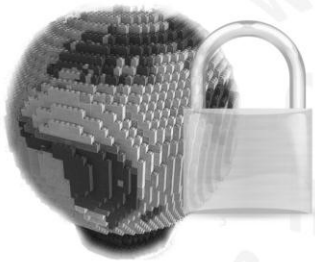
Kementerian Komunikasi dan Informatika
Republik Indonesia

Panduan Keamanan Web Server

2011

Direktorat Keamanan Informasi
Direktorat Jenderal Aplikasi Informatika
Kementerian Komunikasi dan Informatika

Panduan Keamanan Web Server



Direktorat Keamanan Informasi
Direktorat Jenderal Aplikasi Informatika
Kementerian Komunikasi dan Informatika
Republik Indonesia
2011

Diterbitkan oleh : Direktorat Keamanan Informasi
Direktorat Jenderal Aplikasi Informatika
Kementerian Komunikasi dan Informatika

Susunan Redaksi:
Pembina : Ashwin Sasongko
Pengarah : Bambang Heru Tjahjono
Penyusun : Aidil Chendramata
J. Maeran Sunarto
Intan Rahayu
Editor : Fitria Yuningsih
Cetakan Pertama, Desember 2011

Kata Pengantar

Saat ini website merupakan salah satu layanan informasi yang banyak diakses oleh pengguna internet di dunia. Sebagai salah satu layanan informasi maka perlu dibangun Website yang mampu menangani permintaan (*request*) dari banyak pengguna dengan baik (*reliable*). Pada dasarnya terdapat 4 (empat) elemen dasar dari sebuah Website yaitu: Browser, Server, URL, dan Pages. Web Server berisi Web Pages, yang di dalamnya mengandung informasi/dokumen yang ingin disebarluaskan atau diperlukan oleh para Pengguna. Web Server seringkali menjadi target dari berbagai jenis serangan baik yang sifatnya minor maupun major sehingga berakibat fatal. Hal ini dapat terjadi karena aspek keamanan web server kurang diperhatikan atau tidak diterapkan secara optimal, sehingga memungkinkan terjadinya resiko yang cukup signifikan.

Pada kenyataannya, adakalanya Web Server dikelola oleh individu yang memiliki pengalaman minim dalam pengelolaan suatu web server. Meskipun umumnya serangan yang terjadi hanya menimbulkan kesan negatif, memalukan atau ketidaknyamanan (seperti *defacing*), namun tidak tertutup kemungkinan penyerang dapat membuat masalah yang lebih serius atau bahkan sangat merugikan.

Sehubungan dengan itu, kiranya perlu bagi organisasi pemerintah, kalangan bisnis maupun pribadi yang menjalankan layanan website bagi publik,untuk memahami dan menerapkan hal-hal mendasar (*minimum requirement*) dalam pengamanan suatu webserver. Disamping itu bagi organisasi dengan struktur/Sumber Daya Manusia yang baik, dapat menerapkan program keamanan yang komprehensif guna meminimalisir resiko keamanan webserver-nya pada level yang dapat diterima.

Pedoman ini diharapkan dapat memberikan panduan bagi para

pengelola, individu atau unit yang bertanggung jawab dalam merencanakan dan mengelola Web Server (Publik). Dengan demikian perlindungan keamanan terhadap web server dapat diterapkan, baik sebagai minimum requirement maupun sebagai best practice, untuk meminimalisir insiden keamanan dalam pengelolaan Web server.

Jakarta, Desember 2011

Direktur Jenderal Aplikasi Informatika

Kementerian Komunikasi dan Informatika

Ashwin Sasongko

Daftar Isi

Ringkasan Eksekutif

Web server seringkali merupakan *host* yang paling banyak menjadi sasaran dan diserang di lingkup jaringan suatu organisasi. Akibatnya, mengamankan Web server dan infrastruktur jaringan yang mendukungnya merupakan hal yang perlu dilakukan. Ancaman keamanan secara spesifik terhadap Web server umumnya terbagi menjadi beberapa kategori:

1. Entitas *malicious* dapat mengeksploitasi *bug* perangkat lunak dalam Web server, sistem operasi web server, atau *active content* untuk memperoleh akses ilegal ke dalam Web server.
2. Serangan *Denial of Service* (DoS) yang diarahkan pada Web server atau infrastruktur jaringan pendukungnya, sehingga dapat menolak atau menghalangi para pengguna sah yang akan memanfaatkan layanannya.
3. Informasi sensitif pada Web server yang memungkinkan untuk dibaca atau dimodifikasi tanpa otorisasi.
4. Informasi sensitif di database penyangga yang digunakan untuk mendukung elemen interaktif dari suatu aplikasi Web memungkinkan untuk dibobol melalui serangan *command injection,* misalnya injeksi *Structured Query Language* [SQL], injeksi *Lightweight Directory Access Protocol* [LDAP], *cross-site scripting* [XSS].
5. Informasi sensitif yang ditransmisikan tanpa dienkripsi antara Web server dan Browser yang dapat disadap dan terbaca dengan jelas.
6. Informasi pada Web server yang dapat diubah untuk tujuan jahat. *Defacement* (penggantian tampilan) situs Web merupakan contoh yang umumnya dilaporkan untuk ancaman ini.
7. Entitas *malicious* yang berhasil mendapatkan akses ilegal

terhadap sumber daya di tempat lain dalam jaringan organisasi melalui suatu serangan terhadap Web server.

8. Entitas *malicious* yang menyerang organisasi inernal setelah membobol suatu *host* Web server. Serangan tersebut dapat dilancarkan secara langsung, misalnya dari *host* yang bobol terhadap suatu server internal atau secara tidak langsung misalnya dengan menempatkan konten *malicious* pada Web server yang bobol yang berusaha mengeksploitasi kerawanan dalam browser Web dari para pengguna yang mengunjungi situs tersebut.
9. Server yang dapat digunakan sebagai suatu titik distribusi perangkat serangan, pornografi, atau lunak yang dicopy secara ilegal.

Tipe Serangan Tak Langsung terhadap Web server bertujuan mendapatkan informasi dari para penggunanya. Sebagai contoh, pengguna dibujuk atau secara otomatis diarahkan untuk mengunjungi situs Web *malicious* tampak tidak mencurigakan (karena menyerupai website aslinya, dsb). Melalui aktivitas yang dilakukan pengguna pada website *malicious* tersebut, didapat informasi penting pengguna sepeti *username*, *password*, dll. Hasil perolehan tersebut dapat disalahgunakan untuk mengakses website aslinya secara sah atau dengan kata lain merupakan salah satu upaya pencurian identitas. Serangan yang berhasil mampu membobol rahasia sumber daya situs Web atau merusak gambaran/profil mengenai organisasi. Serangan tidak langsung ini dapat berbentuk:

1. *Phishing*, dimana para penyerang menggunakan *social engineering* (rekayasa sosial) untuk memperdaya para pengguna agar melakukan *logging* ke suatu situs palsu.
2. *Pharming*, dimana server DNS atau *file host* para pengguna dibobol sehingga para pengguna diarahkan ke suatu situs *malicious* pengganti situs yang sah.

Dokumen ini dimaksudkan untuk memberikan asistensi pada organisasi dalam instalasi, melakukan konfigurasi, dan mengelola Web server yang aman. Pedoman ini direkomendasikan bagi institusi pemerintah untuk mengelola suatu Web server yang aman. Beberapa rekomendasi yang dapat dilakukan antara lain adalah :

1. **Organisasi sebaiknya merencanakan dengan cermat, termasuk masalah penentuan personil dan keterampilan yang dibutuhkan dalam mengamankan suatu Web server .**

   Hal-hal yang perlu diperhatikan dalam suatu perencanaan untuk menghasilkan tindakan yang efektif adalah sebagai berikut :

   a) Personil yang dibutuhkan, misalnya, para administrator Web server dan sistem, Webmaster, administrator jaringan, pejabat keamanan sistem informasi [ISSO] dan lain-lain.

   b) Keterampilan dan pelatihan yang dibutuhkan oleh personil yang ditugaskan.

   c) Persyaratan individu, yaitu spesifikasi yang disyaratkan dari tipe personil tertentu dan kelompok kerja yang dibentuk untuk menangani aspek keamanan.

2. **Organisasi sebaiknya mengimplementasikan praktek dan kontrol manajemen keamanan yang tepat ketika memelihara dan mengoperasikan suatu Web server yang aman.**

   Untuk memastikan keamanan dari suatu Web server dan infrastruktur jaringan pendukung, hal-hal berikut semestinya diimplementasikan:

   a) Kebijakan keamanan sistem informasi lingkup organisasi

   b) Kontrol dan manajemen konfigurasi/perubahan

   c) Penilaian dan manajemen resiko

   d) Konfigurasi perangkat lunak yang distandarkan yang memenuhi kebijakan keamanan sistem informasi

e) Kesadaran dan pelatihan keamanan

f) Rencana *contingency*, kontinuitas operasi, dan perencanaan pemulihan bencana

g) Sertifikasi dan akreditasi.

2. Organisasi sebaiknya memastikan bahwa sistem operasi Web server dipersiapkan, dikonfigurasikan dan dikelola untuk memenuhi persyaratan keamanan dari organisasi tersebut.

Langkah pertama dalam mengamankan suatu Web server adalah mengamankan sistem operasinya. Mengamankan sistem operasi umumnya akan menyertakan langkah-langkah berikut:

a) Melakukan *patch* dan *upgrade* sistem operasi

b) Menghilangkan atau me-non-aktifkan aplikasi dan layanan yang tidak diperlukan

c) Mengkonfigurasikan otentikasi pengguna sistem operasi

d) Mengkonfigurasikan pengendalian sumber daya.

e) Menginstalkan dan mengkonfigurasikan kendali keamanan tambahan

f) Melakukan testing keamanan terhadap sistem operasi.

3. Organisasi sebaiknya memastikan bahwa aplikasi Web server diinstal, dikonfigurasi dan dikelola sesuai kebutuhan keamanan organisasi.

Mengamankan aplikasi Web server biasanya akan mencakup langkah-langkah berikut:

a) Melakukan *patch* dan *upgrade* aplikasi Web server

b) Menghilangkan atau menon-aktifkan layanan, aplikasi dan contoh konten yang tidak perlu

c) Mengkonfigurasikan otentikasi pengguna dan kendali akses

Web server

d) Mengkonfigurasikan *access control* sumber daya Web server

e) Melakukan tes keamanan aplikasi Web server dan konten Web

4. Organisasi sebaiknya mengambil langkah-langkah untuk memastikan bahwa hanya konten yang layak yang dipublikasikan pada suatu situs Web dan bahwa konten sudah diproteksi dengan baik.

Beberapa informasi yang seharusnya tidak dipublikasikan adalah sebagai berikut :

a) Informasi berklasifikasi rahasia dan sangat rahasia

b) Informasi tentang komposisi atau penyiapan material berbahaya atau beracun

c) Informasi sensitif terkait keamanan dalam negeri

d) Catatan medis

e) Usaha perlindungan terhadap keamanan detil fisik dan informasi dari suatu organisasi

f) Detil tentang infrastruktur jaringan dan sistem informasi organisasi (misalnya, jangkauan address, konvensi penamaan, jumlah akses)

g) Informasi yang menspesifikasikan atau mengimplikasikan kerawanan keamanan fisik

h) Detil rencana, peta, diagram, foto udara, gambar arsitektur dari gedung, properti atau instalasi organisasi

i) Informasi sensitif apapun tentang individu, seperti informasi yang dapat mengidentifikasi secara personal (*personally identifiable information* [PII]), yang dapat menjadi subyek hukum.

5. Organisasi sebaiknya memastikan langkah-langkah tepat yang diambil untuk melindungi konten Web server dari akses atau modifikasi yang tidak sah.

Langkah-langkah yang diambil untuk melindungi konten web tersebut antara lain :

a) Menginstal atau mengaktifkan layanan yang memang diperlukan.

b) Menempatkan konten Web pada suatu *hard drive* atau partisi *log*ik tertentu.

c) Membatasi *upload* ke direktori yang tidak dapat dibaca oleh Web server.

d) Menentukan suatu direktori tunggal untuk semua *script* atau program eksternal yang dieksekusi sebagai bagian dari konten Web.

e) Menon-aktifkan penggunaan *hard /symbolic link*.

f) Menentukan suatu matriks akses konten Web lengkap yang mengidentifikasi folder dan *file* mana dalam direktori dokumen Web server yang bebas atau dibatasi aksesnya (dan oleh siapa).

g) Menon-aktifkan listing direktori.

h) Menggunakan otentikasi, tandatangan digital, dan mekanisme kriptografi yang diperlukan

i) Menggunakan *host-based* sistem deteksi gangguan (*intrusion detection system* [IDS]) dan/atau pengecek integritas *file* untuk mendeteksi gangguan atau memverifikasi konten Web.

j) Memproteksi setiap server penyangga (misalnya server database, server direktori) dari serangan injeksi perintah (*command injection attack*) baik dari Web server maupun server penyangga itu sendiri.

6. **Organisasi sebaiknya menggunakan *active content* secara bijaksana untuk menyeimbangkan antara manfaat yang diperoleh dengan resiko yang menyertainya.**

   Sebagian besar situs Web pada generasi awal memberikan informasi statis yang diletakkan pada server, umumnya berbentuk dokumen berbasis teks. Dalam perkembangannya, elemen interaktif diperkenalkan kepada para pengguna tentang cara baru berinteraksi dengan suatu situs Web. Sayangnya, elemen interaktif tersebut memperkenalkan kerentanan baru yang terkait dengan Web karena elemen-elemen tersebut melibatkan kode eksekusi dinamis baik pada Web server ataupun klien yang menggunakan input sangat besar, mulai dari parameter *Universal Resource Locator* (URL) hingga konten *Hypertext Transfer Protocol* (HTTP) POST maupun konten XML berbentuk pesan-pesan layanan Web. Tekno*log*i *active content* yang berbeda memiliki kerentanan terkait yang berbeda, dan resikonya harus diseimbangkan dengan manfaatnya.

7. **Organisasi harus menggunakan tekno*log*i otentikasi dan enkripsi untuk melindungi data sensitif tertentu.**

   Seringkali diperlukan tekno*log*i untuk mengidentifikasikan dan membuktikan keaslian para pengguna dengan *privilege* yang berbeda untuk mengakses informasi. Beberapa dari tekno*log*i ini berbasiskan fungsi kriptografi yang dapat menyediakan suatu jalur terenkripsi antara klien browser Web dan Web server yang mendukung enkripsi.

   Sekalipun dengan suatu jalur terenkripsi dan mekanisme otentikasi, mungkin saja para penyerang berupaya mengakses situs melalui suatu serangan *brute force.*Teknik otentikasi yang tidak tepat dapat menyebabkan para penyerang mengumpulkan *username* yang sah atau kemungkinan besar mendapatkan akses ke situs Web. Mekanisme otentikasi yang baik dapat pula melindungi terhadap serangan *phishing* dan *pharming.* Oleh

karena itu, harus dimplementasikan otentikasi dengan tingkatan yang tepat berdasarkan pada sensitifitas para pengguna dan konten Web server.

8. Organisasi sebaiknya mengembangkan keamanan infrastruktur jaringan untuk membantu melindungi server Web-nya.

Insfrastruktur jaringan (misalnya, *firewall, router, IDS*) yang mendukung Web server memainkan suatu peran penting dalam hal keamanan dari Web server. Pada sebagian besar konfigurasi, infrastruktur jaringan akan menjadi garis terdepan pertahanan antara suatu Web server dan Internet. Desain jaringan saja tidak dapat melindungi suatu Web server. Frekuensi, kecanggihan dan ragam serangan Web server yang dilakukan kini mendukung gagasan bahwa keamanan Web server harus diimplementasikan melalui mekanisme proteksi yang berlapis dan terpelihara.

9. Organisasi sebaiknya berkomitmen terhadap proses pemeliharaan keamanan Web server yang dilakukan secara rutin untuk memastikan keamanan berkelanjutan.

Memelihara suatu Web server yang aman memerlukan upaya yang konsisten, sumber daya, dan kewaspadaan dari suatu organisasi. Memelihara keamanan dari suatu Web server biasanya akan melibatkan langkah-langkah berikut:

a) Mengkonfigurasi, memproteksi, dan menganalisa *file log*

b) Membuat *backup* informasi kritis secara rutin

c) Menetapkan dan menjalankan prosedur untuk pemulihan dari kebobolan

d) Pengetesan dan penerapan *patch* secara tepat waktu

e) Pengetesan keamanan secara periodik.

1. Pendahuluan

1.1. Latar Belakang

World Wide Web merupakan salah satu cara yang paling penting bagi suatu organisasi untuk mempublikasikan informasi, berinteraksi dengan pengguna Internet, dan membangun keberadaan e-commerce/e-government. Namun demikian, jika suatu organisasi tidak tepat dalam mengkonfigurasikan dan mengoperasikan situs Web-nya, kemungkinan situs Web tersebut akan rawan terhadap berbagai macam ancaman keamanan.

Situs Web yang bobol dapat dijadikan sebagai pintu masuk secara ilegal ke dalam jaringan internal organisasi. Organisasi dapat kehilangan data sensitif atau bahkan sistem jaringan komputernya menjadi tidak berfungsi. Contohnya adalah Serangan *denial of service* (DoS) dapat membuat para pengguna sulit atau bahkan tidak bisa mengakses situs Web suatu organisasi.

Terdapat tiga pokok persoalan keamanan yang utama berkaitan dengan operasi dari suatu situs Web yang dapat diakses secara publik:

1. Kesalahan konfigurasi atau operasi lain yang tidak sesuai pada Web server, yang mungkin berakibat pada penyingkapan atau perubahan terhadap informasi yang sensitif.

2. Kerawanan-kerawanan dalam Web server sehingga dapat terjadi suksesnya para penyerang membobol keamanan server dan *host* lain dalam jaringan komputer organisasi, dengan mengambil tindakan-tindakan seperti berikut ini:

    a) Merusak tampilan situs (*defacing) Web* atau mempengaruhi integritas informasi

    b) Mengeksekusi perintah atau program tidak sah pada sistem operasi *host*, termasuk suatu perintah atau program yang

telah diinstal oleh para penyusup sebelumnya

c) Memperoleh akses yang tidak sah ke sumber daya di suatu tempat dalam jaringan komputer organisasi

d) Meluncurkan serangan dari Web server terhadap situs eksternal, dengan menyembunyikan identitas para penyusup dan mungkin menjadikan organisasi menjadi pihak yang bertanggung jawab atas kerusakan

e) Menggunakan server sebagai pendistribusi *perangkat lunak* yang digandakan secara ilegal, *tools* serangan, atau pornografi, dan mungkin menjadikan organisasi tersebut menjadi pihak yang bertanggung jawab atas dampak yang terjadi

f) Menggunakan *server* untuk mengirimkan serangan terhadap klien-klien *Web* yang rentan untuk membobol mereka.

3. Mekanisme pertahanan yang tidak memadai atau tidak tersedia pada Web server untuk mencegah jenis-jenis serangan tertentu, seperti serangan DoS yang dapat mengacaukan ketersediaan Web server dan mencegah para pengguna yang sah untuk mengakses situs Web ketika dibutuhkan.

Selain itu, seiring membaiknya keamanan instalasi jaringan dan server, aplikasi perangkat lunak dan *script* yang ditulis secara sembarang telah menjadi sasaran penyerangan yang akan memungkinkan para penyerang untuk membobol keamanan Web server atau mengumpulkan data dari *back end* database.

Diperlukan sejumlah langkah untuk menjamin keamanan suatu Web server. Sebagai suatu prasyarat, adanya kebijakan keamanan yang dimiliki organisasi merupakan hal yang mendasar. Langkah-langkah yang dapat dilakukan dalam konteks kebijakan keamanan web server adalah sebagai berikut :

Langkah 1 : Menginstal, mengkonfigurasi dan mengamankan

sistem operasi web server

Langkah 2 : Menginstal, mengkonfigurasi dan mengamankan perangkat lunak Web server

Langkah 3 : Menggunakan mekanisme pertahanan jaringan yang tepat (misalnya, firewall, packet filtering router dan proxy)

Langkah 4 : Memastikan bahwa aplikasi apapun yang dikembangkan secara khusus untuk Web server diprogram sesuai dengan keamanan pemrograman.

Langkah 5 : Memelihara konfigurasi keamanan melalui aplikasi *patch* dan *upgrade*, pengujian keamanan, pemonitoran *log*, serta *backup* data dan sistem operasi web server yang tepat

Langkah 6 : Menggunakan, mempublikasikan, dan melindungi informasi dan data dengan cara yang hati-hati dan sistematis

Langkah 7 : Melakukan administrasi dan proses pemeliharaan yang aman (termasuk *updating server*/aplikasi dan pemeriksaan *log*)

Langkah 8 : Melakukan *scan* kerawanan di awal implementasi dan secara periodik dari setiap Web server dan infrastruktur jaringan pendukung (misalnya, firewall, router).

1.2. Tujuan dan Ruang Lingkup

Tujuan dari Pedoman tentang Pengamanan Web Server adalah merekomendasikan aspek keamanan untuk perancangan, implementasi, dan pengoperasian Web server yang dapat diakses secara publik. Aspek yang direkomendasikan dalam pedoman ini dirancang untuk membantu mengurangi resiko yang berkaitan dengan Web server dan mengetengahkan prinsip umum yang

berlaku untuk semua sistem.

Pedoman ini direkomendasikan bagi departemen dan lembaga pemerintah, namun dapat juga digunakan di sektor swasta dan organisasi yang tertarik dalam meningkatkan keamanan sistem Web server dan guna mengurangi jumlah dan frekuensi insiden keamanan yang terkait dengan Web. Dokumen ini tidak mencakup aspek-aspek lain yang terkait dengan mengamankan suatu Web server berikut:

1. Mengamankan jenis lain dari server jaringan
2. Pertimbangan keamanan yang terkait dengan perangkat lunak klien Web browser
3. Pertimbangan khusus untuk situs Web dengan lalu lintas padat dan *host* berganda
4. Mengamankan server penyangga yang mungkin mendukung Web server (misalnya, server database, server *file*)
5. Layanan selain HTTP dan HTTPS
6. SOAP-style Web Services
7. Proteksi kekayaan intelektual

1.3. Pembaca dan Asumsi

Pedoman ini bersifat teknis dan dapat menjadi acuan bagi pembaca yang terdiri dari :

1. Para Sistem Engineer, ketika merancang dan mengimplementasikan Web server.
2. Para administrator Web dan sistem, ketika mengelola, *patching*, mengamankan, atau meng-*upgrade* Web server.
3. Webmaster, ketika membuat dan mengelola konten Web.
4. Konsultan keamanan, ketika melakukan audit keamanan untuk

menentukan postur keamanan sistem informasi (SI).

5. Para Program manager dan petugas keamanan Tekno*log*i Informasi (*Information Technology Security Officer*), untuk memastikan bahwa tindakan pengamanan yang layak telah diperhitungkan untuk semua fase siklus hidup sistem.

