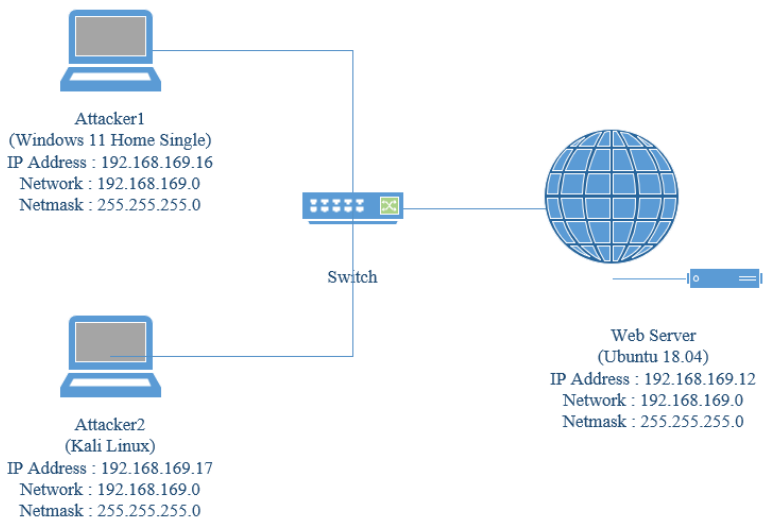
2.2. Tahap *Design*

Tahap design dilakukan setelah tahap analisis selesai, sehingga dari hasil analisis data yang dilakukan sebelumnya didapatkan beberapa rancangan atau design ujicoba jaringan

untuk mengetahui performa dari penerapan *NAXSI* ke*WebServer*apakah dengan menerapkan *NAXSI* dapat meningkatkan kemanan pada *WebServer* atau sebaliknya. *Design* uji coba penerapan *NAXSI* ke *WebServer* dapat dilihat pada gambar di bawah.



Gambar 2. *Design*Topologi Uji Coba *NAXSI* ke *WebServer*



Gambar 3. Design Topologi ujicoba Web Server tanpa NAXSI

Adapun penjelasan design topologi di atas sebagai berikut:

Setiap*request* yang masuk ke dalam server dari *Attacker1* atau *Attacker2* pada server akan di cek oleh *NAXSI*, apabila *request* tersebut sesuai dengan *rulesNAXSI* maka *request*yang masuk langsung di drop dan tidak diteruskan ke server. Sebaliknya apabila di *rulesNAXSI*aturannya diizinkan, maka paket akan diteruskan ke server.Setiap*Request* yang masuk ke server dari *Attacker1* ataupun *Attacker2*maka akanlangsung diteruskan ke server *NGINX* tanpa *FirewallNAXSI*, unt uk melihat perfoma server sebelum dan sesudah mengaktifkan *NAXSI*.

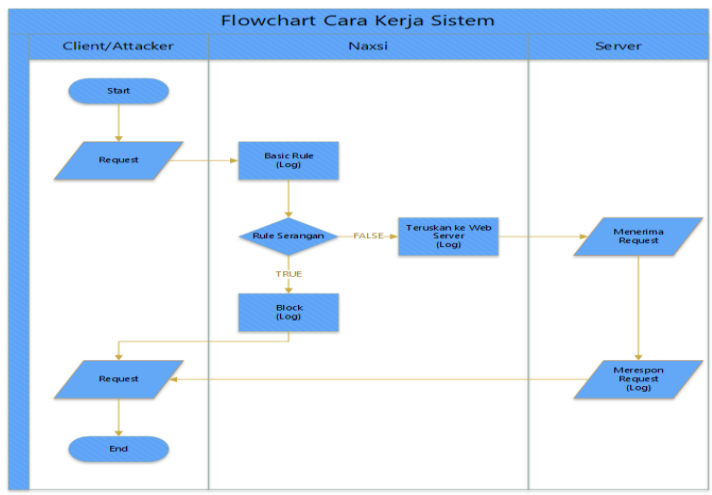
Setelah *design* topologi uji coba dilakukan maka tahap selanjutnya adalah men-*design*pengalamatan IP untuk mengetahui performa Web Server ketika menggunakan

NAXSI.Pengalamatan IP menggunakan kelas C, 192.168.169.0/24.Adapun *design* atau rancangan pengalamatan IP nya dapat dilihat pada table di bawah.