Pedoman ini mengasumsikan bahwa para pembaca memiliki keahlian minimal tentang sistem operasi dan jaringan komputer serta Web server. Karena sifat ancaman dan kerentanan Web server yang dapat berubah, para pembaca diharapkan memanfaatkan sumber lain (termasuk yang terdapat pada lampiran Pedoman ini) untuk informasi yang lebih baru dan detil.

1.4. Termino*log*i

1. **Address Resolution Protocol (ARP) —** Suatu protokol yang digunakan untuk memperoleh alamat fisik suatu *node.* Suatu stasiun klien melakukan *broadcast* suatu permintaan ARP kedalam suatu jaringan dengan alamat Internet Protocol (IP) dari *node* target yang diinginkannya untuk berkomunikasi, dan dengan alamat tersebut *node* menanggapi dengan mengirimkan kembali alamat fisiknya supaya paket itu dapat ditransmisikan padanya.
2. **Administrator Jaringan —** Seseorang yang mengelola local area network (LAN) dalam lingkup suatu organisasi. Tanggungjawabnya meliputi memastikan keamanan jaringan, menginstalasi aplikasi baru, mendistribusikan *upgrades* perangkat lunak, memonitor aktifitas harian, memberlakukan persetujuan lisensi, membangun suatu program manajemen penyimpanan, dan menyediakan *backup* rutin.
3. **Administrator Sistem —** Seseorang yang mengelola suatu sistem komputer, termasuk sistem operasi dan aplikasinya. Tanggung jawab seorang administrator sistem mirip dengan seorang administrator jaringan.

4. **AdministratorWeb Server —** Para administrator Web server merupakan arsitek sistem yang bertanggung jawab atas keseluruhan rancangan, implementasi, dan pemeliharaan dari Web server. Mereka bisa saja bertanggung jawab atas konten Web yang biasanya merupakan tanggung jawab dari Webmaster.

5. **Demilitarized Zone (DMZ) —** Suatu *host* atau segmen jaringan yang disisipkan sebagai suatu "zona netral" diantara jaringan lokal dari organisasi dan Internet .

6. **Hotfix —** Istilah Microsoft untuk "*patch*."

7. ***Host* —** Hampir semua jenis komputer apapun, termasuk *mainframe* terpusat yang merupakan suatu *host* bagi terminal-terminalnya, suatu server yang menjadi *host* untuk klien-kliennya, atau suatu desktop personal computer (PC) yang menjadi *host* untuk *peripheral* nya.

8. **Kerawanan (Vulnerability) —** Penyingkapan keamanan dalam suatu sistem operasi, perangkat lunak sistem atau komponen perangkat lunak aplikasi lainnya. Masing-masing kerawanan jika dieksploitasi secara potensial dapat membobol sistem atau jaringan komputer.

9. **Mandatory Access Control —** Suatu upaya untuk membatasi akses ke sumber daya sistem berbasiskan pada sensitivitas (biasanya ditunjukkan dengan suatu label) dari informasi yang terkandung dalam sumber daya sistem dan otorisasi resmi (yaitu *clearance*) dari para pengguna untuk mengakses informasi dengan sensitivitas semacam itu.

10. **Nonce —** Suatu nilai yang dibangkitkan secara acak yang digunakan untuk mengalahkan serangan "playback" dalam protokol komunikasi.

11. **Patch —** Suatu "pekerjaan perbaikan" untuk sebagian pemrograman; juga dikenal sebagai suatu "fix". Suatu *patch* merupakan solusi segera yang disediakan bagi para pengguna,

yang dapat di-*download* dari situs Web pembuat perangkat lunak.

12. **Pembangkit Konten —** Suatu program pada suatu Web server yang akan menghasilkan secara dinamis halaman *Hyper Text Markup Language* (HTML) untuk para pengguna. Pembangkit konten dapat berada pada kisaran mulai dari *script Common Gateway Interface* (CGI) sederhana yang dieksekusi oleh Web server tersebut hingga ke server aplikasi Java EE atau .NET dimana sebagian besar halaman HTML yang diberikan dihasilkan secara dinamis.

13. **Pharming —** Menggunakan cara teknis untuk mengarahkan kembali para pengguna untuk mengakses suatu situs Web palsu yang berkedok sebagai situs yang sah dan mendapatkan informasi pengguna yang diinginkan, seperti data akun *login*.

14. **Proxy —** *Proxy* adalah suatu aplikasi yang "memutuskan" koneksi antara klien dan server. *Proxy* menerima tipe-tipe tertentu dari lalu lintas yang memasuki atau meninggalkan suatu jaringan, memprosesnya dan meneruskannya.

15. **Service Pack —** Istilah Microsoft untuk suatu kumpulan *patches* yang diintegrasikan kedalam suatu *single large update.*

16. **Sistem Operasi —** Perangkat lunak "aplikasi kontrol utama" yang menjalankan komputer, merupakan program pertama yang dijalankan ketika komputer dinyalakan. Kernel sebagai komponen intinya tersimpan di memori. Sistem operasi mengatur standar untuk semua program aplikasi (semacam Web server) yang bekerja di dalam komputer. Aplikasi berkomunikasi dengan sistem operasi untuk sebagian besar *interface* pengguna dan operasi manajemen *file*.

17. **Virtualisasi —** Semacam alokasi penggunaan perangkat keras secara virtual untuk menjalankan suatu komputasi sehingga lebih dari satu sistem operasi dapat berjalan pada satu komputer.

18. **Web Server —** Suatu komputer yang menyediakan layanan World Wide Web (WWW) pada Internet, yang mencakup perangkat keras, sistem operasi, perangkat lunak Web server, dan konten situs Web site (halaman Web). Jika Web server tersebut digunakan secara internal dan tidak oleh publik, maka dikenal sebagai suatu "*server* intranet".
19. **Webmaster —** Seseorang yang bertanggung jawab atas implementasi dari suatu Web. Webmasters harus menguasai HTML dan satu atau lebih pembuatan *script* serta pembuatan *interface*, semacam JavaScript, CSS, Perl, dll. Webmaster bisa sekaligus merangkap sebagai web administrator.

1.5. Sistematika Penulisan

Sistematika Penulisan Dokumen ini adalah sebagai berikut :

Bab 1 Latar belakang permasalahan keamanan Web server, termino*log*i, tujuan dan ruang lingkup, pembaca dan asumsi serta sistematika penulisan.

Bab 2 Perencanaan dan pengelolaan dari Web server.

Bab 3 Tinjauan umum mengenai pemilihan dan keamanan sistem operasi yang mendasari suatu Web server.

Bab 4 Instalasi dan konfigurasi suatu Web server secara aman.

Bab 5 Keamanan konten Web.

Bab 6 Teknik otentikasi dan enkripsi pada web server.

Bab 7 Proteksi suatu Web server melalui infrastruktur jaringan pendukungnya.

Bab 8 Dasar dari pengelolaan yang aman atas suatu Web server sehari-hari.

Bab 9 Pemberian *checklist* untuk poin keamanan web server.

2. Perencanaan dan Pengelolaan Web Server

Aspek yang paling penting dalam mempersiapkan suatu Web server yang aman adalah perencanaan yang matang sebelum instalasi, konfigurasi, dan *deployment*. Perencanaan ini akan memastikan bahwa Web server dalam keadaan seaman mungkin dan mematuhi seluruh kebijakan-kebijakan organisasi yang relevan. Banyak permasalahan keamanan dan kinerja Web server yang dapat ditelusuri mengarah pada kurangnya perencanaan atau kontrol manajemen.

2.1. Perencanaan Instalasi dan *Deployment*

Pada tahap perencanaan suatu Web server, hal-hal yang harus dipertimbangkan adalah sebagai berikut:

1. Mengidentifikasikan kegunaan dari Web server
    a) Kategori informasi yang akan disimpan pada Web server
    b) Kategori informasi yang akan diproses atau ditransmisikan melalui Web server
    c) Persyaratan keamanan untuk setiap kategori informasi
    d) Penentuan apakah informasi diambil dari atau disimpan pada *host* lain (misal, *back-end database, mail server*)
    e) Persyaratan keamanan untuk *host* lain yang terlibat (misal, *back-end database*, *directory server, mail server, proxy server*)
    f) Penentuan layanan-layanan lain yang akan disediakan oleh Web server. Namun pada umumnya, pengalokasian *host* hanya sebagai Web server merupakan pilihan paling aman.

g) Persyaratan keamanan untuk layanan-layanan tambahan tersebut.

h) Persyaratan untuk operasional layanan yang disediakan oleh Web server, seperti hal-hal yang dispesifikasikan dalam rencana operasi dan rencana pemulihan dari bencana (*disaster recovery*).

i) Penentuan posisi Web server dalam topo*log*i jaringan.

2. Mengidentifikasikan layanan-layanan jaringan yang akan disediakan pada Web server, seperti yang diberikan melalui protokol-protokol berikut:

   a) HTTP

   b) HTTPS

   c) Internet *Caching Protocol* (ICP)

   d) *Hyper Text Caching Protocol* (HTCP)

   e) *Web Cache Coordination Protocol* (WCCP)

   f) SOCKS

   g) Layanan-layanan database (misalnya *Open Database Connectivity* [ODBC]).

3. Mengidentifikasikan perangkat lunak layanan jaringan, baik klien maupun server, untuk dipasang pada Web server dan server pendukung lainnya.

4. Mengidentifikasikan para pengguna atau kategori para pengguna dari Web server dan *host* pendukung.

5. Menenentukan hak-hak istimewa (*privilege*) yang akan dimiliki setiap kategori pengguna pada Web server dan *host* pendukung.

6. Menentukan bagaimana Web server akan dioperasikan (misalnya secara lokal, secara remote dari jaringan internal maupun dari jaringan eksternal).

7. Memutuskan kapan dan bagaimana para pengguna akan diotentikasi dan bagaimana data terotentikasi tersebut akan dilindungi.
8. Menentukan bagaimana akses yang sesuai terhadap sumber informasi akan diberlakukan.
9. Menentukan aplikasi Web server mana yang memenuhi keperluan organisasi. Beberapa hal yang perlu dipertimbangkan:
    a) Biaya
    b) Kesesuaian dengan infrastruktur yang sudah ada
    c) Pengetahuan dari para pegawai yang sudah ada
    d) Hubungan manufaktur (*manufakturer)* yang sudah ada
    e) Sejarah kerawanan di masa lalu
    f) Fungsionalitas
10. Mengetahui beberapa pihlihan perangkat lunak maupun perangkat keras Web server pada tahap perencanaan

Pilihan aplikasi Web server dapat menentukan pemilihan OS, dimana OS Web server perlu menyediakan hal-hal berikut :

1. Kemampuan untuk membatasi kegiatan administratif atau tingkatan *root* hanya untuk pengguna yang sah
2. Kemampuan untuk mengendalikan akses terhadap data pada server
3. Kemampuan untuk menon-aktifkan layanan jaringan yang tidak diperlukan yang mungkin dibangun dalam OS atau perangkat lunak server
4. Kemampuan untuk mengendalikan akses ke berbagai bentuk program yang dapat dieksekusi, seperti *Common Gateway Interface (CGI) script* dan *server plug-in* dalam kasus Web server

5. Kemampuan untuk merekam aktifitas server yang tepat untuk mendeteksi penyerangan dan penyerangan yang diupayakan.
6. Ketentuan dari kemampuan suatu *firewall* yang berbasis *host.*

Sebagai tambahan, organisasi harus mempertimbangkan ketersediaan staf yang terlatih dan berpengalaman untuk mengelola server dan produk-produk server.

Lokasi penyimpanan Web server juga merupakan hal yang penting dan perlu diamankan secara fisik. Dalam merencanakan lokasi Web server maka perlu dipertimbangkan hal-hal sebagai berikut :

1. Mekanisme perlindungan keamanan fisik yang tepat, meliputi:
    a) Kunci-kunci
    b) Akses *card reader*
    c) Penjaga keamanan
    d) ID fisik (misal, sensor gerak, kamera).
2. Kontrol lingkungan yang tepat terutama mengenai kelembaban dan suhu yang sesuai.
3. Ketersediaan sumber listrik cadangan dan indentifikasi berapa lama sumber tersebut dapat menyediakan listrik.
4. Untuk keperluan tertentu, identifikasi ketersediaan koneksi internet cadangan minimal dari dua penyedia layanan internet (ISP) yang berbeda.
5. Jika Web server berlokasi di daerah rawan bencana alam, perlu ditambahkan prosedur pengamanan supaya lebih tahan terhadap bencana. Alternatif lain yaitu meyediakan *backup* di lokasi lain yang aman dari potensi bencana.

2.2. Struktur Manajemen Keamanan

Karena keamanan Web server sangat berkaitan dengan struktur keamanan sistem informasi umum organisasi, sejumlah personel

Tekno*log*i Informasi (TI) dan keamanan sistem akan terlibat dalam perencanaan, implementasi, dan administrasi Web server. Berikut adalah struktur manajemen keamanan serta identifikasi peran dan tanggung jawab yang berkaitan dengan keamanan Webserver.

1. Manajemen TI Senior

Manajemen TI Senior (*Chief Information Officer* [CIO]) bertanggung jawab memastikan bahwa struktur keamanan organisasi cukup memadai. Manajemen TI Senior menyediakan pedoman-pedoman petunjuk dan laporan untuk perlindungan sistem informasi bagi keseluruhan organisasi. Manajemen TI Senior/CIO bertanggung jawab atas aktivitas yang berkaitan dengan Web server, antara lain:

a) Koordinasi pengembangan dan pemeliharaan kebijakan keamanan informasi, standar dan prosedur organisasi.

b) Koordinasi pengembangan dan pemeliharaan kontrol perubahan dan prosedur manajemen organisasi.

c) Memastikan pembangunan, dan pemenuhan yang sesuai dengan, kebijakan keamanan yang tetap untuk organisasi.

d) Mengkoordinasikan dengan manajemen yang lebih tinggi, urusan umum, dan personil lain yang terlibat untuk menghasilkan suatu kebijakan dan proses formal untuk mempublikasikan informasi ke situs Web dan memastikan kebijakan ini diberlakukan.

2. Manajer Program Keamanan Sistem Informasi

Para Manajer Program Keamanan Sistem Informasi (*Information Systems Security Program Managers* [ISSPM]) bertanggung jawab mengawasi implementasi dan pemenuhan standar, peraturan, dan regulasi yang dispesifikasikan dalam kebijakan keamanan organisasi. ISSPM bertanggung jawab terhadap aktivitas yang berkaitan dengan Web server berikut:

a) Memastikan bahwa prosedur keamanan dibuat dan diimplementasikan

b) Memastikan bahwa kebijakan, standar, dan persyaratan keamanan diikuti

c) Memastikan bahwa seluruh sistem yang penting diidentifikasi dan terdapat perencanaan kondisi darurat (*contigency planning*) serta rencana pemulihan dari bencana (*disaster recovery plan)*.

d) Memastikan bahwa sistem-sistem yang kritis diidentifikasi dan dijadwalkan untuk pengujian keamanan secara berkala mengacu pada persyaratan kebijakan keamanan dari setiap sistem.

3. Petugas Keamanan Sistem Informasi

Para Petugas Keamanan Sistem Informasi (*the Information Systems Secutity Officers* [ISSO]) bertanggung jawab dalam mengamati keseluruhan aspek dari keamanan informasi, memastikan bahwa praktek keamanan informasi organisasi mengikuti kebijakan, standar, dan prosedur organisasi. Para ISSO bertanggung jawab terhadap seluruh aktivitas yang berkaitan dengan Web server berikut:

a) Membuat standar dan prosedur keamanan internal untuk Web server dan infrastruktur jaringan pendukung.

b) Bekerjasama dalam pengembangan dan implementasi perangkat, mekanisme, dan teknik mitigasi keamanan.

c) Memelihara profil konfigurasi standar Web server dan infrastruktur jaringan pendukung yang dikendalikan oleh organisasi, termasuk, tetapi tidak terbatas pada, OS, firewall, router, dan aplikasi Web server.

d) Memelihara integritas operasional sistem dengan melakukan uji-uji keamanan dan memastikan bahwa

personil TI yang ditunjuk melakukan pengujian sistem yang terjadwal.

e) Menerima dan menindaklanjuti laporan kejadian terkait insiden keamanan baik dari internal maupun eksternal, untuk kemudian berkoordinasi dengan kontak yang tertera pada Lampiran

4. Administrator Web Server dan Jaringan

Para administrator Web server merupakan arsitek sistem yang bertanggung jawab terhadap keseluruhan rancangan, implementasi, dan pemeliharaan suatu Web server. Para administrator jaringan bertanggung jawab terhadap keseluruhan desain, implementasi, dan pemeliharaan suatu jaringan. Administrator bertanggung jawab terhadap aktivitas yang berkaitan dengan Web server berikut:

a) Menginstal dan mengkonfigurasi sistem dalam kerangka pemenuhan kebijakan keamanan organisasi dan sistem standar dan konfigurasi jaringan.

b) Memelihara sistem dengan cara yang aman, termasuk *backup* berkali-kali dan aplikasi *patch* tepat waktu.

c) Memonitor integritas sistem, tingkat perlindungan, dan kejadian-kejadian yang berkaitan dengan keamanan.

d) Menindaklanjuti gejala anomali keamanan yang terdeteksi yang muncul bersamaan dengan sumber daya sistem informasinya.

e) Melakukan pengujian keamanan sebagaimana yang dibutuhkan.

5. Pengembang Aplikasi Web

Para pengembang aplikasi Web bertanggung jawab atas tampilan, fungsi-fungsi, perform, keamanan dari konten Web dan

aplikasi berbasis Web. Sebagaimana telah disampaikan sebelumnya bahwa ancaman serangan sebagian besar ditujukan pada aplikasi, bukan pada perangkat lunak atau OS Web server. Jika para pengembang aplikasi Web tidak memastikan bahwa kode mereka memperhitungkan keamanan, maka keamanan Web server akan lemah tidak peduli seberapa aman server itu sendiri dan infrastruktur pendukungnya. Para pengembang aplikasi Web harus memastikan aplikasi yang mereka implementasikan memiliki karakteristik-karakteristik berikut:

a) Mendukung suatu otentikasi, otorisasi, dan mekanisme kendali akses yang aman sebagaimana yang dibutuhkan

b) Menyelenggarakan validasi input sedemikian hingga mekanisme keamanan aplikasi tidak dapat dilewati ketika seorang pengguna yang jahat memaksa dengan data yang dikirimkannya ke aplikasi, termasuk permintaan HTTP, *header, query string, cookies, form field, dan field* tersembunyi.

c) Memproses kesalahan dengan cara yang aman sedemikiam hingga tidak mengarah ke arah penyingkapan informasi implementasi yang sensitif.

d) Melindungi informasi sensitif yang diproses dan/atau disimpan oleh aplikasi. Perlindungan yang tidak memadai dapat menyebabkan *data tampering* dan akses kepada informasi konfidensial seperti *username, password*, dan nomor kartu kredit.

e) Memelihara rekaman (*log)* khusus aplikasi tersendiri. Pada banyak contoh kejadian, hasil rekaman (*logging)* Web server tidaklah cukup untuk melacak apa yang dilakukan oleh seorang pengguna pada level aplikasi, untuk memelihara *log* sendiri membutuhkan aplikasi.

f) Sangat sulit melawan serangan DoS pada level aplikasi.

Meskipun serangan DoS sebagian besar ditujukan pada jaringan dan layer transport, aplikasi itu sendiri dapat menjadi suatu sasaran. Jika seorang pengguna yang jahat dapat memonopoli aplikasi atau sumber daya sistem yang dibutuhkan, maka pengguna yang sah dapat terhalang dari penggunaan sistem.

2.3. Tahapan Pengelolaan Keamanan Web Server

Untuk memastikan keamanan dari suatu Web server dan infrastruktur jaringan pendukung, organisasi harus mengimplementasikan hal-hal berikut:

1. **Kebijakan Kemananan Sistem Informasi Organisasi** — Suatu kebijakan keamanan harus mespesifikasikan prinsip dan aturan-aturan dasar keamanan sistem informasi dan tujuan internal yang diharapkan. Kebijakan juga harus menegaskan siapa dalam organisasi yang bertanggung jawab untuk setiap tahap keamanan informasi (misalnya implementasi, pemberlakuan, audit, dan review). Supaya efektif, kebijakan harus diberlakukan secara konsisten diseluruh organisasi. Umumnya, CIO dan manajemen senior bertanggung jawab membuat konsep kebijakan keamanan organisasi.
2. **Konfigurasi/Kontrol Perubahan dan Manajemen** — Proses pengendalian modifikasi terhadap suatu sistem desain, perangkat keras, *firmware*, dan perangkat lunak menyediakan jaminan yang cukup bahwa sistem dilindungi terhadap kemungkinan modifikasi yang tidak sesuai ebelum, selama dan sesudah implementasi sistem. Kontrol konfigurasi mengarah pada konsistensi terhadap kebijakan keamanan sistem informasi. Kontrol konfigurasi pada umumnya diawasi oleh suatu dewan kendali konfigurasi yang merupakan otoritas yang menentukan semua perubahan yang diusulkan terhadap suatu sistem informasi. Jika sumber daya memungkinkan, pertimbangkan penggunaan pengembangan, jaminan kualitas,

dan/atau lingkungan pengujian sedemikian sehingga kemugkinan perubahan yang terjadi dapat diteliti kesalahannya secara kritis dan diuji sebelum *deployment*.

3. **Pengkajian dan Manajemen Resiko** — Pengkajian terhadap resiko merupakan proses analisa dan interprestasi resiko yang berkaitan dengan penentuan jangkauan proses dan metodo*log*i, pengumpulan dan analisa data terkait resiko, dan mereprestasikan hasil analisis resiko. Pengumpulan dan analisis data resiko membutuhkan identifikasi aset, ancaman, kerawanan, penjagaan, konsekuensi, dan peluang suatu serangan dapat berhasil. Manajemen resiko merupakan proses pemilihan dan kontrol implementasi untuk mengurangi resiko hingga level yang dapat diterima organisasi.

4. **Konfigurasi yang Distandarkan** — Organisasi harus mengembangkan konfigurasi pengamanan yang distandarkan untuk penggunaan OS dan aplikasi secara luas. Ini akan memberikan rekomendasi untuk para administrator Web server dan jaringan dalam hal bagaimana mengkonfigurasi sistem secara aman dan memastikan konsistensi dan kepatuhan terhadap kebijakan keamanan organisasi. Karena untuk membobol suatu jaringan cukup hanya dibutuhkan satu *host* yang dikonfigurasikan secara tidak aman, khusus bagi organisasi dengan jumlah *host* yang signifikan dianjurkan untuk mengaplikasikan rekomendasi ini.

5. **Praktek-praktek Pemrograman yang Aman** — Organisasi harus mengadopsi pedoman pengembangan aplikasi terbaik terutama dari segi keamanan untuk memastikan bahwa mereka mengembangkan aplikasi Web mereka dengan suatu cara yang aman.

6. **Kesadaran dan Pelatihan Keamanan** — Suatu program pelatihan keamanan merupakan hal penting untuk keseluruhan personil keamanan informasi di suatu organisasi. Membuat para pengguna dan administrator sadar akan tanggung jawab

keamanan mereka dan mengajarkan praktek yang tepat membantu mereka merubah tingkah-laku mereka untuk menyesuaikan diri dengan praktek terbaik. Pelatihan juga mendukung akuntabilitas individu, yang merupakan metode yang penting untuk meningkatkan keamanan sistem informasi. Mengikutsertakan masyarakat umum dalam pelatihan dapat juga dilakukan untuk meningkatkan kesadaran masyarakat terhadap pentingnya keamanan informasi.

7. ***Contingency*, Kontinuitas Operasi, dan Perencanaan Pemulihan dari Bencana —** Rencana *contingency*, kontinuitas rencana operasi, dan rencana pemulihan dari bencana dibuat terlebih dahulu supaya jika terjadi gangguan, organisasi dapat mengatasinya dengan minimum *down time* dan cepat kembali beroperasi secara normal.
8. **Sertifikasi dan Akreditasi —** Sertifikasi dalam konteks keamanan sistem informasi berarti bahwa suatu sistem telah dianalisis untuk menentukan seberapa baik sistem tersebut memenuhi semua persyaratan keamanan organisasi. Akreditasi muncul ketika manajemen organisasi menyatakan bahwa sistem tersebut memenuhi persyaratan keamanan organisasi.

2.4. Perencanaan Keamanan Sistem

Tujuan dari perencanaan keamanan sistem adalah untuk meningkatkan perlindungan terhadap sumber daya sistem informasi. Perencanaan yang baik dalam melindungi aset informasi membutuhkan para pengelola dan pemilik informasi. Mereka perlu untuk diyakinkan bahwa aset informasi mereka cukup terlindungi dari kehilangan, penyalahgunaan, akses atau modifikasi yang tidak sah, ketidaktersediaan, dan aktivitas yang tak terdeteksi.

Untuk institusi Pemerintah, seluruh sistem informasi harus tercakup oleh suatu rencana keamanan sistem. Organisasi lain harus mempertimbangkan secara matang penyelesaian suatu rencana

keamanan sistem untuk setiap sistem mereka juga. Pemilik sistem informasi, secara umum merupakan pihak yang bertanggung jawab untuk memastikan bahwa rencana keamanan dibangun dan dipelihara, dan bahwa sistem dibangun dan dioperasikan sesuai dengan kebutuhan keamanan yang telah disepakati. Pemilik sistem informasi bertanggung jawab terhadap penentuan parameter operasi sistem, fungsi yang sah, dan persyaratan keamanan. Pemilik informasi yang disimpan, diproses, atau ditransmisikan pada suatu sistem mungkin saja merupakan pihak yang sama dengan pemilik sistem informasi.

2.5. Persyaratan Sumber Daya Manusia

Saat mempertimbangkan implikasi sumber daya manusia terhadap pengembangan dan penyiapan suatu Web server, organisasi harus mempertimbangkan hal-hal berikut:

1. **Personil yang dibutuhkan** — Harus mempertimbangkan posisi yang dibutuhkan, seperti administrator sistem dan Web server, Webmaster, administrator jaringan, dan ISSO.
2. **Kecakapan yang dibutuhkan** — Kecakapan yang dimaksud meliputi apa yang dibutuhkan untuk dapat melakukan perencanaan, pengembangan, dan pemeliharaan Web server dengan cara yang aman. Contohnya, administrasi OS, administrasi jaringan, konten interaktif, dan pemrograman.
3. **Personil yang tersedia** — Termasuk di dalamnya pertimbangan mengenai sumber daya manusia yang tersedia dalam lingkup organisasi serta kecakapan apa yang telah mereka miliki saat ini. Tentukan apakah sudah memadai untuk mengelola Web server. Seringkali, suatu organisasi mendapati bahwa sumber daya manusia yang ada tidak memadai dan perlu mempertimbangkan pilihan-pilihan seperti memberikan pelatihan pada staf yang ada dan/atau memperkerjakan staf tambahan.

2.6. Alternatif *Platform* (Sistem Operasi) Web Server

Meskipun banyak organisasi mengelola *Web server* yang beroperasi menggunakan suatu OS (sistem operasi) yang umum dikenal, terdapat salah satu alternatif OS yang spesifik seperti :

2.6.1 Trusted Operating System

Sistem operasi terpercaya (Trusted operating systems [TOS]) merupakan OS yang dimodifikasi atau ditingkatkan keamanannya dengan menyertakan mekanisme keamanan tambahan yang tidak ditemukan disebagian besar OS *general purpose*. Pada dasarnya TOS tersebut diciptakan untuk memenuhi kebutuhan pemerintah akan sistem kendali akses yang bersifat keharusan (*mandatory access contro/* [MAC]) dengan tingkat keamanan yang tinggi. TOS menyediakan suatu kebijakan kontrol seluas jangkauan sistem yang sangat aman, seperangkat hak khusus akses yang ditentukan dengan sangat hati-hati, dan *logging* yang luas dan kapabilitas audit. TOS umumnya digunakan dalam aplikasi yang mengutamakan keamanan.

Hal-hal berikut merupakan pokok-pokok persoalan yang perlu dipikirkan saat mempertimbangkan suatu Web *platform*:

1. Apa OS yang berjalan dan bagaimana perlakuan khusus terhadapnya dalam pengujian keamanan?
2. Apakah organisasi memiliki staf ahli yang dibutuhkan dalam mengatur TOS?
3. Apakah biaya tambahan untuk membeli dan mendukung suatu TOS diimbangi oleh keuntungannya?
4. Apakah TOS sesuai dengan aplikasi dan *scripts* Web yang sudah ada dalam organisasi?

5. Apakah TOS sesuai dengan aplikasi-aplikasi dan server-server lain dalam organisasi dimana mereka akan beroperasi?

2.6.2. Appliance (Peralatan khusus) Web Server

Appliance Web server adalah suatu kombinasi perangkat lunak dan perangkat keras yang didesain untuk menjadi Web server "*plug-and-play*". *Appliance* tersebut memanfaatkan penggunaan OS yang disederhanakan yang dioptimalkan untuk mendukung suatu Web server. OS yang disederhanakan tersebut meningkatkan keamanan dengan memperkecil sejumlah fitur, layanan, dan pilihan yang tidak diperlukan. Aplikasi Web server pada sistem-sistem ini seringkali diperkuat dan dikonfigurasikan sebelumnya untuk tujuan keamanan. Namun tipe ini tidak cocok untuk situs Web dengan layer yang besar, kompleks, dan beragam.

Tipe-tipe yang paling umum dari *appliance* keamanan diantaranya :

1. Akselerator *SSL*, yang mengambil beban (*off-load)* pemrosesan komputasi yang besar untuk memulai koneksi SSL/TLS.
2. *Gateway* Keamanan, yang memonitor lalulintas HTTP menuju dan dari Web server untuk serangan-serangan yang berpotensi dan mengambil tindakan seperlunya.
3. Filter Konten, yang melakukan monitor lalu lintas menuju dan dari Web server untuk data yang sensitif atau tidak tepat dan mengambil tindakan seperlunya.
4. *Gateway* Otentikasi, yang membuktikan keaslian para pengguna melalui berbagai macam mekanisme otentikasi dan kontrol akses menuju *Universal Resource Locators* (URL) yang menjadikan Web server itu sendiri sebagai *host*-nya.

Sebagian atau seluruh fungsi yang disebutkan di atas dikombinasikan dalam suatu alat tunggal, yang disebut sebagai suatu *reverse proxy*.

Beberapa hal berikut merupakan pokok-pokok persoalan yang harus dipertimbangkan saat pembelian *appliance* Web:

1. Apa OS yang mendasarinya dan bagaimana perlakuan khusus terhadapnya dalam pengujian keamanan?
2. Bagaimana *appliance* itu sendiri diperlakukan dalam pengujian keamanan? Biasanya opsi-opsi konfigurasi dari *appliance* Web dibatasi seperlunya sehingga suatu *appliance Web* umumnya hanya akan menjadi sama amannya dengan konfigurasi instalasi *default*-nya.
3. Seberapa kompleks infrastruktur Web server organisasi? Bisa jadi dengan jenis *appliance* yang berbeda mungkin tidak bekerja sama secara baik.
4. Apakah opsi-opsi ekspansi melekat dalam *appliance* yang dapat diterima pada organisasi?
5. Bagaimana sulitnya untuk mengkonfigurasi *appliance*? Apakah *appliance* cukup fleksibel untuk memenuhi kebutuhan organisasi?
6. Seberapa cepat organisasi menanggapi dan menyediakan *patch* untuk kerawananan yang potensial?
7. Apakah perangkat lunak yang mendasarinya digunakan pada *appliance propiertary, open source,* atau suatu kombinasi dari keduanya?
8. Seberapa lama organisasi pembuat *appliance* akan mendukung *appliance* dan apa sejarah dukungan untuk *appliance* warisan dari organisasi ?

2.6.3 Sistem Operasi *Pre-Hardened* dan Web Servers

Paket *pre-hardened* OS (OS yang fisiknya mendapatkan perkuatan sebelumnya) dan Web server yang dimodifikasi dan dikonfigurasi sebelumnya untuk menyediakan keamanan yang tinggi dan didesain

untuk mempermudah konfigurasi melalui penggunaan *scripts* yang dikompilasi sebelumnya dan *interface* user grafis (*graphical user interfaces* [GUI]), yang terdiri dari :

1. Konfigurasi default awal yang aman
2. Perkuatan OS atau TOS
3. Perkuatan perangkat lunak Web server
4. Kapabilitas audit yang luas
5. *Wrapper* aplikasi
6. *Wrapper* jaringan dan/atau kapabilitas *host*-based firewall
7. *Host*-based IDSs
8. Administrasi keamanan yang disederhanakan (misalnya menus, GUI)

Beberapa pokok-pokok persoalan yang harus dipertimbangkan saat memikirkan pembelian suatu *appliance* Web yang diperkuat (*pre-hardened*):

1. Apa OS yang mendasarinya dan bagaimana perlakuan khusus terhadapnya dalam pengujian keamanan?
2. Bagaimana aplikasi Web server itu sendiri diperlakukan dalam pengujian keamanan?
3. Seberapa sulit pengadministrasiannya?
4. Apakah perkuatan aplikasi Web server dan OS sesuai dengan aplikasi Web dan *script* yang ada dalam organisasi?
5. Seberapa cepat menanggapi dan menyediakan *patches* untuk kerawanan yang potensial?

2.6.4 *Virtual Platforms*

Melalui virtualisasi, suatu komputer *host* tunggal dapat menjalankan

beberapa mesin virtual, masing-masing dengan suatu *guest* OS yang berbeda dan aplikasi yang sesuai. Tipe utama tekno*log*i mesin virtual adalah :

1. *Full Virtualization*, yang mensimulasikan semua perangkat keras yang dibutuhkan oleh *guest* OS. Virtualisasi penuh ini berguna dalam situasi dimana *guest* OS dijalankan pada suatu arsitektur mesin yang berbeda dari *host*. Virtualisasi penuh memberikan hasil dalam suatu performa hasil yang signifikan karena seluruh perintah harus diemulasikan (diusahakan untuk menyamai/melebihi) oleh perangkat lunak.
2. *Native Virtualization*, yang hanya mensimulasikan perangkat keras yang diperlukan untuk menjalankan suatu guest OS yang tidak dimodifikasi. Dalam virtualisasi asal, sebagian besar perintah dapat diteruskan tanpa dimodifikasi ke unit pemroses komputer (*computer processing unit* [CPU]) *host* dengan berakibat mengurangi performa hasil.
3. *Paravirtualisasi*, yang tidak mensimulasikan perangkat keras namun sebaliknya menawarkan suatu aplikasi pemrograman interface (API) yang dapat digunakan oleh suatu guest OS yang dimodifikasi, atau mengambil keuntungan dari kemampuan virtualisasi yang didukung oleh prosesor.

3. Keamanan Sistem Operasi Web Server

Melindungi suatu Web server dari kebobolan melibatkan perkuatan secara fisik dari OS web server, aplikasi Web server, dan jaringan untuk mencegah entitas yang tidak dikenal dapat melakukan penyerangan secara langsung terhadap Web server. Langkah pertama dalam mengamankan suatu Web server yaitu perkuatan secara fisik terhadap OS Web server.

Lima langkah dasar yang dibutuhkan untuk memelihara keamanan OS dasar:

1. Merencanakan instalasi dan penyebaran OS *host* dan komponen lain untuk Web server
2. Melakukan *patch* dan *update* OS *host* seperlunya
3. Memperkuat secara fisik dan mengkonfigurasikan OS *host* untuk mengatasi keamanan secukupnya
4. Menginstal dan mengkonfigurasikan tambahan kontrol-kontrol keamanan, jika dibutuhkan
5. Menguji OS *host* untuk memastikan bahwa keempat langkah sebelumnya cukup mengatasi seluruh pokok persoalan keamanan

3.1. Instalasi dan Konfigurasi Sistem Operasi

Subbab ini memberikan ulasan sepintas terhadap langkah kedua, ketiga, dan keempat dari daftar diatas. Hasil kombinasi dari langkah-langkah tersebut semestinya merupakan suatu tingkat perlindungan yang masuk akal bagi OS Web server.

3.1.1 *Patch* dan *Upgrade* Sistem Operasi Web Server

Sekali suatu OS terinstal, perlu menerapkan *patch* atau *upgrade* yang dibutuhkan untuk memperbaiki kerawanan yang ada. Kerawanan apapun yang dikenal yang dimiliki oleh suatu OS harus diperbaiki sebelum digunakan sebagai *host* dari suatu Web server.

Administrator harus memastikan bahwa Web server, khususnya yang baru, cukup dilindungi selama proses *patching.* Sebagai contoh, suatu Web server yang tidak di-*patch* secara penuh atau tidak dikonfigurasi secara aman mungkin dapat bobol oleh ancaman, jika dapat diakses secara publik ketika sedang dilakukan *patch*. Saat mempersiapkan Web server baru, maka administrator harus melakukan hal-hal berikut:

1. Menjaga server tidak terhubung dengan jaringan atau menghubungkan mereka hanya kepada suatu jaringan terisolasi hingga seluruh *patch* telah dipindahkan ke server melalui alat *out-of-band* (misalnya CD) dan diinstal, serta telah dilakukan langkah konfigurasi lain pada Bagian 3.1.
2. Menempatkan server dalam suatu *virtual local area network* (VLAN) atau segmen jaringan lain yang sangat membatasi tindakan apa yang dapat dijalankan oleh *host* pada server tersebut dan komunikasi apa yang dapat mencapai *host*, yaitu hanya memperbolehkan melakukan *patch* dan mengkonfigurasi *host*. VLAN dapat dengan mudah disalah-konfigurasikan dengan cara-cara yang dapat mengurangi atau menghilangkan fungsinya sebagai kendali keamanan. Organisasi yang berencana menggunakan VLAN harus memastikan bahwa VLAN dikonfigurasi dengan tepat dan bahwa perubahan konfigurasi apapun diverifikasi dengan hati-hati.

Para administrator secara umum sebaiknya tidak mengaplikasikan *patch* ke Web server tanpa mengujinya terlebih dahulu dalam sistem lain yang identik dan sudah dikonfigurasi. Hal ini dilakukan karena *patch* dapat menyebabkan timbulnya masalah yang tidak

diharapkan. Meskipun para administrator dapat mengkonfigurasi Web server untuk men-*download patch* secara otomatis, tetapi server sebaiknya tidak dikonfigurasi untuk itu, supaya *patch* dapat dicek terlebih dahulu.

3.1.2 Me-nonaktif-kan Layanan dan Aplikasi Yang Tidak Perlu

Idealnya, suatu Web server semestinya berada pada suatu *single purpose host* (*host* satu fungsi) yang sudah ditetapkan. Jika memungkinkan, instal konfigurasi OS *default* dan kemudian tambah atau hilangkan layanan-layanan dan aplikasi-aplikasi sesuai kebutuhan. Pilih opsi "instalasi minimum", jika tersedia, untuk meminimalkan upaya yang dibutuhkan dalam menghilangkan layanan-layanan yang tidak perlu. Beberapa tipe umum dari layanan dan aplikasi yang biasanya di-non-aktifkan jika tidak dibutuhkan, antara lain:

1. Layanan sharing *file* dan printer sharing, misalnya Windows Network berbasis Input/Output System [NetBIOS] sharing *file* dan printer, Network *File* System [NFS], *File* Transfer Protocol [FTP]
2. Layanan jaringan nirkabel
3. Program *remote control* dan *remote access*, khususnya bagi yang tidak mengenkripsi jalur komunikasinya (misalnya, Telnet). Jika suatu *remote control* atau *remote acces* benar-benar dibutuhkan dan komunikasinya tidak dienkripsi dengan kuat, harus diberi *tunnel* (saluran) melalui protokol yang menyediakan enkripsi, seperti *secure shell* (SSH) atau IP Security (Ipsec).
4. Layanan direktori, misalnya Lightweight Directory Access Protocol [LDAP], Kerberos, Network Information System [NIS]
5. Layanan email (misalnya Simple Mail Transfer Protocol [SMTP])
6. *Language compilers dan libraries*

7. *Tools* pengembang sistem
8. *Tool* dan *utilities* manajemen sistem dan jaringan, termasuk Simple Network Management Protocol (SNMP).

3.1.3 Konfigurasi Otentikasi Pengguna Sistem Operasi

Bagi *Web server*, para pengguna sah yang dapat mengkonfigurasi OS dibatasi pada sejumlah kecil administrator Web server dan Webmaster yang ditunjuk. Para pengguna yang dapat mengakses Web server, di antaranya adalah sebagian kelompok komunitas Internet yang tak-terbatas hingga yang terbatas. Jika dibutuhkan, kebijakan pembatasan pengguna dapat diberlakukan. Administrator Web server harus mengkonfigurasi OS untuk membuktikan otentikasi sah atau tidaknya seorang calon pengguna dengan mensyaratkan bukti tertentu.

Untuk memastikan otentikasi yang tepat terhadap pengguna, dapat dilakukan langkah-langkah berikut:

1. **Hapus atau NonAktif-kan Akun dan Grup Default yang Tidak Dibutuhkan atau Non-Interaktif —** Konfigurasi default OS seringkali termasuk akun *guest* (dengan atau tanpa password), akun level administrator atau *root*, dan akun yang berkaitan dengan layanan-layanan lokal dan jaringan. Nama dan password untuk akun-akun tersebut populer. Hilangkan atau non-aktifkan akun yang tidak perlu untuk menghilangkan penggunaannya oleh penyerang, termasuk *guest akun* pada komputer yang mengandung informasi sensitif. Jika tidak ada kebutuhan untuk mempertahankan suatu guest atau grup akun, batasi akses dengan ketat dan rubah *password* sesuaikan dengan kebijakan *password* organisasi. Untuk sistem Unix, non-aktifkan *log*in shell atau sediakan suatu *log*in shell dengan fungsi NULL (misalnya /bin/false).
2. **Ciptakan Grup-grup Pengguna —** Bagikan para pengguna ke dalam grup-grup yang tepat. Kemudian tetapkan hak untuk

grup, sebagaimana yang didokumentasikan dalam rencana penyebaran. Pendekatan ini lebih diutamakan untuk menetapkan hak kepada para pengguna individu, yang mana menjadi semakin susah digunakan dengan jumlah pengguna yang besar.

3. **Ciptakan Akun Pengguna** — Rencana penyebaran mengidentifikasikan siapa yang akan diberi wewenang untuk menggunakan setiap komputer dan layanan-layanannya. Ciptakan hanya akun-akun yang perlu. Perbolehkan penggunaan akun bersama hanya saat tidak ada alternatif yang dapat dijalankan.

4. **Cek Kebijakan *Password* Organisasi** — Atur *password akun* secara tepat. Kebijakan ini harus mencakup hal-hal berikut:

   a) **Ukuran** — Panjang minimum suatu *password*. Tetapkan panjang minimum paling sedikit delapan karakter.

   b) **Kompleksitas** — Disyaratkan campuran karakter. *Password* disyaratkan yang mengandung huruf besar dan huruf kecil dan paling sedikit suatu karakter non-alphabet, dan tidak merupakan kata dari kamus.

   c) **Durasi Waktu** — Seberapa lama suatu *password* boleh tetap tidak berubah. Para pengguna disyaratkan untuk mengubah *password* mereka secara periodik. Password level administrator atau root sebaiknya diubah tiap 30 hingga 120 hari. Periode untuk *password* level pengguna sebaiknya ditentukan oleh kombinasi panjang dan kompleksitas *password* dengan tingkat sensitivitas informasi yang dilindungi.

   d) **Penggunaan Kembali** — Apakah suatu *password* mungkin digunakan kembali. Dimana pengguna tidak diijinkan hanya menambah karakter pada password yang telah digunakan, misalnya, password yang asli adalah ”rahasiaku” dan diubah menjadi ”1rahasiaku” atau ”rahasiaku1”.

e) **Otoritas** — Siapa yang diijinkan untuk mengubah atau mengatur kembali *password* dan pembuktian seperti apa yang diperlukan sebelum memulai suatu perubahan.

f) **Keamanan *Password*** — Bagaimana *password* harus diamankan, seperti halnya tidak menyimpan *password* secara tidak terenkripsi pada mail server, dan mensyaratkan para administrator untuk menggunakan *password* yang berbeda untuk akun administrasi email mereka dari akun adminstrasi mereka lainnya.

5. **Konfigurasikan Komputer untuk Mencegah *Password Guessing*** — Merupakan hal yang relatif mudah bagi seorang pengguna yang tidak sah untuk mencoba mendapatkan akses ke suatu komputer dengan menggunakan perangkat lunak *tool* otomatis yang mencoba semua password. Jika OS menyediakan kemampuan itu, konfigurasikan OS untuk meningkatkan periode waktu antara percobaan *login* dari setiap percobaan yang tidak berhasil. Jika hal itu tidak memungkinkan, alternatifnya adalah menolak *log*in setelah sejumlah batas tertentu percobaan gagal (misal, tiga). Biasanya, akun ”dikunci” untuk suatu periode waktu tertentu (contohnya 30 menit) atau hingga seorang pengguna dengan otoritas yang tepat mengaktifkannya kembali.

6. **Instal dan Konfigurasikan Mekanisme Keamanan Lain untuk Menguatkan Otentikasi**- Jika informasi pada Web server membutuhkannya, pertimbangkan untuk menggunakan mekanisme otentikasi lain seperti *biometric, smart card,* sertifikat *client/server*, atau sistem one-time *password*. Mekanisme tersebut dapat saja lebih mahal dan susah untuk diimplementasikan, namun mungkin dapat dibenarkan dalam beberapa keadaan. Saat mekanisme otentikasi dan peralatan seperti itu digunakan, kebijakan organisasi harus diubah menyesuaikan, jika dibutuhkan. Beberapa kebijakan organisasi mungkin telah mensyaratkan pengggunaan mekanisme otentikasi yang kuat.