Tabel 1. Design atau Rancangan Pengalamatan IP

No	Perangkat	IPAddress	Netmask
1	Web Server	192.168.169.12	255.255.255.0
2	Attacker1(Win dows11)	192.168.169.16	255.255.255.0
3	Attacker2(KaliLinux)	192.168.169.17	255.255.255.0

Adapun alur kerja atau *flowmap* pada penelitian ini agar mempermudah peneliti dan pembaca memahami alur penelitiannya dapat dilihat sebagai berikut.



Gambar 4. *Design* atau Rancangan Alur Kerja Sistem

Flowmap di atas menjelaskan proses kerja sistem pada penelitian. Setiap *packetrequest* yang dilancarkan*Attacker*akan filteroleh *NAXSI* sebelum menuju ke server tujuan. Apabila aturan di rules*NAXSI* mengizinkan *packet*tersebut untuk lewat, maka paket diteruskan keserver tujuan. Apabila pada rules *NAXSI* terdapat aturan yang tidak mengizinkan packet request untuk diteruskan, maka *packetrequest* tersebut di drop dan di blokir. *NAXSI* mengeluarkan pemberitahuan *error* apakah paket diizinkan atau dibuang berupa log.

2.3. Tahap Konfigurasi

Pada tahapan ini ada beberapa konfigurasi dilakukan oleh peneliti, diantaranya:

1). Konfigurasi *WebServer*

Konfigurasi*webserver*dengan mengatur IP *AdressInterface*-nya dilakukan supaya*webserver*dapat terhubungke jaringan yang sedang dikelola. Setelah peneliti melakukan pengaturan konfigurasi berikutnya peneliti*update* Ubuntu ke versi 18.04. *Update* ini peneliti lakukansupaya saat melakukan instalasi *packet* aplikasi yang diperlukan dapat berjalan dengan baik. Berikutnya menginstalasi *packet* yang diperlukan dalam membangun *webserver*seperti

BIND sebagai *DomainNameSystem* (DNS), *NGINX* sebagai *packetserver* internet, *MySQLserver* sebagai *DatabaseManagementSystem*, dan *WordPress* sebagai CMS.

2). Konfigurasi *Attacker*

Berikutnya melakukan konfigurasi IP *Addressinterface* dikomputer *Attacker*dilanjutkan dengan mengatur DNS pada komputer *client*agar dapat terhubung ke komputer *webserver* yang beda jaringan.

3). Konfigurasi *NAXSI*

Melakukan konfigurasi *NAXSI*dan IP *Address* dimasing-masing *interface* serta menyeting*NAXSI* sebagai *firewall* agar *request packet*yang ke*webserver* harus melalui *FirewallNAXSI* terlebih dahulu untuk di filter. Setelah itu melakukan konfigurasi pada *NAXSI* supaya informasi*error* yang diterima langsung dikirim ke direktori yang sudah disediakan sebelumnya berupa file log.

2.4. Tahap Implementasi (Uji Coba)

Tahapan ini menjelaskan tentang pengujian hasil konfigurasi yang telah dilakukan pada tahap sebelumnya.Pengujiandilakukan dengan mengkoneksikan antara komputer *Server*dengan komputer *Attacker*. Komputer *Attacker* melakukan serangan pada halaman *Website* yang ditanamkan pada komputer *server*. Pengujian Topologi jaringan ini mengikuti *flowmap* atau alur kerja yang sudah dirancang sebelumnya.

Pengujian atau uji coba juga dilakukan dengan cara menyisipkan*rules* yang telah disiapkan untuk *firewallNAXSI*. Pengujian ini komputer *Attacker* melakukan beberapa macam serangan terhadap komputer *server*.