Sebagaimana telah disebutkan sebelumnya, para penyerang yang menggunakan *network sniffer* dapat dengan mudah menangkap *password* yang dilewatkan melalui suatu jaringan dalam bentuk teks terang. Organisasi harus mengimplementasikan tekno*log*i otentikasi dan enkripsi, seperti SSL/TLS, *Secure Shell* (SSH), atau *virtual private networking* (VPN), untuk melindungi *password* selama transmisi dan mengurangi kemungkinan berhasilnya serangan *man-in-the-middle* dan *spoofing.*

3.1.4 Konfigurasi Kontrol Sumber Daya Secara Tepat

Seluruh OS server yang modern yang umum digunakan memberikan kemampuan untuk menetapkan secara khusus *privilege user* untuk *file-file*, direktori-direktori, peralatan, dan sumber daya yang bersifat komputasi lainnya. Dengan pengaturan access control dan penolakan akses pribadi yang tidak sah secara hati-hati, administrator Web server dapat mengurangi pelanggaran keamanan yang disengaja maupun tidak disengaja.

3.1.5 Instalasi dan Konfigurasi Kontrol Keamanan Tambahan

OS seringkali tidak menyertakan seluruh kontrol keamanan yang dibutuhkan untuk mengamankan OS, layanan-layanan, dan aplikasi-aplikasi. Dalam kasus seperti itu, para administrator perlu memilih, menginstal, dan mengkonfigurasikan perangkat lunak tambahan untuk menyediakan kontrol yang tidak ada. Kontrol-kontrol yang dibutuhkan secara umum termasuk hal-hal berikut:

1. Perangkat lunak Anti-malware, seperti perangkat lunak anti-virus, perangkat lunak anti-*spyware*, dan *rootkit detector*, untuk melindungi OS lokal terhadap malware dan untuk mendeteksi dan memberantas munculnya suatu infeksi.
2. Perangkat lunak pendeteksi dan pencegah penyerangan berbasis *host*, untuk mendeteksi serangan-serangan yang dijalankan terhadap Web server, termasuk serangan DoS.

3. Firewall berbasis *host*, untuk melindungi server dari akses yang tidak sah.
4. Perangkat lunak manajemen *patch* untuk memastikan bahwa kerawanan diatasi secara tepat. Perangkat lunak manajemen *patch* hanya dapat digunakan untuk mengaplikasikan *patch* atau juga untuk mengidentifikasi kerawanan baru dalam OS, layanan-layanan, dan aplikasi-aplikasi Web server.

Saat merencanakan kontrol keamanan, para administrator Web server sebaiknya mempertimbangkan sumber daya yang akan diserap oleh kontrol keamanan. Performa suatu server dapat menurun jika tidak memiliki memori yang cukup dan kapasitas pemrosesan untuk kontrol.

3.2. Pengujian Keamanan Sistem Operasi

Metode-metode yang efektif untuk pengujian OS termasuk scanning kerawanan dan Penetration Testing. Scanning kerawanan biasanya memerlukan penggunaan suatu scanner kerawanan yang otomatis untuk melakukan scan terhadap suatu*host* atau *grup host* pada suatu jaringan untuk menemukan kerawanan aplikasi, jaringan, dan OS. Penetration Testing merupakan suatu proses pengujian yang didesain untuk membobol suatu jaringan menggunakan *tool* dan metodo*log*i-metodo*log*i dari seorang penyerang. Scanning kerawanan harus dilakukan secara berkala, paling tidak mingguan hingga bulanan, dan Penetration Testing harus dilakukan paling tidak tahunan. Karena teknik-teknik pengujian tersebut juga dapat diaplikasikan untuk menguji aplikasi Web server.

Secara umum pengujian seharusnya tidak dijalankan pada Web server produksi itu sendiri. pengujian untuk *patch* dan perubahan pada sistem harus dijalankan pada suatusistem yang terpisah atau web server simulasi.

4. Keamanan Web Server

Setelah OS diinstal dan diamankan, instalasi perangkat lunak Web server dapat dimulai. Sebelum memulai proses ini, baca dengan seksama dokumentasi dari manufaktur Web server dan pahami berbagai opsi yang ada selama proses instalasi. Instalasi harus dimulai hanya setelah langkah awal selesai.

Suatu server yang terkonfigurasi dan/atau di *patch* secara parsial sebaiknya tidak terbuka ke jaringan eksternal (misalnya Internet) atau para pengguna eksternal. Akses ke jaringan internal harus sangat dibatasi sampai semua dilakukan instalasi, *patch* dan konfigurasi dengan aman. Web server yang tidak aman dapat bobol dalam hitungan menit setelah ditempatkan di Internet.

4.1. Instalasi Web Server Secara Aman

Instalasi dan konfigurasi aplikasi Web server yang aman mencerminkan proses OS yang dibahas pada Bab 3. Seperti sebelumnya, prinsip yang sangat penting adalah hanya menginstal layanan-layanan yang dibutuhkan Web server dan mengurangi kerentanan apapun yang diketahui melalui *patch* atau *upgrade.* Aplikasi, layanan ataupun *script* yang tidak perlu harus dihilangkan segera setelah proses instalasi selesai. Selama instalasi Web server, langkah-langkah yang harus dilakukan adalah sebagai berikut :

1. Instal perangkat lunak Web server pada *dedicated host* atau pada *dedicated guest* OS jika menggunakan virtualisasi.
2. Instal *patch* atau *upgrade* perangkat lunak Web Server untuk mengatasi kerawanan yang telah diketahui.
3. Sediakan media *storage/disk* secara fisik atau partisi secara *log*ik untuk konten Web, yang terpisah dari OS dan aplikasi Web server.

4. Hapus atau nonaktifkan layanan-layanan yang di-*install* oleh aplikasi Web server namun tidak dibutuhkan, misal gropher, FTP, administrasi remote.
5. Hapus atau nonaktifkan semua akun *login default* yang tidak dibutuhkan, yang tercipta pada saat instalasi Web server.
6. Hapus semua dokumentasi manufaktur dari server.
7. Hapus semua *file* contoh atau tes dari server, termasuk script dan *executable code*.
8. Terapkan *template* keamanan yang sesuai atau *script* untuk memperkuat keamanan ke server.
9. Konfigurasikan kembali HTTP service banner (dan yang dibutuhkan lainnya untuk tidak mempublikasikan atau membuat pemberitahuan mengenai tipe dan versi Web server serta OS.

Dalam menginstalasi Web server sebaiknya dengan nama direktori, lokasi direktori dan nama *file* yang tidak standar. Banyak *tools* serangan dan *worm* yang menjadikan Web server sebagai targetnya, hanya dengan mencari *file* dan direktori dalam lokasi *default*.

4.2. Konfigurasi Kontrol Akses

Kebanyakan Web server yang menjadi *host* OS mempunyai kemampuan untuk menetapkan *access previleges* bagi masing-masing individu untuk mengakses *file*, perangkat dan sumber daya yang bersifat komputasi lainnya Merupakan hal yang penting untuk membuat *permission* yang identik untuk OS dan aplikasi Web server. Para administrator Web server harus mempertimbangkan hal yang terbaik untuk mengkonfigurasi kendali akses guna melindungi informasi yang tersimpan pada Web server dari dua perspektif:

1. Membatasi akses dari aplikasi Web server pada sebagian sumber daya yang bersifat komputasi.

2. Membatasi akses para pengguna melalui kontrol akses yang diberlakukan oleh Web server, dimana perlu diterapkan tingkat kontrol akses yang lebih detil.

Berikut adalah *file-file* yang secara umum dapat diterapkan kontrol aksesnya, yaitu :

1. *File* perangkat lunak aplikasi dan konfigurasi
2. *File* yang berkaitan langsung dengan mekanisme pengamanan:
    a) *File* yang berisinilai hash dari *password* dan *file* lainnya yang digunakan dalam otentikasi
    b) *File* yang berisi informasi otorisasi yang digunakan untuk mengendalikan akses
    c) Material kunci kriptografis yang digunakan dalam layanan konfidensialitas, integritas dan non-repudiasi
3. *File log* server dan audit sistem
4. *File* perangkat lunak sistem dan konfigurasi
5. *File* konten Web

4.2.1. Konfigurasi *Permission* pada Aplikasi Web Server

OS Web server dapat digunakan untuk membatasi *file-file* yang dapat diakses oleh proses-proses layanan Web server. Proses ini seharusnya memiliki akses *read-only* (hanya baca) terhadap *file-file* yang diperlukan untuk melakukan layanan dan sebaiknya tidak memiliki akses terhadap *file-file* lainnya, seperti *file log* server.

Hal yang penting juga adalah untuk memastikan bahwa aplikasi Web server tidak dapat menyimpan (atau membaca) *file-file* diluar struktur *file* untuk konten Web. Pastikan bahwa direktori dan *file-file* diluar struktur *file* untuk konten web tidak dapat diakses, sekalipun jika para pengguna melakukan browsing langsung dengan cara mengakses URL dari *file-file* tersebut atau melalui *directory traversal*

attacks (serangan lintas direktori) terhadap proses Web server.

Untuk mengurangi efek tipe-tipe tertentu dari serangan DoS, konfigurasikan Web server untuk membatasi jumlah sumber daya OS yang dikonsumsi, sebagai berikut :

1. Menempatkan konten Web pada *hard drive* atau partisi *log*ik yang berbeda dengan yang digunakan untuk OS dan aplikasi Web server.
2. Menempatkan suatu batasan pada sejumlah ruang *hard drive* yang khusus diperuntukkan bagi *upload*, jika *upload* ke Web server diijinkan. Idealnya, *upload* sebaiknya ditempatkan pada suatu partisi terpisah untuk menyediakan jaminan yang lebih kuat bahwa batasan hard drive tidak dapat dilampaui.
3. Jika *upload* ke Web server diijinkan, memastikan bahwa *file-file* tersebut tidak dapat dibaca oleh Web server hingga setelah beberapa proses pemeriksaan kembali secara otomatis atau manual digunakan untuk menyaringnya. Tindakan ini mencegah Web server digunakan untuk mempropagasi *malware* atau lalulintas perangkat lunak bajakan, *tools* serangan, pornografi, dsb. Dimungkinkan juga untuk membatasi ukuran dari setiap *file* yang di-upload, yang dapat membatasi efek potensial dari suatu serangan DoS yang melibatkan *upload* beberapa *file* yang besar.
4. Memastikan bahwa *file-file log* disimpan dalam suatu lokasi yang diukur secara tepat. Idealnya, *file-file log* sebaiknya disimpan dalam suatu partisi yang terpisah. Jika suatu serangan dapat mengakibatkan ukuran *file log* meningkat diluar batasan yang dapat diterima, suatu partisi fisik dapat membantu menjamin Web server memiliki sumber daya yang cukup untuk menangani situasi dengan tepat.
5. Mengkonfigurasikan jumlah maksimum dari proses Web server dan/atau koneksi jaringan yang seharusnya diijinkan oleh Web server.

4.2.2. Konfigurasi Direktori Konten Web yang Aman

Jangan menggunakan link, alias, atau jalan pintas dalam pohon direktori *file* konten Web yang menunjuk ke direktori atau *file-file* ditempat lain dalam *host server* atau sistem *file* jaringan. Jika memungkinkan, nonaktifkan kemampuan perangkat lunak Web server dalam hal mengikuti link dan alias. Sebagaimana telah dinyatakan sebelumnya, *file log* Web server dan *file* konfigurasi harus terletak diluar pohon direktori *file* yang dikuhususkan untuk konten Web.

Langkah-langkah berikut dibutuhkan untuk membatasi akses ke suatu pohon direktori *file* konten Web yang khusus:

1. Tentukan suatu*hard drive* atau partisi *log*ik tunggal yang diperuntukkan bagi konten Web dan buatlah subdirektori terkait khusus untuk *file* konten Web server, termasuk grafik namun tidak memuat *script* dan program-program lain.
2. Tetapkan suatu pohon direktori tunggal khusus untuk seluruh script atau program eksternal yang dieksekusi sebagai bagian dari konten Web (misalnya, CGI, Active Server Page [ASP], PHP).
3. Nonaktifkan eksekusi script yang tidak secara khusus berada dibawah kontrol dari akun administratif. Tindakan ini dihasilkan dengan pembuatan dan pengontrolan akses terhadap suatu direktori terpisah yang dimaksudkan untuk berisikan script yang sah.
4. Nonaktifkan penggunaan *hard links* atau *symbolic links.*
5. Definisikan suatu matriks akses konten Web yang lengkap. Identifikasikan folder dan *file* mana dalam dokumen Web server yang harus dibatasi dan yang mana yang dapat di akses (dan oleh siapa).

4.2.3. *Uniform Resource Identifiers* dan *Cookies*

Uniform Resource Identifiers (URI) merupakan tekno*log*i pengalamatan dimana URL diciptakan. Secara teknis URL merupakan suatu subbagian dari URI. Ada sejumlah pokok persoalan keamanan yang muncul dari URI. Karena URI dikirimkan secara terang, data apapun yang tersimpan didalamnya dapat dibobol dengan mudah. Sebagai contoh, URI tercatat dalam sejumlah lokasi, termasuk *log* Web browser (yaitu, browser history), *log server proxy*, dan *log* pengacu HTTP pihak ketiga. Dengan demikian, tidak direkomendasikan untuk menyembunyikan data sensitive seperti halnya *usernames* dan *password* atau sumber daya tersembunyi dari server lainnya didalam URI.

Suatu *cookie* merupakan bagian kecil dari informasi yang mungkin ditulis pada *hard drive* pengguna saat pengguna mengunjungi suatu situs Web. *Cookie* dimaksudkan untuk memungkinkan server mengenali suatu browser (pengguna) tertentu. Pada intinya, *cookie* menambahkan *state (*pernyataan) ke protokol HTTP yang tak berpernyataan (*stateless*). Karena *cookie* biasanya dikirim dalam bentuk terang dan disimpan secara terang dalam *host* pengguna, *cookie* rentan untuk dibobol. Sebagai contoh, ada beberapa kerawanan yang diketahui dalam versi terbaru dari Internet Explorer, yang mengijinkan suatu situs Web tak dikenal (*malicious)* untuk mengumpulkan seluruh *cookie* penggunjung secara *remote* tanpa sepengetahuan pengunjung. Oleh karena itu, *cookie* semestinya tidak pernah mengandung data yang dapat digunakan secara langsung oleh seorang penyerang (misal, nama pengguna, password).

Situs Web Pemerintah direkomendasikan untuk tidak menggunakan *cookie,* kecuali ada suatu keadaan yang memaksa untuk mengumpulkan data pada situs, dan hanya dengan adanya persetujuan, pemberitahuan, dan keamanan yang baik.

4.2.4. Mengendalikan Dampak Web "Bots" pada Web Server

Web *bots* (juga dikenal sebagai *crawler* atau *spider*) merupakan aplikasi software yang digunakan untuk mengumpulkan, menganalisis, dan meng-indeks konten Web. Web bots digunakan oleh berbagai organisasi untuk berbagai tujuan. Beberapa contoh termasuk

1. MSNBot, Slurp, dan Googlebot dapat menganalisis, meng-indeks, dan mencatat situs Web untuk mesin pencari (*search engines*) Web seperti halnya Windows Live Search, Yahoo! dan Google.
2. *Mediabot* digunakan oleh Google untuk menganalisa konten yang disajikan oleh suatu halaman AdSense sehingga iklan-iklan yang relevan secara konteks akan disediakan.
3. Hyperlink "validator" digunakan oleh Webmaster untuk memvalidasi *hyperlink* secara otomatis pada situs Web mereka.
4. EmailSiphon dan Cherry Picker merupakan *bots* yang didesain secara khusus untuk bergerak dengan pelan pada situs Web guna mendapatkan alamat *electronic mail* (e-mail) untuk ditambahkan pada *spam mailing list*.
5. Beberapa *spambots* bergerak dengan pelan dalam situs Web untuk mencari formulir *log*in yang digunakan untuk menciptakan alamat email gratis yang merupakan tempat asal pengiriman spam atau untuk *spam blog, guestbook, wikis, dan forum-forum* untuk mendorong urutan *search engine* dari suatu situs Web tertentu.
6. *Screen scrapers* mendapatkan kembali konten dari situs Web untuk meletakkan suatu copy pada server lain. Copy-copy tersebut dapat digunakan untuk *pishing* atau untuk berusaha menghasilkan *ad revenue* (bayaran dari iklan) dengan membuat para pengguna mengunjungi copy tersebut.

7. Beberapa *bot* tak dikenal bergerak dengan pelan dalam situs Web mencari aplikasi yang rawan yang mengandung data sensitif (misal, Social Security Numbers [SSN], data kartu kredit).

Dampak web bots ini dapat diatasi dengan cara :

1. Membuat suatu *file* teks terang yang bernama ”robots.txt”. *File* harus selalu memiliki nama ini, dan harus berada dalam direktori dokumen root Web server. Hanya satu *file* yang diperbolehkan per situs Web. Karena *file* robots.txt merupakan salah suatu standar yang digunakan oleh para pemrogram bot, sehingga bot tak dikenal (seperti EmailSiphon dan Cherry Picker) seringkali mengabaikan *file* ini.*File* robots.txt merupakan suatu *file* teks sederhana yang mengandung beberapa kata kunci dan spesifikasi *file*. Setiap baris *file* dapat saja kosong atau berisi kata kunci tunggal dan informasi yang terkait. Kata kunci digunakan untuk memberitahu robot bagian mana dari situs Web yang tidak disertakan. Kata kunci yang diperbolekan antara lain *User-agent* dan *disallow,* misalkan untuk *disallow* suatu bot tertentu (dalam hal ini Googlebot) dari pemeriksaan suatu halaman Web tertentu:

   User-agent:GoogleBot

   Disallow:tempindex.htm

2. Perlindungan password merupakan jalan satu-satunya yang dapat diandalkan untuk meniadakan *noncompliant bots* atau para pengguna yang ingin tahu. Seringkali, *spambots* mengabaikan robots.txt dan mencari alamat email pada situs Web dan/atau format dimana mereka dapat menambahkan konten yang berkaitan dengan spam.

Ada beberapa teknik untuk mengurangi jumlah *spam*, yaitu :

1. Memblokir formulir pengajuan yang menggunakan kata kunci yang berkaitan dengan *spam.*

2. Menggunakan kata kunci **rel=”nofollow”** dalam semua link yang diajukan, yang akan mengakibatkan *search engine* untuk menghilangkan link pada algoritma *pageranking*- nya, yang secara langsung mempengaruhi tujuan dari suatu *spambot.*

3. Mensyaratkan pengusul untuk memecahkan suatu*automated public Turing test* to tell computers and humans apart (CAPTCHA) sebelum diijinkan mengumpulkan konten.

5.Keamanan Konten Web server

Dua bagian utama dari keamanan Web adalah keamanan aplikasi dan OS server serta keamanan *konten* yang aktual. Dari ketiga hal tersebut, keamanan *konten* seringkali diabaikan. Pemeliharaan keamanan *konten* yang efektif dapat dilakukan dengan tidak menempatkan informasi yang *proprietary*, terklasifikasi, dan informasi sensitif pada suatu Web server yang dapat diakses, kecuali jika dilakukan perlindungan informasi dengan otentikasi pengguna dan enkripsi. Seiring dengan makin meningkatnya perlindungan fisik perimeter jaringan, OS, dan Web server, maka para penyerang pun banyak beralih kepada ekploitasi kerawanan dalam aplikasi Web dan cara informasi diproses pada Web server. Serangan terhadap layer aplikasi tersebut mengeksploitasi elemen-elemen situs Web yang interaktif.

5.1. Batasan Informasi yang dipublikasikan pada Situs Web

Kebijakan terhadap batasan informasi yang dapat dipublikasikan seringkali tidak dibuat. Perlu disadari bahwa karena situs Web seringkali merupakan tempat pertama dimana entitas *malicious* mencari informasi yang bernilai. Para penyerang juga dapat mengambil keuntungan dari konten yang ada dalam suatu situs Web untuk digunakan dalam suatu serangan rekayasa sosial (*social engineering)* atau menggunakan informasi identifikasi individu dalam pencurian identitas. Suatu situs Web seharusnya tidak mengandung informasi berikut:

1. Informasi atau Rekaman (*record*) yang sensitif dan berklasifikasi
2. Aturan dan prosedur personil internal
3. Informasi pribadi tentang para personil dan klien suatu

organisasi

4. Nomor telepon, alamat e-mail, atau daftar umum dari staf kecuali jika diperlukan untuk memenuhi persyaratan organisasi. Ketika suatu alamat e-mail harus dipublikasikan pada suatu situs Web, pertimbangkan penggunaan alamat e-mail umum atau alias (misalnya webmaster@namadomain.go.id sebagai pengganti Amir@namadomain.go.id).
5. Jadwal para pemimpin organisasi atau lokasi aktivitasnya.
6. Informasi tentang komposisi atau persiapan materi yang sensitif
7. Informasi sensitif yang berkaitan dengan keamanan nasional
8. Catatan investigasi
9. Catatan-catatan keuangan, diluar yang sudah tersedia untuk publik.
10. Catatan-catatan medis
11. Prosedur keamanan fisik dan informasi dari organisasi
12. Informasi tentang jaringan dan infrastruktur sistem informasi dari organisasi.
13. Informasi yang berimplikasi pada kerawanan keamanan
14. Rencana-rencana, peta-peta, diagram-diagram, foto udara, dan rencana arsitektural pembangunan, properti, atau instalasi organisasi yang bersifat sensitif dan strategis.
15. Informasi tentang rencana pemulihan bencana, atau rencana kelanjutan operasi kecuali yang mutlak diperlukan
16. Rincian prosedur tanggap darurat, rute evakuasi, atau personil penanggungjawab organisasi untuk persoalan-persoalan tersebut
17. Materi hak cipta tanpa ijin tertulis dari pemilik
18. Privasi atau kebijakan keamanan yang mengindikasikan tipe-

tipe tindakan keamanan yangada hingga ke tingkat yang mungkin dimanfaatkan oleh seorang penyerang.

Untuk memastikan suatu pendekatan yang konsisten, suatu organisasi harus menciptakan suatu kebijakan formal dan proses untuk menentukan dan menyetujui informasi untuk dipublikasikan pada suatu Web server. Hal ini merupakan tanggung jawab CIO dan/atau pejabat urusan publik. Proses seperti itu mengikutsertakan langkah-langkah berikut:

1. Mengidentifikasi informasi yang semestinya dipublikasikan pada Web
2. Mengidentifikasi *audience* sasaran
3. Mengidentifikasi dampak negatif yang memungkinkan dari publikasi informasi
4. Mengidentifikasi siapa yang harus bertanggung jawab untuk pembuatan, publikasi, dan pemeliharaan informasi sensitif ini
5. Membuat atau menyusun informasi untuk publikasi Web
6. Meninjau kembali informasi dalam hal sensitifitas dan kontrol distribusi/pengeluaran (termasuk sensifitas informasi secara keseluruhan)
7. Menentukan akses yang tepat dan kontrol keamanan
8. Mempublikasi informasi
9. Memverifikasi informasi yang dipublikasikan
10. Secara berkala meninjau kembali informasi yang dipublikasikan untuk mengkonfirmasi kepatuhan berkelanjutan terhadap ketentuan yang berlaku.

5.2. Reduksi Serangan-serangan Tidak Langsung terhadap Konten

Serangan-serangan konten yang tidak langsung bukan merupakan

serangan langsung pada suatu Web server atau kontennya. Motif serangan yang paling umum dari serangan-serangan tersebut adalah memaksa para pengguna untuk mengunjungi suatu situs Web yang berniat jahat yang dibuat oleh penyerang dan membuka rahasia informasi pribadi karena para pengguna percaya bahwa situs yang mereka kunjungi merupakan situs Web yang sah. Disamping membutuhkan informasi pribadi yang berkaitan dengan situs Web yang dijadikan sasaran, serangan-serangan tersebut juga dapat diarahkan melawan komputer pengguna dari situs Web yang berniat jahat yang dikunjungi. Tipe-tipe serangan yang tidak langsung yang dijabarkan pada bagian ini adalah *phishing* dan *pharming*.

5.2.1 *Phishing*

Para penyerang p*hishing* menggunakan teknik-teknik rekayasa sosial untuk menipu para pengguna agar mengakses suatu situs Web palsu dan memberikan rahasia informasi pribadi. Dalam beberapa serangan *phishing*, para penyerang mengirimkan suatu e-mail yang terlihat sah yang meminta para pengguna untuk meng-*update* informasi mereka pada situs Web organisasi, namun URL pada e-mail sebenarnya menunjuk pada suatu situs Web yang salah. Serangan-serangan *phishing* lainnya mungkin lebih canggih dan memanfaatkan kerawanan dalam aplikasi situs Web yang sah.

Meskipun *phishing* tidak dapat dicegah secara keseluruhan melalui cara-cara teknis yang digunakan dalam suatu Web server, beberapa teknik dapat mengurangi kemungkinan para pengguna suatu situs Web akan terpikat kedalam suatu serangan *phishing*.

1. Memastikan kewaspadaan pelanggan akan bahaya serangan *phishing* dan bagaimana menghindarinya dengan cara antara lain :

    a) Jangan membalas pesan email atau iklan *popup* yang meminta informasi pribadi atau keuangan.

    b) Jangan mempercayai nomor telepon dalam email atau

iklan *popup*. Tekno*log*i VoIP dapat digunakan untuk mendaftar suatu telepon dengan kode wilayah apapun, yang dapat digunakan untuk penipuan.

c) Gunakan antivirus, anti-spyware, dan perangkat lunak firewall. Perangkat lunak-perangkat lunak tersebut dapat mendeteksi malware dalam suatu mesin pengguna yang terlibat dalam serangan *phishing*.

d) Jangan mengirimkan informasi pribadi atau keuangan lewat email.

e) Tinjau kembali pernyataan kartu kredit dan nomer rekening bank secara teratur.

f) Berhati-hatilah saat mengakses situs Web yang tidak dipercaya karena beberapa kerawanan Web browser dapat dieksploitasi hanya dengan mengunjungi situs semacam itu. Para pengguna harus berhati-hati dalam membuka suatu *attachment* atau men-*download file* apapun dari email atau situs Web yang tidak dipercaya.

2. Memvalidasi komunikasi resmi dengan mempersonalkan email (membuat email menurut selera) dan menyediakan informasi identifikasi yang unik yang seharusnya hanya diketahui oleh organisasi dan pengguna.

3. Menggunakan *signature*pada e-mail. Meskipun *signature* tidak dapat divalidasi secara otomatis oleh aplikasi e-mail pengguna.

4. Menjalankan validasi konten dalam lingkup aplikasi Web. Kerawanan dalam aplikasi Web mungkin digunakan dalam suatu serangan *phishing*.

5. Membuat konten Web tambahan sebagai identifikasi atas situs web yang sah.

6. Menggunakan otentikasi berbasis *token* atau otentikasi dua arah (*mutual authentication*) pada situs Web untuk mencegah para pelaku *phis*ing dari penggunaan kembali informasi

otentikasi sebelumnya untuk berpura-pura sebagai pengguna.

Saat mempertimbangkan tindakan-tindakan anti-*phishing*, mempertimbangkan tipe informasi yang menjadi *host* pada situs Web merupakan hal yang penting. Situs Web yang menyimpan *Personally Identifiable Information*/PII harus mempertimbangkan dengan cermat untuk mengimplementasikan tindakan anti-*phishing* yang lebih kuat. Situs Web dengan informasi publik yang tidak sensitif tidak perlu mengimplementasikan tindakan anti-*phishing* yang lebih canggih yang menghabiskan banyak biaya.

5.2.2 *Pharming*

Para penyerang *pharming* lebih menggunakan pendekatan teknis daripada rekayasa sosial, untuk mengarahkan para pengguna agar mengakses suatu situs Web palsu yang berpura-pura sebagai suatu situs Web yang sah dan memdapatkan rahasia informasi pribadi. *Pharming* biasanya dilakukan dengan mengeksploitasi kerawanan dalam perangkat lunak DNS yang digunakan untuk mengubah nama domain Internet yang dapat dibaca orang menjadi alamat IP, ataupun dengan mengubah *file-file host* yang dikelola pada suatu komputer klien untuk mengubah nama domain Internet. Teknik-teknik yang bervariasi dapat membantu mengurangi kemungkinan menjadikan seorang pengguna situs Web terlibat dalam suatu serangan *pharming*:

1. Menggunakan Versi Terkini Perangkat Lunak DNS yang Mengaplikasikan *Patches* Keamanan Terkini — Suatu server DNS yang bobol akan mengijinkan penyerang untuk mengarahkan pengguna kepada suatu server yang berniat jahat sambil tetap mempertahankan suatu nama DNS yang resmi.
2. Mengimplementasikan Proteksi Server-Side DNS Terhadap Pharming — Ada beberapa tool yang tersedia untuk mengurangi ancaman terhadap perangkat lunak DNS, seperti *DNS Security Extensions*

3. Mengawasi Domain Organisasi dan Registrasi Domain-domain yang Serupa — Serangan-serangan *pharming* mungkin mengambil keuntungan dari para pengguna yang salah mengeja nama domain organisasi saat mengakses situs.

4. Menyederhanakan Struktur dan Jumlah Nama Domain Organisasi — Struktur penamaan domain bagi pemerintah mengikuti ketentuan yang berlaku berbasis domain go.id. Namun banyak domain pemerintah yang telah menyederhanakan struktur domainnya seperti misalnya http://www.pajak.go.id

5. Menggunakan Koneksi yang Aman (yaitu, HTTPS) untuk *Log*in, yang dapat memverifikasi Keabsahan Sertifikat Server dan Keterkaitannya dengan Suatu Situs Web yang Sah — Browser yang modern akan memberitahukan seorang pengguna jika nama DNS tidak sesuai dengan yang diberikan oleh sertifikat, namun beberapa situs *pharming* dapat memiliki suatu sertifikat yang sah.

6. Memastikan Kewaspadaan Pengguna akan Bahaya Serangan *Pharming* dan Bagaimana Menghindarinya — *Pharming* merupakan suatu fenomena baru dan belum banyak diketahui apa yang harus diperhatikan dalam hal serangan *pharming*.

7. Memverifikasi Resolusi *Host* Pihak Ketiga — Sejumlah vendor menyediakan plug-in Web browser yang mendukung pencocokan alamat Internet Protocol (IP) dari suatu situs Web, sehingga dapat memberikan suatu peringatan kepada para pengguna jika situs Web mencurigakan.

8. Menggunakan *Pre-shared Secret* — *Pre-shared secret* dapat digunakan untuk melawan serangan *pharming*. Suatu implementasi yang umum dari *pre-shared secret* adalah agar para pengguna yang sah membuat pertanyaan tertentu dan menjawab dengan apa yang seharusnya mereka tahu.

5.3. Pengamanan Active Content dan Tekno*log*i Pembangkit Konten

5.3.1. Tekno*log*i *Active Content*

Active content mengacu pada pada elemen program interaktif yang di download ke klien (Web browser) dan diproses di tempat klien. Ada berbagai variasi tekno*log*i konten yang aktif; contoh yang populer adalah ActiveX, Java, VBScript, JavaScript, dan Asynchronous JavaScript dan XML (AJAX). Penggunaan active content seringkali mensyaratkan para pengguna untuk mengurangi pengaturan keamanan pada Web browser. Jika tidak diimplementasikan dengan tepat, *active content* dapat menghadirkan suatu ancaman yang serius pada *end user*. Sebagai contoh, *active content* dapat bertindak secara independen tanpa sepengetahuan atau ijin dari pengguna. Disamping menciptakan problem beresiko kepada klien, *active content* juga dapat menciptakan problem beresiko kepada Web server. Oleh karena itu, hasil dari pemrosesan yang dilakukan pada klien melalui elemen *aktif konten* sebaiknya tidak dipercaya oleh server; dan perlu dilakukan verifikasi hasil oleh server.

5.3.2. Tekno*log*i Pembangkit Konten *Server-Side*

Pembangkit konten merupakan suatu program pada Web server yang secara dinamis menghasilkan halaman-halaman HTML untuk para pengguna; halaman-halaman tersebut mungkin dihasilkan dengan menggunakan informasi yang diambil dari suatu server *back end*.

Tekno*log*i pembangkit konten si sisi server antara lain CGI, ASP .NET, Java EE, dan interface-interface server lainnya. Penggunaan umum eksekusi sisi server termasuk akses database, aplikasi e-commerce/e-government, chat room dan lain sebagainya.

Pembangkit konten sisi server dapat menciptakan kerawanan keamanan berikut ini pada server:

1. Sengaja atau tidak dapat membocorkan informasi tentang aplikasi Web server dan OS *host* yang dapat membantu seorang penyerang, sebagai contoh, dengan mengijinkan akses ke informasi di luar wilayah yang dirancang untuk kegunaan Web.
2. Ketika memproses input yang diberikan pengguna, misalnya seperti konten dari suatu formulir, parameter URL, atau *search query* (daftar tunggu pencarian), hal-hal itu mungkin rentan untuk diserang dengan jalan mana pengguna menyiasati aplikasi untuk mengeksekusi perintah sembarang yang diberikan dalam rangkaian input (misalnya script lintas situs atau injeksi SQL).
3. Memungkinkan para penyerang untuk *deface* (merubah tampilan halaman web) atau memodifikasi konten situs.

CGI script adalah mekanisme awal yang digunakan untuk membuat situs Web berinteraksi dengan database dan aplikasi-aplikasi lain. Walaupun demikian, karena Web berkembang, metode pemrosesan sisi server telah dikembangkan menjadi lebih efisien dan mudah diprogram; sebagai contoh, Microsoft menyediakan ASP.NET untuk IIS servernya, Sun/Netscape mendukung Java servlet, dan *freeware* PHP didukung oleh sebagian besar platform Web terkemuka termasuk Apache dan IIS. Beberapa hal penting untuk dipertimbangkan ketika memikirkan penyiapan CGI :

1. Sistem *file host* menyediakan keamanan bagi CGI.
2. Sebagian besar server mengijinkan pembatasan CGI per direktori.
3. CGI itu sendiri menyediakan pemberlakuan keamanan sedikit.
4. Perl (*Practical Extraction and Report Language)* memfasilitasi pemrograman aman yang sebagian besar bahasa yang lain (misalnya C, C++, sh) tidak melakukannya.

5. CGI *wrapper* yang tersedia dari pihak ketiga menawarkan proteksi tambahan bagi CGI.

Server Side Includes (SSI) adalah suatu bahasa script sisi server terbatas yang didukung oleh sebagian besar Web server. SSI menyediakan seperangkat fitur dinamis, termasuk waktu saat ini atau tanggal modifikasi terakhir dari *file* HTML, sebagai suatu alternatif dari penggunaan suatu program CGI untuk menjalankan fungsi tersebut. Beberapa hal yang penting untuk dipertimbangkan ketika memikirkan penyebaran SSI:

1. Keamanan SSI sangat lemah jika perintah exec dimungkinkan pada Web server.
2. Dampak SSI dapat merugikan performa Web server yang bebannya berat.
3. Keamanan SSI sangat bergantung pada OS *host* dan aplikasi Web server untuk keamanan.

Microsoft ASP.NET adalah tekno*log*i script sisi server dari Microsoft yang dapat digunakan untuk mencipatkan aplikasi Web yang dinamis dan interaktif. Suatu halaman ASP berisi script sisi server yang bekerja ketika suatu browser meminta suatu sumber ".asp" dari server Web. Berikut ini adalah beberapa hal penting untuk dipertimbangkan ketika memikirkan penyebaran ASP.NET:

1. ASP.NET sangat bergantung pada OS *host* dan aplikasi Web server dalam hal keamanan.
2. Keamanan klien diintegrasikan sangat baik dengan server Web dan layanan otentikasi OS *host*.
3. ASP.NET mendukung Microsoft Code Access Security, yang menyediakan metode-metode untuk pengembang konten atau administrator untuk membatasi *privilege.*
4. ASP.NET relatif kebal terhadap banjir *buffer.*

5. ASP.NET didokumentasikan sangat baik dan tekno*log*i yang matang.

Java EE berbasikan tekno*log*i Java dan menyediakan suatu tipe *applet* sisi server yang disebut *servlet.* Web server terrlebih dahulu menentukan apakah permintaan browser memerlukan informasi yang dibangkitkan secara dinamis dari suatu *servlet*, yang memproses permintaan dan membangkitkan respon HTTP, suatu Java Server Page (JSP), atau suatu halaman HTML statis.

Beberapa hal penting untuk dipertimbangkan ketika memikirkan penyebaran servlet Java :

1. Java EE diintegrasikan ketat dengan keamanan OS *host* dan otentikasi Web server untuk keamanan yang kuat.
2. Java EE memfasilitasi pemrograman yang aman dengan
    a) Meningkatkan keamanan bahasa pemrograman Java
    b) Menggunakan suatu model keamanan yang kuat yang mendukung pembatasan oleh para pengembang dan administrator server
    c) Menerapkan penanganan kesalahan yang aman.
3. Java EE terdokumentasi dengan baik dan merupakan tekno*log*i yang matang.
4. Sejumlah besar kode pihak ketiga dari IBM, Sun, Apache Foundation, dan para pengembang lain tersedia untuk digunakan dengan Java EE.

PHP adalah suatu bahasa script yang digunakan untuk menciptakan halaman Web dinamis. Dengan syntax dari C, Java, dan Perl, kode PHP ditambahkan ke dalam halaman HTML untuk eksekusi sisi server. Sebagian besar server Windows dan Unix mendukung bahasa tersebut dan banyak digunakan bersama database mySQL. Berikut adalah beberapa hal penting untuk dipertimbangkan ketika

menggunakan PHP:

1. Versi terbaru PHP yang ada sebaiknya digunakan karena versi yang lebih lama memiliki sejumlah kerawanan keamanan.
2. PHP menyediakan sejumlah opsi yang menyederhanakan pengembangan; beberapa dari opsi ini (misalnya opsi *register_globals* yang mengkonversikan semua parameter input kedalam variabel PHP dan boleh mengganti nilai dalam script PHP) dapat mempersulit para pemula untuk membuat program-program yang aman.
3. Banyak sekali kode pihak ketiga yang tersedia bebas untuk PHP ditulis dengan buruk dari perspektif keamanan.Ada sejumlah alasan untuk keamanan yang buruk dari banyak script PHP. Yang paling kelihatan adalah banyak script yang ditulis tanpa perhatian terhadap keamanan. Sebagai tambahan, karena relatif nyamannya coding script PHP, banyak pemula yang memiliki sedikit pengetahuan tentang pemrograman yang aman, menciptakan dan sering mendistribusikan secara bebas script yang ditulis dengan buruk.

5.3.3. Pertimbangan Keamanan Pembangkit Konten *Server-Side*

Ketika memeriksa atau menulis suatu *active content* yang dapat dieksekusi atau script, pertimbangkan hal-hal berikut:

1. Kode yang dapat dieksekusi sebaiknya sesederhana mungkin. Semakin panjang atau kompleks semakin mungkin untuk memiliki problem.
2. Kemampuan kode yang dapat dieksekusi untuk membaca dan menulis program sebaiknya dibatasi. Kode yang membaca *file* memungkinkan melanggar pembatasan akses dengan kurang hati-hati atau meneruskan informasi sistem yang sensitif. Kode yang menulis *file* dapat memodifikasi atau merusak dokumen atau memunculkan *Trojan horse.*

3. Interaksi kode dengan program atau aplikasi lain sebaiknya dianalisa untuk mengidentifkasi kerawanan keamanan. Sebagai contoh, banyak script CGI mengirim email sebagai tanggapan atas input formulir dengan membuka suatu koneksi dengan program *sendmail*. Pastikan interaksi ini dijalankan dengan suatu cara yang aman.

4. Pada *host* Linux/Unix, kode sebaiknya tidak berjalan dengan suid (set-user-id).

5. Kode harus menggunakan nama path eksplisit ketika meminta program eksternal. Bersandar pada variabel lingkungan PATH untuk menyelesaikan nama-nama *path* parsial tidak direkomendasikan.

6. Web server sebaiknya di-*scan* secara berkala dalam hal kerawanan, sekalipun jika server-server tersebut tidak menerapkan active content. Scanner keamanan jaringan dapat mendeteksi kerawanan dalam server Web, OS, atau layanan-layanan lain pada Web server. Scanner kerawanan aplikasi Web secara khusus men-scan dalam hal kerawanan pembangkit konten (lihat Lampiran C untuk lebih banyak informasi).

7. Kode pembangkit konten Web sebaiknya di scan dan/atau di audit (tergantung pada sensitifitas dari Web server dan kontennya). *Tool* yang ada secara komersil dapat men-scan .NET atau kode Java. Sejumlah entitas komersil menawarkan layanan peninjauan kembali kode.

8. Kode pembangkit konten Web sebaiknya dikembangkan mengikuti yang direkomendasikan berdasarkan keamanan terkini.

9. Untuk formulir *data entry*, tentukan suatu daftar karakter yang diharapkan dan memfilter karakter yang tidak diharapkan dari data input yang dimasukkan oleh seorang pengguna sebelum memproses suatu formulir. Sebagai contoh, pada sebagian formulir, data yang diharapkan digolongkan dalam

kategori: huruf a-z, A-Z, dan 0-9. Kehati-hatian harus dilakukan ketika menerima karakter-karakter khusus semacam &,’,”,@,dan !. Simbol-simbol ini mungkin memiliki arti khusus dalam bahasa pembangkitan konten atau komponen lain dari aplikasi Web.

10. Pastikan bahwa halaman yang dibangkitkan secara dinamis tidak berisi meta karakter berbahaya. Dimungkinkan bagi seorang pengguna yang berniat jahat untuk menempatkan tag-tag ini dalam database atau suatu *file*. Ketika suatu halaman dinamis dihasilkan menggunakan data yang diubah, kode yang berniat jahat yang ditambahkan dalam tag mungkin diteruskan ke klien browser. Selanjutnya browser pengguna dapat diperdaya untuk menjalankan suatu program yang dipilih penyerang. Program ini akan mengeksekusi dalam konteks keamanan browser untuk berkomunikasi dengan Web server yang sah, bukan konteks keamanan browser untuk berkomunikasi dengan penyerang. Jadi, program tersebut akan mengeksekusi dalam suatu konteks keamanan yang tidak tepat dengan hak istimewa akses yang tidak tepat.

11. Perangkat karakter pengkodean harus diatur secara eksplisit dalam setiap halaman. Selanjutnya data pengguna harus di *scan* dalam hal rangkaian byte yang merepresentasikan karakkter khusu untuk skema pengkodean yang diketahui.

12. Setiap karakter dalam perangkat karakter tertentu dapat dikodekan menggunakan nilai numeriknya. Pengkodean output dapat digunakan sebagai suatu pengganti bagi pem-filteran data. Pegkodean menjadi penting secara khusus ketika karakter khusus semacam simbol copyright dapat menjadi bagian dari data dinamis. Meskipun demikian, pengkodean data dapat diberdayakan intensif, dan suatu keseimbangan harus ditemukan antara pengkodean dan metode lain untuk mem-filter data.

13. *Cookie* sebaiknya diperiksa untuk karakter khusus apapun dan

karakter tersebut sebaiknya di-filter.

14. Suatu mekanisme enkripsi sebaiknya digunakan untuk mengenkripsi *password* yang masuk berbentuk script .

15. Untuk aplikasi web yang dibatasi oleh *username* dan *password*, tak satupun dari halaman Web dalam aplikasi yang semestinya dapat diakses tanpa mengeksekusi proses *login* yang tepat.

16. Banyak Web server dan perangkat lunak Web server lain menggunakan script sampel atau dapat dieksekusi selama proses instalasi. Beberapa diantaranya telah diketahui kerawanan dan sebaiknya dihilangkan segera. Lihat dokumentasi manufaktur yang sesuai atau Web site untuk informasi lebih banyak.

5.3.4 Lokasi Pembangkit Konten Server-Side

Lokasi *active content* pada Web server adalah penting. Jika berlokasi dalam direktori yang salah atau dalam direktori dengan ijin yang salah, masalah lokasi akan segera mengarah ke bobolnya Web server. Untuk menghindari hal ini :

1. *File* yang dapat ditulis sebaiknya diidentifikasi dan ditempatkan dalam folder terpisah. Tidak ada *file* script yang muncul dalam folder yang dapat ditulis. Sebagai contoh, data buku tamu biasanya disimpan dalam *file-file* teks sederhana. *File-file* ini membutuhkan ijin untuk menulis bagi *guest* untuk dapat memberikan komentarnya

2. *File-file* yang dapat dieksekusi (misalnya CGI, .EXE, .CMD, dan PL) sebaiknya ditempatkan dalam folder terpisah. Tidak ada dokumen yang dapat dibaca dan dapat ditulis lainnya yang harus ditempatkan dalam folder-folder ini.

3. *File-file* script (misalnya ASP, PHP, dan PL) sebaiknya memiliki folder terpisah. Mungkin menguntungkan pula untuk menyimpan script-script dalam suatu folder dengan nama yang

tidak nyata terlihat (misalnya bukan ”Script”) untuk membuatnya lebih sulit bagi seorang penyerang untuk menemukan script melalui browsing langsung.

4. Cakupan *file-file* (misalnya INC, SHTML, SHTM dan ASP) yang diciptakan untuk penggunaan kembali kode harus ditempatkan dalam direktori terpisah. SSI harus tidak digunakan secara umum pada Web server. *File* cakupan ASP harus memiliki suatu ekstensi .asp dan bukan .inc. Catat bahwa sebagian besar resiko dalam hal *file* cakupan berada dalam kapabilitas menjalankannya. Jika kapabilitas menjalankannya dilumpuhkan, resiko ini berkurang secara drastis.