Ada beberapa tahapan pengujian serangan pada*webserver*, diantaranya:

1) Pengujian menggunakan *WhatWeb* dengan tujuan pengumpulan informasi *WebServer* yang dipakai dan jenis *CMS*.
2) Pengujian dengan *DirBuster* bertujuan untuk mencari file dan folder dalam *WebServer*.
3) Pengujian dengan *BruteForceAttack*, yaitu melakukan penyerangan pada*WebServer* untuk masuk secara paksa dengan mencari pengguna aktif pada *Wordpress* dengan *WPScan*kemudian menggunakan *Wordlist* acak sebagai *Password*.
4) Pengujian dengan *XSSAttack*, yaitu dengan mencoba injeksi skrip kode pada sisi *client*.
5) Pengujiaan *HTTPAttack* menggunakan *HOIC* dengan dengan tujuan untuk mengganggu fungsi normal *server*, sehingga membuat *WebServer* melambat dan tidak dapat diakses.

2.5. Tahap Analisis Hasil Uji Coba

Tahap ini dilakukan analisis dari hasil uji coba terhadap semua teknik serangan yang telah dilakukan oleh *attacker*. Analisa yang dilakukan mencakup performa *WebServer* sebelum diaktifkan *NAXSI* dan setelah diaktifkan *NAXSI*. Sehingga dari hasil analisa dapat diperoleh hasil dan kesimpulan apakah *NAXSI* dapat menghalau dan menjaga performa *WebServerNGINX* dari berbagai serangan yang telah diuji coba. Bagaimana respon *NAXSI*

terhadap serangan yang dilakukan, bagaimana aksi loging aktivitas yang dilakukan oleh *NAXSI* pada *WebServer*, serta teknik kontrol akses dengan *rules* yang dapat dilakukan dengan *NAXSI*.

3. HASIL DAN PEMBAHASAN

Tabel 2. Perbandingan sebelum dan setelah diaktifkan *NAXSI*

No	Jenis Serangan	Sebelum diaktifkan NAXSI	Setelah diaktifkan NAXSI
1	*WhatWeb*		**Terdeteksi Pesan** 2023/06/21 16:01:17 [error] 59636#59636: *2 NAXSI_EXLOG: ip=192.168.169.17&server=192.168.169.12&rid=cc18f780823d75166a0fb8e87d5ba4d4&uri=%2F&id=10000020&zone=HEADERS&var_name=user-agent&content=WhatWeb%2F0.5.5, client: 192.168.169.17, server: 192.168.169.12, request: "GET / HTTP/1.1", host: "192.168.169.12"
2	*HTTP Attack*		**Terdeteksi Pesan** 2023/06/27 16:56:41 [error] 5794#5794: *1542 NAXSI_EXLOG: ip=192.168.169.16&server=192.168.169.12&rid=4e8edbbc3dbdfff1c427&uri=%2F&id=10000022&zone=HEADERS&var_name=user-agent&content=HOIC%202, client: 192.168.169.16, server: 192.168.169.12, request: "GET / HTTP/1.1", host: "192.168.169.12"
3	*XSS Attack*	Serangan yang masuk ke *webserver* tidak dapat di identifikasi sebagai sebuah serangan berbahaya.	**Terdeteksi Pesan** 2023/06/29 23:30:27 [error] 10312#10312: *3 NAXSI_EXLOG: ip=192.168.169.17&server=192.168.169.12&rid=0f39712b130411cdf30c5659778bbdab&uri=%2F&id=10000023&zone=HEADERS&var_name=user-agent&content=curl%2F7.83.0, client: 192.168.169.17, server: 192.168.169.12, request: "GET / HTTP/1.1", host: "192.168.169.12"
4	*WPScan*		**Terdeteksi Pesan** 2023/06/25 02:37:38 [error] 8532#8532: *30 NAXSI_FMT: ip=192.168.169.16&server=192.168.169.12&uri=/&config=block&rid=4e804a524386ba9074e92b2c49251396&score0=$UWA&core0=16&zone=HEADER&id0=10000021&var_name0=user-agent, client: 192.168.169.16, server: 192.168.169.12, request: "GET /?author=10 HTTP/1.1", host: "192.168.169.12", referer: "http://192.168.169.12"