5.3.5 Kerawanan Script Lintas Situs

Cross-site scripting (XSS) adalah suatu kerawanan yang umumnya ditemukan dalam aplikasi Web interaktif yang mengijinkan injeksi kode oleh para pengguna Web *malicious* kedalam halaman Web yang dilihat oleh para pengguna lainnya. Biasanya muncul dalam halaman Web yang tidak melakukan batasan pengetesan yang sesuai terhadap data yang dimasukkan oleh para pengguna. Kerawanan XSS yang sudah tereksploitasi dapat digunakan oleh para penyerang untuk membobol komputer para pengguna lainnya atau untuk menerima data dari sesi Web pengguna lain (misalnya user ID dan *password* atau sesi *cookies)*.

Kerawanan XSS sangat bervariasi dan seringkali unik terhadap suatu aplikasi Web tertentu. Kerawanan XSS meliputi dua kategori umum:

1. ***Persistent XSS***(kerawanan XSS yang menetap) memungkinkan serangan yang lebih hebat. Kerawanan ini terjadi ketika data yang diberikan ke suatu aplikasi Web oleh pengguna yang tidak dapat dipercaya disimpan secara tetap pada server dan ditayangkan ke para pengguna lain sebagai konten Web namun tidak divalidasi atau disandikan menggunakan HTML. Contoh umum dari kerawanan XSS yang menetap adalah *online*

message boards, wikis dan *blogs* dimana para pengguna diijinkan untuk menempatkan suatu pesan berformat HTML untuk dilihat para pengguna lain. Dalam contoh ini, setelah seorang pengguna *malicious* menempatkan suatu pesan atau jawaban *malicious*, pengguna lain manapun yang mengakses suatu halaman yang menayangkan data itu dan yang browsernya rentan terhadap eksploitasi, dapat dibobol.

2. ***Non Persistent XSS*** (kerawanan XSS yang tidak tetap), terkadang disebut terefleksikan, lebih umum dan sedikit kurang berbahaya dibanding kerawanan yang tetap. Kerawanan tidak tetap muncul ketika data yang diberikan klien Web digunakan segera oleh script sisi server untuk membangkitkan suatu halaman hasil bagi pengguna tersebut (misalnya *login screen,* halaman *search result*). Jika data yang diberikan klien yang tidak divalidasi diikutsertakan dalam *returned page* tanpa pengkodean HTML apapun, ini akan memungkinkan kode sisi klien diinjeksikan kedalam halaman dinamis. Kondisi ini mungkin tidak tampak sebagai problem di permukaannya karena seorang penyerang dapat hanya mengeksploitasi dirinya sendiri. Namun demikian, seorang penyerang dapat mengirimkan suatu URL yang dirancang khusus ke seorang pengguna dan meperdaya pengguna melalui rekayasa sosial agar meng klik pada URL yang dirancang *malicious*. Jika Web browser pengguna rentan terhadap eksploitasi, mesin pengguna dapat dibobol. Karena serangan ini memerlukan semacam rekayasa sosial, dianggap kurang berbahaya dibandingkan dengan kerawanan yang tetap.

Solusi untuk serangan XSS adalah memvalidasi semua input pengguna dan membuang data yang tidak diharapkan atau yang beresiko secara potensial. Solusi lain adalah menggunakan suatu versi HTML-quoted dari input pengguna manapun yang diberikan kembali ke pengguna lain. Hal ini akan mencegah Web browser para pengguna lain dari menterjemahkan input tersebut dan berlaku sebagaimana adanya perintah yang ditambahkan.

6. Penggunaan Teknik Otentikasi dan Enkripsi Pada Web Server

6.1. *Brute Force Attack.*

Ada banyak W*eb site* yang membuktikan otentikasi penggunanya melalui kombinasi *username* dan *password* baik itu melalui HTTP dasar, HTTP *digest*, atau bentuk W*eb* melalui SSL. Tanpa menghiraukan pengimplementasiannya, kombinasi *username* dan *password* dapat sangat rawan terhadap *brute force attack*. *Brute force attack* dapat terjadi dalam bentuk sebagai berikut :

1. ***Username Harvesting.*** Aplikasi yang membedakan antara suatu invalid *password* dan invalid *username* dapat memperbolehkan penyerang untuk membuat suatu daftar akun pengguna yang valid.
2. ***Dictionary attack.*** Para penyerang menggunakan kata-kata umum yang terdapat dalam kamus dan variannya untuk mencoba mendapatkan akses terhadap akun pengguna.
3. ***Brute force attack.*** Para penyerang mencoba setiap *password* yang mungkin untuk mendapatkan akses dari akun pengguna.

Ada sejumlah metode yang dapat dilakukan pada Web Server untuk mengurangi kerawanan terhadap *brute force attack* yaitu :

1. **Gunakan Otentikasi yang Kuat.** Teknik-teknik otentikasi yang kuat, seperti *perangkat keras token*, *one-time password*, otentikasi biometri, dan sertifikat klien SSL/TLS, akan lebih tahan terhadap *brute force attack* daripada *password*. Otentikasi yang lebih kuat dapat diperoleh dengan mengkombinasikan beberapa mekanisme otentikasi untuk membentuk suatu skema otentikasi multi-faktor. Akan tetapi, otentikasi yang kuat mungkin saja menjadi terasa mahal atau

sulit digabungkan ke dalam suatu sistem.

2. **Gunakan *Timeout.*** Melakukan suatu penundaan beberapa detik setelah suatu *log*in yang gagal dapat memperlambat penyerangan. Namun, penyerang dapat berusaha melakukan *log*in ganda pada saat yang sama dari klien yang berbeda.

3. **Gunakan *Lockout.*** Mengunci akun pengguna setelah serangkaian *log*in yang gagal mencegah penyerang melakukan *log*in ke suatu akun. Kelemahan utama dari teknik ini yaitu dapat membuat sistem terbuka untuk serangan DoS. Seorang penyerang dapat mencoba beberapa *password* yang umum terhadap *username* yang acak, yang kemungkinan membawa penyerang untuk dapat mengakses sistem dan melewati *lockout.*

4. **Berlakukan Kebijakan *Password.*** Dengan mempersyaratkan suatu *password* memiliki panjang tertentu dan terdiri dari huruf kecil, huruf besar, angka, dan/atau simbol, maka suatu *dictionary attack* sederhana tidak akan dapat dijalankan pada sistem.

5. **Berlakukan Suatu Kebijakan Merubah *Password.*** Dengan mempersyaratkan terhadap suatu *password* yang harus dirubah secara teratur, seorang penyerang akan menghabiskan lebih banyak waktu untuk melakukan *brute force attack*. Namun, kebijakan merubah *password* yang terlalu ketat dapat membuat pengguna frustrasi dan menghasilkan *password* yang lemah karena pengguna mengikuti pola tertentu, seperti misalnya menggunakan *password*1, *password*2, dan sebagainya.

6. **Gunakan *Blacklist.*** Upaya mengantisipasi dan menghentikan penyerangan terhadap system dengan memblokir alamat-alamat IP atau domain yang diketahui berusaha melakukan *brute force attack*, tetapi ada peluang bahwa beberapa serangan mungkin berasal dari sistem yang bobol yang sebaliknya akan dianggap sebagai sistem yang sah.

7. **Gunakan Perangkat Lunak Monitoring *Log*.** Monitoring dengan seksama *log-log* dalam hal percobaan *password* yang tidak sah dapat membantu suatu organisasi dalam mendeteksi *brute force attack*, secara potensial dapat memberikan organisasi waktu untuk merespon suatu serangan sebelum serangan tersebut berhasil dilakukan.

Mekanisme penggunaan otentikasi multi faktor mulai dari parameter username dan password, sertifikat elekronik sampai *biometric system* perlu dipertimbangkan untuk mempersulit penyerang mendapatkan akses ke suatu sistem.

6.2. Metode Otentikasi pada Web Server

Metode Otentikasi yang dapat digunakan pada web server antara lain:

1. Otentikasi Address-Based

   Mekanisme otentikasi yang paling sederhana yang didukung oleh sebagian besar Web server adalah otentikasi *address-based* (berbasis alamat). Kendali akses berdasarkan pada alamat IP dan/atau nama *host* dari *host* yang meminta informasi. Otentikasi ini rentan terhadap beberapa jenis serangan termasuk IP *spoofing* dan DNS *poisoning*. Jenis otentikasi ini seharusnya digunakan hanya ketika keamanan minimal yang diperlukan, kecuali jika digunakan bersamaan dengan metode otentikasi yang lebih kuat.

2. Otentikasi dasar

   Tekno*log*i otentikasi dasar menggunakan struktur direktori konten Web server. Secara khusus, semua *file* dalam direktori yang sama dikonfigurasikan untuk dapat diakses dengan privilege user yang sama. Seorang pengguna yang meminta layanan memberikan suatu identifikasi pengguna yang dikenali dan *password* untuk akses ke *file* di direktori yang sudah

ditentukan. *vendor* perangkat lunak Web server memiliki metode dan sintaksis (*syntax)* masing-masing guna mendefinisikan dan menggunakan mekanisme otentikasi dasar ini.Otentikasi dasar didukung oleh Web browser yang *standard-compliance* dan bermanfaat untuk memproteksi informasi terhadap *malicious bots* karena *bot* tidak memiliki bukti terpercaya yang diperlukan untuk mengakses direktori yang diproteksi. Namun demikian, mekanisme ini sebaiknya tidak dianggap aman terhadap serangan yang lebih canggih.

Dari sudut pandang keamanan, kekurangan tekno*log*i ini adalah bahwa semua *password* yang ditransfer hanya dikodekan dan bukan dienkripsikan. Siapapun yang mengenal skema pengkodean yang menjadi standarnya dapat membuka kode *password* setelah menangkapnya dengan metode *sniffer* jaringan. Terlebih lagi konten Web ditransmisikan sebagai teks terang yang tidak terenkripsi, jadi konten inipun dapat disadap dan diketahui oleh pihak yang tidak berwenang Keterbatasan ini dapat diatasi menggunakan otentikasi dasar yang disertai dengan SSL/TLS.

3. Otentikasi Digest

Karena kekurangan dari otentikasi dasar, teknik yang lebih tinggi yang dikenal dengan otentikasi *digest* diperkenalkan dalam protokol HTTP versi 1.1. Otentikasi *digest* menggunakan mekanisme *challenge-response* untuk otentikasi pengguna. Dengan pendekatan ini, suatu *nonce atau* bilangan acak sembarang dikirim ke pengguna, yang mempunyai ID dan *password* sebagaimana pada otentikasi dasar. Dalam kasus ini, informasi yang dimasukkan oleh pengguna diproses dengan teknik *hash* kriptografis. Nilai hash ini digabungkan dengan nonce dan nilai hash dari metode dan URL yang diminta, lalu hasilnya diproses hash lagi menjadi response value yang dikirim ke server.

Password pengguna yang dikirim juga sudah tidak dalam bentuk teks terang dan nilai hash dari ID dan password inilah

yang akan digunakan untuk mengotentikasi pengguna. Dan karena *nonce* dapat dibuat dari informasi tanggal dan jam terkini, *replay attack* juga dihindari. Jadi otentikasi *digest* lebih aman daripada otentikasi dasar. Seperti halnya otentikasi dasar, otentikasi *digest* berguna untuk memproteksi informasi dari *bots.* Tetapi semua data lain dikirim dalam bentuk teks terang (tidak terenkripsi), dan data ini rentan terhadap penyadapan dan pengubahan. Keterbatasan ini dapat diatasi menggunakan otentikasi *digest* yang disertai dengan SSL/TLS Dan *offline dictionary attack* dapat terjadi pada *password* otentikasi *digest* yang tersadap. *Password* otentikasi *digest* yang tersadap yang dikirim melalui koneksi yang diproteksi SSL tidak rentan terhadap *offline dictionary attack.*

6.3. SSL/TLS

Otentikasi antara klien web browser dan web server serta enkripsi pada jalur komunikasi dapat dilakukan menggunakan protokol SSL dan TLS SSL pertama kali diperkenalkan oleh Netscape Communication tahun 1994 dan telah direvisi dua kali. SSL versi 3 merupakan versi terakhir, versi sebelum 3.0 tidak aman dan sebaiknya tidak digunakan lagi. Pada tahun 1996, Internet *Engineering Task Force* (IETF) mendirikan kelompok kerja TLS untuk memformalkan dan meningkatkan protokol SSL hingga level standar Internet. Protokol TLS versi 1.0 dispesifikasi secara resmi dalam IETF sebagai *Request for Comment* (RFC) 2246. yang dipublikasikan pada tahun 1999 dan sebagian besar berbasiskan SSL versi 3. SSL versi 3 dan TLS versi 1 pada dasarnya identik dan dibahas secara bersamaan dalam dokumen ini. Mayoritas Web browser, mendukung penggunaan SSLv3, TLS 1.0. dan TLS 1.1, yang dispesifikasikan dalam RFC 4436 yang diterbitkan, pada bulan April 2006.

6.3.1. Kapabilitas SSL/TLS

Kapabilitas SSL/TLSHTTP dan protokol layer aplikasi lain adalah sebagai berikut :

1. **Otentikasi Server** — dengan SSL/TLS maka suatu klien Web browser mengkonfirmasikan identitas suatu Web server. Klien Web browser yang mengaktifkan SSL/TLS dapat memeriksa bahwa suatu nama server dan kunci publik terkandung dalam suatu sertifikat adalah sah diterbitkan oleh suatu CA yang terdaftar dalam daftar klien CA yang terpercaya. Informasi ini menjadi sangat penting jika pengguna tersebut, misalnya mengirimkan nomor kartu kredit melalui jaringan dan ingin mengkonfirmasikan identitas server yang menerima.

2. **Otentikasi Klien** — SSL/TLS memungkinkan suatu Web server mengkonfirmasikan suatu identitas pengguna menggunakan teknik yang sama dengan otentikasi server dengan cara sebaliknya.Informasi ini mungkin penting jika server tersebut, misalnya, merupakan suatu bank yang sedang mengirimkan informasi keuangan rahasia kepada seorang pelanggan dan menginginkan untuk mengkonfirmasi identitas penerima. Jika otentikasi klien harus diselenggarakan maka otentikasi server harus juga diselenggarakan. Sertifikat yang digunakan untuk otentikasi klien berbeda dengan sertifikat untuk otentikasi server, hal ini akan dijelaskan lebih lanjut pada segmen berikutnya. Namun demikian, otentikasi klien ini jarang diterapkan pada Web server dikarenakan *log*iktik yang terkait dalam penyediaan sertifikat klien bagi para pengguna dan perlu terinstal secara benar untuk dapat digunakan oleh Web browser.

3. **Enkripsi Komunikasi —** SSL/TLS dapat mengenkripsi sebagian besar informasi yang ditransmisikan antara suatu Web browser (klien) dan Web server atau bahkan antara dua Web server. Dengan algoritma enkripsi yang sesuai, SSL/TLS menyediakan tingkat konfidensialitas.Dan juga data yang dikirim melalui suatu koneksi SSL/TLS yang terenkripsi dilindungi dengan suatu mekanisme untuk mendeteksi

terjadinya perubahan pada saat transmisi dengan melakukan pemutusan koneksi secara otomatis.

6.3.2. Kelemahan SSL/TLS

Terdapat beberapa keterbatasan pada SSL/TLS.

1. Paket dienkripsi pada layer TCP, sehingga informasi layer IP tidak dienkripsi. Walaupun SSL/TLS melindungi data Web yang sedang ditransmisikan, seseorang yang memonitor suatu sesi SSL/TLS dapat menentukan pengirim dan penerima melalui informasi alamat IP yang tidak terenkripsi.
2. SSL/TLS hanya melindungi data pada saat ditransmisikan, bukan saat disimpan di kedua sisi. Jadi, data masih tetap rentan selama dalam penyimpanan (misalnya database kartu kredit) kecuali jika diterapkan keamanan pada media penyimpanan di kedua sisi.
3. Penggunaan SSL/TLS juga rentan terhadap serangan *man in the middle* yang dilakukan oleh malicious entitas yang berhasil menyadap komunikasi antara klien Web browser dan Web server, dimana klien berusaha untuk melakukan koneksi melalui SSL/TLS. Penyerang menyadap kunci-kunci yang sah yang dilewatkan bolak-balik selama proses sinkronisasi (*handshake)* dan menggantinya dengan kunci-kunci penyerang, membuat klien Web percaya bahwa penyerang adalah Web server dan sebaliknya bagi Web server penyerang tampak sebagai klien Web. Kondisi ini memungkinkan program penyerang tidak hanya membaca semua data yang mengalir antara klien Web dan Web server yang sesungguhnya namun juga memungkinkan mengganti data tanpa terdeteksi.

Sangatlah penting bagi para pengguna Web diberikan pembelajaran tentang bahayanya serangan sejenis "*man in the middle*" ini dan perlunya mengkonfirmasikan sertifikat sebelum menggunakan protokol SSL/TLS. Pemeriksaan dapat dilakukan dengan meng-klik

pada *padlock icon* (lambang kunci gembok) pada pojok kanan bawah atau kanan atas dari browser dimana icon ini hanya akan muncul pada saat mengakses suatu web server yang dilindungi dengan SSL/TLS. Ancaman ini dapat dikurangi jika sertifikat server diterbitkan oleh CA atau *self-signedcertificate* yang terpercaya Keberadaan suatu sertifikat *self-signed* mungkin merupakan indikasi bahwa suatu serangan *man-in-the-middle* sedang berlangsung. Browser dapat melakukan pemeriksaan secara otomatis, namun hal ini tidak dapat diandalkan.

Walaupun tanpa melakukan serangan *man-in-the-middle*, para penyerang dapat memperdaya para pengguna untuk mengakses suatu situs Web yang invalid. Ada beberapa kemungkinan metode serangan antara lain :

1. Membuat *self signed certificate* yang tidak jelas dan membuat user untuk menerimanya sebagai sertifikat yang benar. Meskipun web server dapat dikonfigurasi untuk menampilkan warning ketika terdapat *self signed certificate*, tetapi organisasi yang menggunakan *self signed certificate* biasanya menginstruksikan penggunanya untuk mengabaikan warning-warning yang ada.
2. Mengeksploitasi kerentanan dalam suatu browser Web sehingga situs Web nampak bagi seorang pengguna tak terlatih.
3. Mengambil keuntungan dari suatu kerawanan *cross-site scripting*pada suatu situs Web yang sah. Web Browser akan mengakses dua situs yang berbeda, namun bagi pengguna akan tampak bahwa hanya situs yang sah saja yang sedang diakses.
4. Mengambil keuntungan dari proses *certificate approval* untuk menerima suatu sertifikat yang sah dan menerapkannya bagi situs penyerang itu sendiri. Dengan menggunakan suatu sertifikat yang sah maka akan tampak seperti situs web yang sah dan pengguna haruslah menyadari bahwa situs yang diaksesnya adalah *malicious*.

Serangan *spoofing* SSL dapat terjadi tanpa memerlukan intervensi apapun dari pengguna. Sebagai contoh suatu nama domain organisasi/perusahaan terkenal yang didaftarkan oleh seseorang yang tidak sah maka situs yang dibangunnya akan tampak valid. Kemudian sertifikat SSL yang dibuat sesuai dengan nama domain tersebut, sehingga tidak muncul peringatanbagi pengguna Pada *address bar*, URL tampak identik dengan situs Web asli yang dijadikan obyek *spoofing.*

6.3.3. Teknik Implementasi SSL/TLS

Tahapan penerapan sertifikat elektronik yang ditandatangani oleh pihak ketiga yang terpercaya untuk suatu Web Server secara umum antara lain :

1. Membangkitkan dan mengajukan suatu permintaan penandatanganan sertifikat (*certificate-signing request*—CSR)
2. Mengambil sertifikat SSL/TLS yang ditandatangani dari CA
3. Menginstal sertifikat tersebut dan mengkonfigurasikan *Web server* untuk menggunakan SSL/TLS.

CSR terdiri dari tiga bagian: informasi permintaan sertifikat, pengidentifikasi algoritma penandatangan, dan tandatangan elektronik terhadap informasi permintaan sertifikat. Web Server yang menyediakan protokol SSL/TLS dapat memberikan instruksi untuk membangkitkan CSR. Terdapat dua tipe CSR :

1. Yang paling populer yaitu yang dikodekan dalam format Public Key Cryptography Standard (PKCS) #10 atau Certification Request Syntax Standard, yang banyak digunakan oleh Web Server versi terbaru.
2. CSR yang berbasis pada spesifikasi Privacy Enhanced Mail (PEM), biasa disebut Format PEM message header atau Web site professional, dan terbatas hanya pada Web Server versi lama.

Dengan meningkatnya penggunaan *phising* dan teknik-teknik lainnya untuk menipu para pengguna agar mengunjungi situs Web palsu atau malicious, ketidakberhasilan dalam membuktikan keaslian identitas server sebelum menggunakan SSL/TLS adalah tidak boleh terjadi. Jika suatu W*eb sever* tertentu hanya akan diakses dari sistem milik organisasi itu sendiri maka dapat diterapkan self signing root CA dan otentikasi Web server.

Dengan tidak tergantung pada penerbit dan format sertifikat SSL/TLS untuk W*eb server*, para administrator harus sangat berhati-hati dalam mengamankan sertifikat dan kunci privat miliknya. Berikut beberapa hal yang direkomendasikan:

1. Membuat dan menyimpan salinan sertifikat cadangan dalam media *read-only* untuk berjaga-jaga apabila sertifikat yang asli tidak sengaja terhapus. Jika sertifikat hilang dan tidak dapat ditemukan dalam media cadangan, maka suatu sertifikat baru harus dibuat.
2. Membuat dan menyimpan salinan kunci privat cadangan dalam media *read-only* untuk berjaga-jaga bila kunci tersebut tidak sengaja terhapus. Jika kunci tersebut hilang dan tidak dapat ditemukan dalam media cadangan, maka suatu pasangan kunci baru dan sertifikat harus dibuat. Sebagai catatan, salinan kunci privat harus diamankan secara fisik dan dienkripsi pula.
3. Menyimpan sertifikat yang asli dalam suatu folder atau partisi yang hanya dapat diakses oleh administrator *Web* atau sistem dan diamankan dengan mekanisme otentikasi yang sesuai.
4. Mempertimbangkan untuk menjalankan suatu pengecek integritas *file-file* dalam W*eb server* dengan menggunakan *Intrusion Detection Prevention System* (IDPS) yang melakukan pengawasan terhadap setiap perubahan yang terjadi pada sertifikat.
5. Memeriksa *log* sistem secara rutin untuk memvalidasi dan

mencegah terhadap akses yang tidak berhak terhadap sistem.

Jika seseorang yang dapat memperoleh akses yang tidak sah pada suatu web server dan berhasil mengubah pasangan kunci maka keseluruhan integritas server akan hilang seketika. Sekali suatu kunci dalam sertifikat SSL/TLS bocor maka selamanya akan tetap bocor dan beberapa CA tidak menerbitkan informasi mengenai pembatalan, sehingga banyak implementasi klien yang tidak memperoleh atau memproses informasi pembatalan.

Ketika suatu sertifikat telah siap, sertifikat tersebut perlu dipasang, dan SSL perlu diaktifkan dan dikonfigurasikan. Beberapa langkah yang umum untuk semua W*eb server* adalah sebagai berikut:

1. Matikan (*disable*) SSL 1.0 dan SSL 2.0.
2. Konfigurasikan SSL/TLS untuk membatasi algoritma kriptografi hanya *cipher suite* yang telah dipilih.
3. Indikasikan lokasi dari sertifikat SSL/TLS dan instruksikan kepada server untuk memulai menggunakan SSL/TLS. Dalam kasus tertentu, W*eb server* harus diinstruksikan untuk memulai penggunaan SSL/TLS, dan bahkan diberikan lokasi sertifikat SSl/TLS dan kunci privat jika disimpan sebagai *file* dalam suatu *hard drive*.
4. Instruksikan kepada server untuk membuka *port* 443 TCP. *Port* ini adalah *port* default untuk mengakses SSL/TLS.Dan jika tidak menggunakan SSL/TLS sebaiknya port ini ditutup untuk alasan keamanan. Selain port 443, karena web server adalah *host* untuk konten HTTP dan HTTPS maka port 80 harus dibuka juga.
5. Konfigurasikan server untuk melindungi sumber-sumber yang diperlukan (direktori dan/atau *files*) menggunakan SSL/TLS. Konfigurasikan aplikasi *Web server* sehingga sumber-sumber yang layak terlindung dengan SSL/TLS. Sumber-sumber ini hanya dapat diakses melalui suatu URL yang diawali dengan

”https://”.

Selain keamanan dengan SSL/TLS dilakukan, adalah penting untuk memastikan bahwa penerapan patch keamanan telah dilakukan. Kelemahan keamanan dari implementasi SSL/TLS dapat menjadi celah penyerang untuk melakukan *spoofing* terhadap sertifikat elektronik, memalsukan tandatangan elektronik, melakukan serangan DOS atau mengeksekusi sembarang kode dalam Web Server.

7. Implementasi Web Server pada Infrastruktur Jaringan Yang Aman

Infrastruktur jaringan yang mendukung *Web server* memainkan suatu peranan penting dalam keamanan *Web server*. Intensitas, kecanggihan, dan ragam serangan yang dilakukan saat ini menuntut keamanan *Web* harus diimplementasikan melalui mekanisme perlindungan yang berlapis dan beragam (*defense-in- depth)*. Bagian ini membahas komponen-komponen jaringan tersebut yang dapat mendukung dan melindungi Web server guna meningkatkan keamanan secara menyeluruh. Selain masalah keamanan, hal penting lainnya dan patut dipertimbangkan adalah biaya, performa dan kehandalan.

7.1. Komposisi dan Struktur Jaringan

7.1.1. *Layout* Jaringan Yang Tidak Disarankan

Beberapa Layout yang tidak disarankan adalah :

1. Meletakkan Web server pada jaringan *produksi* internal mereka. Web server berada pada jaringan yang sama dengan para pengguna internal dan server. Kelemahan utama dari *layout* ini adalah membuka komponen jaringan internal terhadap resiko tambahan. Web server seringkali menjadi target dari para penyerang. Jika penyerang berhasil membobol suatu Web server, mereka akan memiliki akses ke jaringan internal dan akan dapat dengan mudah membobol *host-host* internal. Oleh karena itu, *layout* seperti ini tidak direkomendasikan.
2. Meletakkan Web server didepan suatu*firewall* atau *router IP filtering.* Pada struktur ini*firewall* atau *router IP filtering* tidak dapat memberi perlindungan untuk Web server. Oleh karena itu

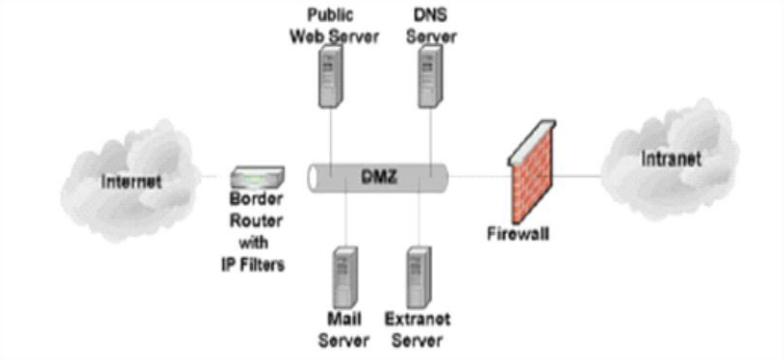
Web server harus menjaga keamanannya dan dapat terjadi *single point of failure*. Untuk menjadi aman di lokasi tersebut, OS Web server dan aplikasinya harus sangat diperhatikan keamanannya dimana segala layanan yang tidak perlu dan tidak aman dinonaktifkan/dimatikan dan menerapkan semua *patch* keamanan yang diperlukan.Untuk mempertahankan keamanan pengaturan tersebut, administrator Web server harus terus mengikuti perkembangan terakhir mengenai kerawanan dan *patch* yang terkait. Keterbatasan lain dari struktur ini adalah sulit untuk menyediakan semacam administrasi *remote* yang aman atau kemampuan untuk meng- update konten.

7.1.2. *Demilitarized Zone*

Demilitarized Zone [DMZ] menggambarkan suatu *host* atau suatu segmen jaringan yang disisipkan sebagai suatu ”zona netral” antara jaringan internal suatu organisasi dan Internet. Kondisi ini mencegah para pengguna diluar Web server memperoleh akses langsung kedalam jaringan internal suatu organisasi (intranet). Suatu DMZ mengurangi resiko dari penempatan suatu Web server pada suatu jaringan internal atau membukanya secara langsung ke Internet. Kondisi semacam itu adalah suatu solusi kompromi yang menawarkan banyak keuntungan dengan jumlah resiko yang sedikit. DMZ memperbolehkan akses ke sumber daya yang berada didalamnya baik untuk para pengguna internal maupun external. Ada banyak variasi konfigurasi DMZ, masing-masing dengan kekuatan dan kelemahannya, sebagai berikut :

1. DMZ Single-Firewall, suatu DMZ melibatkan penempatan suatu*firewall* diantara *border router* dari suatu organisasi dan jaringan internalnya, dan penciptaan suatu segmen jaringan baru yang hanya dapat dijangkau melalui peralatan DMZ. Web server diletakkan pada segmen yang baru, bersama dengan komponen infrastruktur jaringan lainnya dan *server-server* yang diperlukan untuk dapat diakses secara ekternal. Pada beberapa konfigurasi, *border router* itu sendiri dapat bertindak sebagai

firewall dasar. Gambar mengilustrasikan suatu contoh DMZ sederhana yang menggunakan suatu*router* dengan daftar pengendali akses (*Access control list/*ACL) untuk membatasi tipe tertentu dari lalu lintas jaringan dari dan ke DMZ.

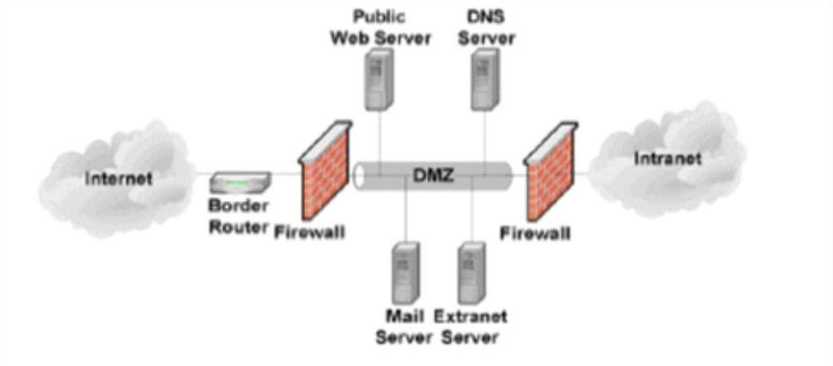


Gambar. Single-Firewall DMZ Sederhana

DMZ dengan firewall tunggal adalah suatu pendekatan biaya yang rendah karena suatu organisasi hanya perlu menambah suatu *firewall* dan menggunakan *border router* yang ada untuk menyediakan perlindungan kepada DMZ. Hal tersebut biasanya hanya tepat untuk digunakan pada suatu organisasi kecil yang menghadapi suatu ancaman minimal. Kelemahan mendasar pada pendekatan tersebut adalah bahwa meskipun *router* mampu melindungi terhadap sebagian besar serangan pada jaringan, tetapi *router* tidak "sadar" akan protokol *layer* aplikasi Web server (misal. HTTP) dan oleh sebab itu tidak dapat melindungi dari serangan *layer* aplikasi yang tertuju kepada Web server.

2. DMZ dua-*firewall* memperbaiki proteksi melalui suatu DMZ *router-firewall* karena firewall yang dikhususkan (*dedicated firewall)* tersebut dapat memiliki perangkat aturan keamanan yang lebih rumit dan kuat. Sebagai tambahan, karena *dedicated*

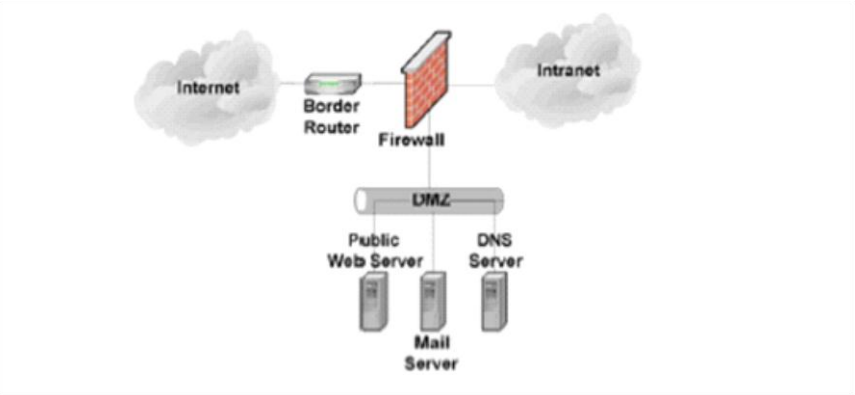
firewall tersebut seringkali mampu menganalisa lalu lintas HTTP yang masuk dan keluar, *firewall* dapat mendeteksi dan bertahan terhadap serangan *layer* aplikasi yang ditujukan ke Web server.



Gambar. DMZ Dua-Firewall

Tergantung pada pengaturan *firewall* dan tingkat lalu lintas yang diterima DMZ, DMZ tipe ini dapat mengakibatkan degradasi performansi.

3. DMZ Tiga-Interface Firewall, untuk organisasi-organisasi yang menginginkan pengamanan DMZ dua-*firewall* tetapi tidak memiliki sumber daya untuk membeli dua *firewall*, ada opsi lain yang disebut DMZ "*service leg*". Pada konfigurasi ini, suatu *firewall* dibangun dengan tiga (atau lebih) *interface* jaringan. Satu *interface* jaringan ditempelkan ke *border router*, *interface* yang lain ditempelkan ke jaringan internal dan yang ketiga dihubungkan ke DMZ, seperti pada gambar dibawah ini.



Gambar. DMZ Tiga-Interface Firewall

Konfigurasi ini menjadikan *firewall* sebagai subyek untuk peningkatan resiko dari degradasi layanan pada saat terjadi serangan DoS yang ditujukan pada DMZ. Pada konfigurasi jaringan DMZ firewall tunggal yang telah dibahas diatas, suatu serangan DoS terhadap Web server umumnya hanya mempengaruhi Web server. Pada konfigurasi jaringan DMZ *service-leg*, *firewall* sebagai penahan bagian terberat dari serangan DoS karena *firewall* harus memeriksa lalu lintas jaringan terlebih dahulu sebelum lalu lintas tersebut mencapai Web server, maupun sumber daya DMZ dan jaringan internal lainnya.Dapat terjadi peningkatan dari serangan DOS menjadi serangan DDOS dan menghabiskan semua *bandwidth* jaringan yang masuk dan perangkat terkait (misalnya Internet border routers) sebelum sempat mencapai firewall DMZ.

Keuntungan DMZ dari segi keamanan adalah sebagai berikut :

1. Web server dapat terlindungi dengan lebih baik, dan lalu lintas jaringan dari dan ke Web server dapat dipantau.
2. Bobolnya Web server tidak langsung mengancam jaringan

produksi internal.

3. Pengendalian yang lebih besar dapat disediakan melalui pengamanan dari Web server karena lalu lintas dari dan ke Web server dapat dikontrol.
4. Konfigurasi jaringan DMZ dapat dioptimalkan untuk mendukung dan melindungi Web server.

Kerugian DMZ dari segi keamanan sebagai berikut :

1. Serangan DoS yang tertuju kepada Web server dapat membawa pengaruh terhadap jaringan dalam.
2. Tergantung kepada konfigurasi *firewall* yang mengendalikan lalu lintas antara DMZ dan jaringan internal, dimungkinkan bagi Web server untuk digunakan menyerang atau membobol *host-host* pada jaringan internal. Dengan kata lain, perlindungan yang ditawarkan oleh DMZ sebagian besar bergantung pada konfigurasi firewall.

Untuk organisasi yang membangun Web servernya sendiri, suatu DMZ menjadi piihan terbaik. DMZ menawarkan perlindungan untuk Web server dan server-server yang dapat diakses secara ekternal lainnya tanpa membuka jaringan internal. Meskipun demikian, DMZ akan dianggap aman ketika digunakan bersama dengan langkah-langkah lainnya yang telah dibahas pada dokumen ini.

7.1.3. Pengelolaan Jaringan

Web server dan komponen penting lainnya dapat dihubungkan satu sama lain dan dikelola melalui suatu jaringan standar organisasi atau melalui suatu jaringan terpisah yang dikenal sebagai suatu jaringan manajemen. Konsol dan *host-host* lain yang digunakan untuk mengelola komponen Web ditempelkan hanya ke jaringan manajemen melalui *interface*. Tatanan ini secara efektif mengisolasi jaringan manajemen dari jaringan produksi. Manfaat dari tindakan

ini adalah untuk melindungi komponen-komponen dari berbagai serangan misalkan serangan DDoS. Kerugian dari penggunaan suatu jaringan manajemen adalah adanya biaya-biaya tambahan dari peralatan jaringan dan perangkat keras lainnya (misal komputer untuk konsol) dan ketidaknyamanan bagi para administrator komponen Web karena menggunakan komputer yang terpisah untuk pengelolaan dan pemantauan.

7.2. Konfigurasi Elemen Jaringan

Sekali Web server telah diletakkan pada jaringan, elemen infrastruktur jaringan harus dikonfigurasikan untuk mendukung dan melindunginya. Elemen utama dari suatu infrastruktur jaringan yang mempengaruhi keamanan Web server adalah *firewall, routers, IDS*, sistem pencegah gangguan (*intrusion prevention system/IPS), switch*, *load balancer (*penyeimbang beban) dan *reverse proxie,* yang masing-masing memainkan peranan peranan penting dan vital terhadap keseluruhan strategi perlindungan Web server melalui pertahanan berlapis. Sayangnya, ketika tiba saatnya untuk mengamankan suatu Web server, tidak ada solusi ”peluru perak” tunggal (satu solusi untuk semua masalah). Suatu*firewall* atau *IPS* saja tidak dapat memberikan perlindungan yang memadai bagi suatu Web server dari segala ancaman atau serangan.

7.2.1 Konfigurasi *Router/Firewall*

Terdapat beberapa jenis *firewall* antara lain :

1. *Router IP filtering*, adalah *firewall* yang paling mendasar yang dapat menyediakan kendali akses untuk paket IP.Suatu *firewall* (atau *router* yang bertindak sebagai suatu *firewall)* yang melindungi suatu Web server harus dikonfigurasikan untuk mem-blok semua akses ke Web server dari Internet kecuali *port* yang diperlukan, seperti TCP ports 80 (HTTP) dan 443 (HTTPS). Suatu *firewall* adalah garis pertahanan pertama bagi suatu Web server; meskipun demikian, untuk benar-benar

aman, organisasi perlu menerapkan perlindungan berlapis bagi Web server dan jaringannya. Lebih penting lagi, organisasi harus berusaha sekuat tenaga untuk mempertahankan semua sistem dalam keadaan aman dan tidak hanya sekedar bergantung pada *firewall*, *router*, atau segala macam komponen tunggal lainnya untuk menghentikan para penyerang.

2. *Stateful firewall,* adalah jenis *firewall* menengah yang dapat menyediakan kendali akses berbasis TCP dan *User Datagram Protocol* (UDP) sebaik IP. *Stateful inspection firewall* adalah peralatan *transport layer* yang memasukkan "kewaspadaan" status dari suatu koneksi TCP. *Firewall* jenis ini menjaga informasi internal, seperti misalnya status koneksi yang melaluinya dan konten dari beberapa aliran data. Kondisi ini memungkinkan ditentukannya seperangkat aturan yang lebih baik dan akurat serta penyaringannya. *Firewall* pemeriksaan keadaan menambahkan kemampuan untuk memberlakukan aturan berdasarkan status koneksi hingga kemampuan dari suatu *router* penyaring.

3. *Application Layer atau Proxy Firewall,* adalah firewall yang paling kuat yang mampu memahami dan menyaring konten Web.*Firewalllayer* aplikasi dianggap jenis *firewall* yang paling aman dan memiliki sejumlah keuntungan melebihi *routers* penyaring paket dan *firewall* pemeriksaan keadaan, termasuk hal-hal sebagai berikut :

    a) Kemampuan *log*ging

    b) Kemampuan menyaring (dapat menyaring tipe-tipe khusus dari konten Web dan perintah-perintah khusus HTTP)

    c) Memudahkan konfigurasi

    d) Kemampuan mengotentikasi pengguna

Untuk dapat berhasil melindungi suatu Web server dengan menggunakan suatu*firewall*, pastikan bahwa *firewall* di-*patch* ke

tingkat aman terakhir atau tingkat yang paling aman (baik aplikasi maupun OS yang mendasarinya) dan dikonfigurasi untuk melakukan hal-hal sebagai berikut :

1. Mengendalikan semua lalu lintas antara internet dan Web server
2. Memblokir semua yang ada didalam batas lalu lintas Web server kecuali yang diperlukan, misalnya seperti TCP ports 80 (HTTP) dan atau 443 (HTTPS)
3. Memblokir semua yang ada didalam batas lalu lintas dengan suatu alamat IP internal (untuk mencegah serangan IP *spoofing*)
4. Memblokir koneksi dari Web server ke Internet dan jaringan internal organisasi (ini akan mengurangi dampak dari beberapa pembobolan yang berhasil)
5. Memblokir (dalam hubungannya dengan deteksi penyerangan atau sistem pencegahan alamat IP atau subnets (sub jaringan) yang dilaporkan IDS atau IPS sedang menyerang jaringan organisasi
6. Memberitahukan administrator jaringan atau administrator Web server atau personil pengamanan yang tepat tentang kegiatan mencurigakan melalui alat/cara yang sesuai (misalnya *page,* e-mail, *network trap*)
7. Menyediakan penyaring konten
8. Melindungi dari serangan DoS
9. Mendeteksi permintaan dengan format yang salah atau serangan yang dikenal pada permintaan URL
10. Melakukan *log* kejadian-kejadian kritis, termasuk detil-detil berikut :
    a) Waktu/tanggal
    b) *Interface a*lamat IP

c) Nama kejadian spesifik manufaktur

d) Kejadian serangan standar (jika ada)

e) Sumber dan tujuan alamat IP

f) Sumber dan tujuan nomor *port*

g) Protokol jaringan.

7.2.2. Intrusion Detection System (IDS)

Intrusion Detection System (IDS) adalah suatu aplikasi yang memonitor peristiwa yang terjadi pada suatu sistem atau jaringan dan menganalisanya untuk mendapatkan tanda-tanda potensi insiden yang merupakan pelanggaran atau ancaman, mengenai pelanggaran dari kebijakan pengamanan komputer, kebijakan penggunaan yang dapat diterima atau praktek pengamanan standar. Suatu IPS mempunyai semua kemampuan dari suatu IDS dan juga dapat mencoba menghentikan potensi insiden. Karena sistem IDS dan IPS menawarkan banyak kemampuan yang sama, keduanya seringkali disebut sebagai sistem deteksi dan pencegahan intrusi (*Intrusion Detection and Prevention Systems [*IDPS]). Ketika suatu IDPS mendeteksi adanya suatu potensi insiden, IDPS akan memberitahukan kepada para administrator melalui pesan konsol IDPS, e-mails, *pages* atau mekanisme lainnya.

Dua jenis IDPS yang paling relevan untuk pengamanan Web yaitu berbasis *host* maupun berbasis jaringan, yang mensyaratkan *update* database tandatangan serangan secara sering agar dapat mengenali serangan-serangan yang baru. Suatu IDPS yang tidak diperbaharui secara berkala akan gagal mengenali serangan termutakhir (dan seringkali terpopular). Kedua jenis IDPS tersebut mungkin saja terbatas dalam hal kemampuannya mendeteksi serangan *zero day* (serangan pada hari ke nol) karena tak mungkin tersedia suatu tandatangan yang sesuai. Suatu IDPS berbasis *host* mungkin mempunyai kesempatan yang lebih baik dalam hal deteksi suatu serangan *zero day* karena mampu mendeteksi tindakan yang

mungkin diambil oleh penyerang setelah adanya suatu ekploitasi yang berhasil merupakan hal yang lebih baik (misalnya *akun privileged* baru yang tidak sah, pemasangan perangkat lunak *malicious*).

Untuk melindungi suatu Web server menggunakan suatu IDPS, pastikan bahwa IDPS tersebut dikonfigurasikan untuk :

1. Memonitor lalu lintas jaringan dari dan ke Web server
2. Memonitor perubahan terhadap *file* penting pada Web server (kemampuan mengecek integritas *file*). *File-file* penting tertentu semacam *file* yang menyimpan *password* pengguna dan *file log* , akan berubah secarareguler dan karenanya harus tidak diproteksi dengan suatu pengecek integritas *file*. Kecenderungan ini akan bervariasi bergantung pada Web server da OS yang digunakan
3. Memonitor sumber daya sistem yang ada pada *host* Web server (berbasis *host*)
4. Memblokir (dikombinasikan dengan firewall) alamat IP atau subnet (sub jaringan) yang menyerang jaringan organisasi
5. Memberitahu para pihak yang tepat (misalnya administrator IDPS, administrator Web server, tim tanggap insiden) tentang serangan yang diduga melalui alat yang tepat berdasarkan kebijakan dan prosedur tanggap insiden dari organisasi
6. Mendeteksi seluas mungkin beragam pemeriksaan dan serangan dengan suatu tingkat kesalahan positif yang dapat diterima
7. Membuat *log-log* kejadian, termasuk rincian sebagai berikut :
    a) Waktu/tanggal
    b) Sensor alamat IP
    c) Nama serangan khusus manufaktur

d) Nama serangan standar (jika ada)

e) Sumber dan tujuan alamat IP

f) Sumber dan tujuan nomor *port*

g) Protokol jaringan

8. Untuk kejadian jaringan, menangkap informasi *header* paket untuk membantu dalam hal proses analisa dan forensik

9. Sering meng-update tandatangan penyerang yang baru (misalnya berdasarkan harian hingga mingguan, umumnya setelah mengetes update tersebut).