		Terdeteksi Pesan
5	*DirBuster*	2023/07/04 01:48:30 [error] 3529#3529: *14 NAXSI_EXLOG: ip=192.168.169.17&server=192.168.169.12&rid=8040e11daab7ecfb35c58c858df3b477&uri=%2F&id=10000024&zone=HEADERS&var_name=user-agent&content=DirBuster-1.0-RC1%20%28http%3A%2F%2Fwww.owasp.org%2Findex.php%2Fcategory%3AOWASP_DirBuster_Project%29, client: 192.168.169.17, server: 192.168.169.12, request: "GET /wp-login/ HTTP/1.1", host: "192.168.169.12"

Terlihat pada tabel perbandingan diatas bahwa keseluruhan serangan tidak dapat diidentifikasi sebelum diaktifkan *NAXSI*. Sebaliknya serangan dapat diidentifikasi secara keseluruhan setelah diaktifkan *NAXSI* pada *WebServer*.

Tabel 3. Tabel Perbandingan *NAXSI* sebelum dan setelah diaktifkan

NO	JenisSerangan	Sebelumdiaktifkan NAXSI	Setelahdiaktifkan*NAXSI*
1	WhatWeb	Intrusiyangdilakukan oleh*attacker*berhasil masukke*webserver*	Tertolak
2	HTTPAttack		Tertolak
3	XSSAttack		Tertolak
4	WPScan		Tertolak
5	DirBuster		Tertolak

Terlihat pada tabel 3 sebelum *NAXSI* diaktifkan, terlihat bahwa semua serangan yang lancarkan*attacker*kepada*webserver*berhasil masuk. Sebaliknya serangan dapat diblokir setelah *NAXSI* diaktifkan. *Rules* yang dibuat pada *NAXSI* dapat melakukan *log* serangan secara *RealTime* untuk menghalau berbagai serangan berbahaya sehingga permintaan mencurigakan yang sesuai dengan aturan yang dibuat akan menghasilkan peringatan *error* saat serangan berhasil diblokir.

4. KESIMPULAN

Berdasarkan dari penelitian yang telah dilakukan, *NAXSI* dan *LogAccessNginx* mampu bekerja sama dengan baik, sehingga ketika ada intrusi yang masuk, serangan dapat langsung diblokir secara *RealTime* dengan memberikan pesan *error* serta melakukan pemblokiran ketika terjadi serangan. *NAXSI* sendiri dapat mencegah serangan *WhatWeb* dan *DirBuster* dari *attacker* untuk mengumpulkan informasi terkait *server*. Sebelum mengakftifkan *NAXSI*, *attacker* dengan *WhatWeb* mendapatkan informasi *WebServer* yang digunakan dan *CMS* yang dipakai. Sementara

serangan menggunakan *DirBuster* berhasil medapatkan informasi file dan folder pada *WebServer*. Setelah mengaktifkan *NAXSI* durasi serangan menggunakan *DirBuster* menjadi lebih singkat karena serangan yang berhasil diblokir. *NAXSI* juga berhasil mencegah serangan *HTTPAttack* menggunakan *toolsHOIC*yang teridentifikasi dari nilai *CPUusage*, penggunaan Memori dan *NetworkHistory*yang tinggi. Sebelum diaktifkan *NAXSI*, nilai *CPUusage* adalah 98%-99%, penggunaan memori dari 85%-88% dan *NetworkHistory* Sending 49-61 KiB/s serta *Receiving*=803-1015 KiB/s. Sedangakan setelah diaktifkan *NAXSI* meskipun dalam keadaan diserang *WebServer* tetap dapat melayani *requestHTTP* dengan nilai *CPUusage* adalah 4%-7%, Penggunaan Memori dari 64%-67% Serta *NetworkHistorySending* dari 9-13 KiB/s dan Receiving dari 91-211 bytes/s yang mengindikasikan Nilai *CPUusage*, penggunaan Memori dan *NetworkHistory* pada kisaran nilai normal dan *NAXSI* mampu mencegah serangan *XSSattack* yang teridentifikasi dari tampilan *WebServer* yang berubah. Sebelum diaktifkan *NAXSI*, serangan ini membuat *Attacker* dapat menyuntikkan skrip berbahaya pada *server* sehingga menggangu kenyamanan *user*. Sedangakan setelah *NAXSI*diaktifkan *Attacker* tidak dapat menyuntikkan skrip berbahaya dengan keterangan *error* pada *ErrorLogNginx* yang mengindikasikan serangan berhasil diblokir oleh *NAXSI*. *NAXSI* berhasil mendeteksi dan memblokir serangan *WhatWeb*, *HTTPAttack*, *XSSAttack*, *WPScan* dan *DirBuster* dengan menampilkan *alertserror* yang tercatat pada *ErrorLogNginx*. Performa dari *NAXSI* sebagai *WebApplicationFirewall* sangat baik dalam melindungi *WebServerNGINX* dari ancaman *Attacker*, karena *NAXSI* dapat mendeteksi dan mencegah adanya serangan yang mencurigakan setelah menerapkan *rules* pengamanan.

DAFTAR PUSTAKA

[1] BPS (Badan Pusat Statistik), "Statistik Telekomunikasi Indonesia 2022," 2022. https://www.bps.go.id/publication/2023/08/31/131385d0253c6aae7c7a59fa/statistik-telekomunikasi-indonesia-2022.html (accessed Nov. 12, 2023).

[2] T. A. Cahyanto, H. Oktavianto, and A. W. Royan, "Analisis Dan Implementasi Honeypot Menggunakan Donaea Sebagai Penunjang Keamanan Jaringan," *J. Sist. Teknol. Inf. Indones.*, vol. 1, no. 2, pp. 86–92, 2016.

[3] B. M. S. Ery Setiyawan Jullev Atmaji, "Monitoring Keamanan Jaringan Komputer Menggunakan Network Intrussion Detection System (NIDS)," *Monit. Keamanan Jar. Komput. Menggunakan Netw. Intrussion Detect. Syst.*, pp. 118–122, 2016.

[4] P. Technology, "Attacks on web applications: 2018 in review," p. 14, 2019, [Online]. Available: https://www.ptsecurity.com/ww-en/analytics/web-application-attacks-2019/

[5] J. K. Anggraenni, "Simulasi Keamanan Pada Aplikasi Web Dengan Web Application Firewall," *Ilm. Komput.*, pp. 45–50, 2013.

[6] M. Hemmati and M. A. Hadavi, "Bypassing Web Application Firewalls Using Deep Reinforcement Learning**," *ISeCure*, vol. 14, no. 2, pp. 131–145, 2022, doi: 10.22042/isecure.2022.323140.744.

[7] X. D. Crystallography, *Nginx HTTP Server*. 2016.

[8] EFENDI MULYO, "Analisa Waf (Web Application Firewall) Menggunakan Nginx Terhadap Serangan Sql Injection," 2021, [Online]. Available: http://digilib.mercubuana.ac.id/

[9] R. Riska and H. Alamsyah, "Penerapan Sistem Keamanan Web Menggunakan Metode Web Aplication Firewall," *J. Amplif. J. Ilm. Bid. Tek. Elektro Dan Komput.*, vol. 11, no. 1, pp. 37–42, 2021, doi: 10.33369/jamplifier.v11i1.16683.

[10] I. Alfiani, A. Widjajarto, and A. Budiyono, "Implementasi dan Aanalisis Open Source Raptorwaf Pada Aplikasi Web Berdasarkan Standard PTES," *e-Proceeding Eng.*, vol. 8, no. 5, pp. 9243–9251, 2021.