Selain itu, sangatlah penting untuk memperkuat IDPS berbasis jaringan dan OSnya. Karena IDPS berbasis jaringan sering menjadi target penyerang. Secara khusus, IDPS berbasis jaringan sebaiknya tidak menanggapi apapun jenis interogasi sistem melalui monitor *interfacenya*. Jika manajemen jarak jauh ingin dilakukan, kondisi tersebut semestinya dilakukan melalui *out-of-band* (jaringan yang diisolasi terpisah). Meskipun terkadang sulit untuk mengadministrasikan dan menterjemahkan, IDPS adalah suatusistem peringatan awal yang penting dan dapat menyediakan informasi yang diperlukan administrator Web server untuk mempertahankan Web server dari serangan.

7.2.3 HoneyWeb

HoneyWeb adalah merupakan program aplikasi yang ditujukan untuk membuat jebakan bagi para penyusup/penyerang untuk melindungi web server. Honeyweb sengaja dipersiapkan untuk diserang, namun data yang berhasil ditembus adalah merupakan data atau informasi yang telah dipalsukan sebelumnya(bukan data asli dari web server). Namun demikian para penyerang tidak akan curiga bahwa data tersebut palsu karena data dibuat sedemikian rupa sehinga tidak menimbulkan kecurigaan.

Suatu Web Server palsu didesain dan diberikan suatu link (link admin) yang akan memancing penyerang, dan membuat penyerang akan seolah-oleh sukses menyerang dengan memasuki halaman secara terotorisasi (verifikasi userid dan password) dan tertantang untuk mengeksplorasi lebih jauh dan menghabiskan waktunya cukup lama sehingga secara otomatis web server yang asli akan aman dari mereka.

HoneyWeb dapat melengkapi keamanan web server disamping firewall yang dapat memblok penyerang/penyusup dan IDS yang dapat mendeteksi penyusup namun telah dianggap sebagai musuh oleh penyerang, Sedangkan Honeyweb adalah pengelabuan terhadap penyerang.

7.2.4 *Switch* Jaringan

Switch jaringan adalah peralatan yang menyediakan keterhubungan antara dua atau lebih *host* yang terletak pada segmen jaringan yang sama. Serupa dengan *hub* dalam hal memperbolehkan komunikasi diantara dua *host*, tetapi *switch* tersebut lebih "pintar" dan mengirimkan komunikasi hanya kepada *host-host* tujuan komunikasi tersebut. Keuntungan dari segi keamanan adalah ketika switch diterapkan pada suatu jaringan, maka akan jauh lebih sulit untuk menyadap komunikasi diantara *host* lainnya yang berada pada segmen jaringan. Hal ini sangatlah penting ketika suatu Web server berada pada segmen jaringan yang digunakan oleh *host* lainnya. Sebagai contoh, jika suatu *hub* digunakan dan suatu*host* pada DMZ yang telah bobol, maka seorang penyerang mungkin akan dapat menyadap komunikasi dari para *host* lainnya pada DMZ yang dapat memperluas menjadi bobolnya *host-host* tersebut, atau informasi yang dikomunikasikan melalui jaringan.

Banyak *switch* jaringan yang menyertakan tatanan keamanan khusus yang dapat menambah tingkat keamanan lebih lanjut dari suatu jaringan dengan mempersulit entitas jahat untuk "menaklukkan"

switch. Beberapa contoh termasuk kemampuan untuk memperkecil resiko dari *Address Resolution Protocol* (ARP) *spoofing* dan ARP Poisoning Attack. ARP poisoning terjadi ketika seorang penyerang berhasil mengupdate ARP *cache* pada suatu *host* sasaran dengan menggunakan entry ARP yang dipalsukan. Umumnya hal ini digunakan untuk mengarahkan kembali lalulintas jaringan guna tujuan malicious. Jika suatu*switch* memiliki kemampuan pengamanan ini, maka *switch* tersebut harus diaktifkan dengan melihat dokumen manufakturnya.

Switch dapat memiliki dampak yang negative pada IDPS berbasis jaringan. Sebagian besar *switch* jaringan memperbolehkan para administrator jaringan untuk mengkonfigurasikan suatu*port* khusus pada *switch*, yang dikenal sebagai suatu*span port*, sehingga *switch* dapat mereplikasikan semua lalu lintas *switch* ke port yang digunakan oleh IDPS. Kondisi ini memperbolehkan suatu IDPS berbasis jaringan untuk dapat melihat seluruh lalu lintas pada suatu segmen jaringan tertentu. Dengan beban tinggi *switch* tersebut mungkin harus menghentikan pengiriman lalu lintas ke *span port*, yang menyebabkan IDPS tidak dapat memonitor kegiatan jaringan. Perangkat-perangkat lainnya juga memerlukan span port dan hanya tersedia sedikit span port pada suatu switch, sehingga dapat terjadi tidak terhubungnya suatu IDPS dengan suatu *switch* karena semua *span port*nya sedang digunakan.

7.2.4 *Load Balancer*

Load balancer (penyeimbang beban) menyalurkan HTTP yang diminta melalui berbagai Web server, sehingga suatu organisasi untuk menambah kapasitas dari situs Web-nya dengan menambahkan server-server tambahan. *Load balancer* bertindak sebagai *virtual server* yang menerima semua permintaan HTTP pada situs Web. Berdasarkan kebijakan *load balancer*, permintaan-permintaan ini diteruskan ke salah satu *server* yang menjadi *host* situs Web. Kebijakan *load balancer* berupaya untuk memastikan

bahwa setiap *server* menerima sejumlah permintaan yang sama. Banyak *load balancer* mampu memantau *server* dan menggantikannya jika salah satu dari *server* tersebut tidak tersedia.

Load balancer seringkali ditambah dengan mekanisme *caching (*menyimpan cadangan). Banyak dari permintaan HTTP pada suatu Web server yang diterima adalah identik dan mengembalikan jawaban HTTP yang identik pula. Dan ketika pembangkit konten dinamis sedang dipergunakan, jawaban yang identik ini perlu dibangkitkan kembali setiap kali permintaan dibuat. Untuk mengurangi hal ini dan lebih jauh lagi sebagai beban pada Web server individu, organisasi dapat menempatkan *caching* server.

Seperti halnya *switch* jaringan, *load balancer* bukan dikhususkan menjadi alat keamanan, tetapi merupakan *tools* utama untuk menjaga ketersediaan dari suatu situs Web. Dengan memastikan bahwa beberapa Web server individu berbagi bebannya, dan bukan menempatkannya pada suatu Web server tunggal, maka akan mampu bertahan dengan lebih baik terhadap permintaan bervolume besar yang digunakan pada banyak serangan DoS. *Firewall, switch* dan *router* harus juga dikonfigurasikan (apabila memungkinkan) untuk membatasi jumlah lalu lintas yang melalui Web server, yang mengurangi lebih jauh resiko atas keberhasilan serangan DoS.

7.2.5 *Reverse Proxy*

Reverse proxy adalah perangkat yang ditempatkan antara suatu web server dan klien web server tersebut. Istilah "*reverse proxy"* digunakan karena aliran datanya merupakan kebalikan dari suatu *proxy* tradisional yang progresif. *Reverse proxy* dapat berlaku sebagai suatu tambahan yang bernilai terhadap keamanan dari suatu Web server. Istilah *reverse proxy* agak jarang digunakan pada industri. Dapat mencakup beberapa atau semua fungsi berikut ini :

1. Encryption Accelerator, yang digunakan untuk menurunkan beban proses perhitungan yang dibutuhkan untuk memulai koneksi SSL/TLS.
2. *Security Gateway,* yang memonitor lalu lintas HTTP dari dan ke Web server terhadap potensi serangan dan mengambil langkah yang diperlukan, bertindak sebagai suatu *firewall* tingkat aplikasi.
3. Content Filter, yang dapat memonitor lalu lintas dari dan ke Web server terhadap data sensitif yang berpotensi atau tidak sesuai; dan mengambil langkah yang diperlukan.
4. *Authentication Gateway,* yang mengotentikasikan para penggunanya melalui suatu mekanisme yang bervariasi dan mengendalikan akses ke URL yang *host* –nya pada Web server itu sendiri.

Reverse proxy harus dipertimbangkan pada saat mengembangkan Web server beresiko tinggi. Sementara memang ada penambahan resiko dengan kebutuhan penempatan perangkat keras dan perangkat lunak, resiko tersebut biasanya sebanding dengan manfaatnya. Sebagai tambahan pada fungsi-fungsi diatas, *proxy* Web juga berharga karena*dapat memberikan*suatu*layer* tambahan diantara Web server dan para penggunanya yang kurang dapat dipercaya. Berkenaan dengan sifat dasarnya yang sangat khusus, *proxy* dapat lebih mudah diamankan dari pada Web server. *Proxy* juga dapat menyembunyikan konfigurasi, tipe, lokasi dan detil lainnya dari suatu Web server yang berhubungan dengan para penyerang. Sebagai contoh, Web server memiliki *banner* (spanduk) yang secara berkala menunjukkan tipe dan versi Web server, dan *banner* ini kadang-kadang tidak dapat dirubah. Dengan suatu*reverse proxy,* hal ini bukan merupakan suatu pokok persoalan karena *proxy* dapat menuliskan kembali *banner* sebelum dikirimkan ke para pengguna.

8. Pengelolaan Web server

Setelah inisiasi proses deployment suatu Web server, para administrator perlu melakukan pemeliharaan keamanannya secara koninu. Bab ini memberikan rekomendasi umum untuk pengelolaan Web server secara aman. Aktivitas utamanya mencakup menangani dan menganalisid *file log*, melakukan*back-up Web server*, pemulihan dari bobolnya server, pengujian keamanan *Web server* secara reguler, dan melakukan remote administrasi secara aman.

8.1. *Logging*

Logging pada web server merupakan suatu proses pencatatan aktivitas yang telah terjadi, dalam rangka keamanan maka data pada *log* dimonitor dan dipelihara. *Log* pada jaringan dan sistem adalah hal yang penting, khususnya dalam hal dimana komunikasi data diproteksi menggunakan HTTPS, biasanya monitoring jaringan kurang efektif. Perangkat lunak Web server dapat menyediakan *log* data tambahan yang berkaitan dengan kejadian tertentu.

8.1.1. Mengidentifikasi Kemampuan *Logging*dari suatu Web server

Setiap jenis perangkat lunak Web server mempunyai kemampuan *logging* yang berbeda. Berikut adalah jenis-jenis *log* yang bergantung pada perangkat lunak Web server yang digunakan :

1. ***Transfer Log*** — Setiap transfer direpresentasikan sebagai satu *entry* yang menunjukkan informasi utama yang berkaitan dengan transfer.
2. ***Error Log*** — Setiap kesalahan direpresentasikan sebagai satu entry, termasuk penjelasan tentang alasan untuk laporan kesalahan.
3. ***Agent Log*** — *Log* ini mengandung informasi tentang perangkat

lunak klien yang digunakan untuk mengakses konten Web.

4. ***Referrer Log*** — *Log* ini mengumpulkan informasi yang berkaitan dengan akses HTTP yang mencakup URL dari halaman yang berisi link yang diikuti oleh perangkat lunak klien yang digunakan untuk akses ke halaman Web tersebut.

8.1.2. Identifikasi kebutuhan *Log*ging Tambahan

Jika suatu Web server mendukung eksekusi program, *script*, atau *plug-ins*, maka mungkin diperlukan *logging* tambahan berbasiskan mekanisme *logging* yang disediakan oleh sistem operasi *host* Web server tersebut. Seringkali, kejadian yang kritis terjadi dalam lingkup kode aplikasi Web dan tidak di *log* oleh Web server. Informasi *log* yang terkait dengan program-program, *script* dan *plug-ins* dapat ditambahkan pada *log* sehingga dapat menjadi informasi yang signifikan untuk investigasi terhadap suatu kasus/kejadian.

8.1.3. Rekomendasi Konfigurasi Umum pada *Log*ging

Konfigurasi berikut adalah konfigurasi umum yang direkomendasikan untuk *logging*pada Web server public:

1. Gunakan format *log* yang dikombinasikan untuk menyimpan Transfer *Log*, atau informasi dikonfigurasi secara manual, yang dikombinasikan menjadi format standar bagi *Log* Transfer.
2. Aktifkan *Referrer Log* atau *Agent Log* jika format *log* yang dikombinasikan tidak ada.
3. Buatlah nama *file log* yang berbeda untuk masing-masing situs Web virtual yang berbeda pada sebuah Web server .
4. Gunakan identitas pengguna *remote* sebagaimana yang dispesifikasikan dalam RFC 1413.
5. Pastikan adanya prosedur atau mekanisme agar *file log* tidak memenuhi *hard disk*.

Pengelolaan *file log* seperti memonitor ukuran *file log*, menghapus dan mengarsipkan *log* secara periodik perlu dilakukan.

Beberapa program Web server mempunyai kemampuan untuk memberlakukan atau menonaktifkan pengecekan *access kontrol*tertentu selama *start-up* program. Tingkat kendali ini mungkin diperlukan, sebagai contoh untuk menghindarkan pengubahan pada *file log* disebabkan kesalahan dalam pengadministrasian akses *file*.

8.1.4. Review dan Pemeliharaan *File Log*

Review *file log* adalah sesuatu tinjauan kembali pada *file log* yang membutuhkan ketelitian dan waktu, kegiatan ini akan memberikan informasi pada administrator mengenai aktifitas yang telah dilakukan atau telah terjadi. Frekuensi review tergantung pada faktor-faktor berikut ini :

1. Kuantitas lalu lintas yang diterima server
2. Tingkat ancaman secara umum yang dapat terjadi pada situs-situs penting.
3. Ancaman tertentu yang terjadi berkali-kali yang membutuhkan frekuensi analisis terhadap *file log*.
4. Kerawanan dari Web server
5. Nilai data dan layanan yang disediakan oleh Web server.

Berikut adalah rekomendasi dalam melakukan review dan pemeliharaan *file log* :

1. Review terhadap *log-log* harus dilakukan secara teratur, misalnya per hari, ketika suatu kegiatan yang mencurigakan teridentifikasi atau adanya notifikasi suatu peringatan ancaman. Tugas ini dapat menjadi beban berat bagi seorang

administrator Web server. Untuk mengurangi beban ini, gunakan tools untukmenganalisa *log* secara otomatis.

2. Diperlukan suatu analisa dengan jangka waktu tertentu dan lebih cermat dari suatu*log*. Dikarenakan suatu serangan Web server umumnya dapat dilakukan oleh ratusan *request*yang unik, seorang penyerang dapat mencoba menyamarkan suatu serangan Web server dengan cara meningkatkaninterval dalam *request-request* tersebut. Pada kasus ini, review terhadap *log-log* yang dilakukan per hari atau per minggu tidak akan menampilkan suatu kecenderungan yang dapat dikenali. Sebaiknya analisa dilakukan terhadap hasil review secara periodik sehingga serangan berkali-kali dari *host* yang sama atau subnet akan lebih mudah dikenali.

3. *File log* harus dilindungi untuk memastikan bahwa jika penyerang membobol suatu Web server maka *file log* tidak dapat diubah untuk menutupi serangan tersebut. Meskipun enkripsi dapat sangat bermanfaat dalam melindungi *file-file log*, solusi terbaik adalah menyimpan *file log* pada suatu*host* yang terpisah dari Web server, yang disebut server *logging* terpusat. *Logging* terpusat seringkali dilaksanakan menggunakan *syslog*, yaitu suatu protokol *logging* standar yang diadopsi oleh banyak system operasi. Administrator dapat melakukan konfigurasi tujuan dari pesan-pesan yang dikirim oleh proses-proses yang sedang berlangsung. Pesan-pesan yang diterima oleh sys*log* dapat disimpan dalam *file-file* yang berbeda dan ditampilkan pada konsol. Disamping itu juga sys*log* dapat dikonfigurasi untuk meneruskan pesan melintasi suatu jaringan menuju proses sys*log* pada komputer lain.

4. Memanfaatkan tekno*log*i *security information and event management*/SIEM baik yang diinstal pada *host* maupun pada server terpusat. SIEM adalah kombinasi dari kategori produk yang terpisah yaitu SIM *(security information management*) dan SEM *(security event management*). Tekno*log*i SIEM dapat melakukan analisis real-time dari alert keamanan yang dibuat

oleh hardware dan aplikasi jaringan.

5. *File-file Log* harus di*back-up* dan diarsipkan secara periodikdilakukan pada kurun waktu tertentu sesuai ketentuan yang berlaku. Masa Retensi dan perpanjangan waktu dalam pengarsipan *file-file log* bergantung pada nilai dari data dan layanan webserver, tingkat kerawanan dan ketentuan yang berlaku.

8.1.5 *Tools* untuk Menganalisa *File Log* secara Otomatis

Banyaknya lalu lintas data yang diterima Web server pada umumnya cukup signifikansehingga ukuran *file-file log* dapat menjadi besar. Untuk meringankan pekerjaan pada administrator Web server dapat dilakukan instalasi suatu *tools* untuk menganalisa *log* secara otomatis. *Tools* ini dapat menganalisa *entry* pada *file log* Web server dan mengidentifikasi kegiatan yang tak biasa dan mencurigakan. Dapat pula digunakan untuk melakukan analisa *file log* secara otomatis. Contoh produk komersial tools SIEM adalah AccelOps, AraKnos, ArcSight, BLUESOC, Cisco Security MARS, ImmuneSecurity, *LogLog*ic, *Log*ICA, SenSage, dan lain-lain.

Kebanyakan *tool* beroperasi pada format *log* umum maupun kombinasi dari *Transfer Log*. *Tools* ini dapat mengidentifikasi alamat IP koneksi dan transfer. *Tools Error Log* dapat menunjukkan tidak hanya kesalahan pada konten Web (seperti *file* hilang) tapi juga kesalahan mengakses URL yang tidak ada. Dari Tools ini dapat membantu untuk penyelidikan terhadap adanya kerawanan yang dapat digunakan jika terjadi serangan, pengumpulan informasi dan kejadian pada konten yang spesifik seperti misalnya database.

Penganalisis *log* harus meneruskan kejadian yang mencurigakan apapun kepada administrator Web server yang bertanggung jawab atau tim *incident response* keamanan sesegera mungkin untuk dilakukan penyelidikan lebih lanjut. Dalam operasionalnya dapat

dilakukan oleh dua atau lebih penganalisa *log*.

8.2. Prosedur Back-Up Web Server

Salah satu fungsi terpenting dari seorang administrator Web server adalah menjaga integritas data pada Web server. Hal ini penting karena Web server seringkali merupakan server yang paling penting dan paling tidak terlindungi pada jaringan komputersuatu organisasi. Ada dua komponen penting untuk melakukan *back up* data pada suatu Web server: *back-up* data dari aplikasi dan sistem operasi pada web server secara teratur, dan pemeliharaan dari data *back up* yang terlindungi dan secara terpisah dari konten Web.

8.2.1 Kebijakan dan Strategi *Back-Up* Web Server

Semua organisasi perlu menciptakan suatu kebijakan pelaksanaan *back-up* data Web server. Meskipun setiap kebijakan back-up Web server akan berbeda pada masing-masing organisasi, namun komponen dalam kebijakan tersebut harus mencakup hal-hal sebagai berikut :

1. Tujuan dari kebijakan
2. Para pihak yang terkait oleh kebijakan
3. Web server yang teratasi oleh kebijakan
4. Penjelasan dasar dari beberapa hal penting, khususnya hukum dan teknis
5. Persyaratan rinci dari sudut pandang hukum, bisnis dan organisasi
6. Frekuensi *back-up* yang dibutuhkan
7. Prosedur untuk memastikan data dipertahankan dan dilindungi secara tepat
8. Prosedur untuk memastikan data dihancurkan atau diarsipkan secara tepat ketika tidak diperlukan lagi

9. Prosedur untuk menyediakan informasi atas dasar permintaan, penyelidikan hukum dan permintaan serupa lainnya
10. Tanggung jawab dari pihak-pihak yang terlibat dalam hal retensi, proteksi dan kegiatan pemusnahan data
11. Waktu retensi untuk setiap tipe informasi yang di-*log* dan perlu dipertimbangkan secara hati-hati, sesuai dengan klasifikasi informasi yang berlaku.

Ada tiga tipe utama *back-up*:

1. *FullBack-up,* termasuk untuk aplikasi OS, dan data yang tersimpan pada Web server. Keuntungan dari *fullback-up* adalah mudah untuk melakukan *recovery* atau mengembalikan seluruh Web server ke keadaan semula yang sama (misalnya konfigurasi, level *patch,* data) dengan ketika dilakukan *back-up*. Kekurangan dari *full back-up* adalah sangat memakan waktu dan sumber daya. *Fullback-up* dilakukan dengan frekuensi yang lebih sedikit dibandingkan dengan yang diminta *incremental back-up,* biasanya mingguan hingga bulanan atau pada saat munculnya perubahan yang signifikan.
2. *IncrementalBack-up,* dapat mengurangi dampak dari *back-up* dengan cara hanya melakukan *back-up* data yang telah berubah sejak *back-up* sebelumnya (baik *full* maupun *incremental*).
3. *DiferentialBack-up,*mengurangi jumlah *back-up* yang harus diakses untuk mengembalikan suatu konfigurasi dengan cara mem*back-up* semua perubahan pada data sejak terakhir dilakukannya *full back–up.* Setiap *differential back-up* membutuhkan waktu proses dan tempat penyimpanan yang tidak sedikit. Secara umum *back up incremental* ataupun *diferential* lebih sering dilakukan, biasanya per hari hingga minggu.

8.2.2. Simulasi Web Server

Dalam menguji coba berbagai konten dan layanan yang akan diterapkan maupun teknik pengujian yang akan dilakukan, sebaiknya dilakukan pada Webser server simulasi. Idealnya, *server* simulasi ini memiliki perangkat keras dan perangkat lunak yang serupa dengan *Web server* yang beroperasi secara langsung serta diletakkan pada jaringan internal (intranet) yang dapat sepenuhnya dilindungi oleh perimeter keamanan jaringan. Biaya pemeliharaan suatu W*eb server* tambahan adalah merupakan suatu konsekuensi. Web server simulasi ini dapat digunakan sebagai :

1. *Platform* untuk menguji *patch* dan kumpulan layanan baru sebelum diaplikasikan pada W*eb server* produksi
2. *Platform* pengembangan untuk *Webmaster* dan administrator W*eb server* untuk mengembangkan dan menguji konten baru dan beserta aplikasinya
3. *Platform* untuk menguji pengaturan konfigurasi sebelum diaplikasikan ke W*eb server* produksi
4. *Platform* pengembangan dan pengujian perangkat lunak untuk mengetahui suatu resiko pengamanan yang tidak dapat diterima pada server produksimisalnya *software complier,* perangkat *tool* yang bersifat administrasi, *software* untuk akses remote.

Web server untuk keperluan simulasi pengujian ini harus dipisahkan dari server yang memelihara*back up*konten pada W*eb server* produksi.

8.3. Pemulihan Dari suatu Kebobolan Keamanan

Suatu serangan dalam usaha membobol keamanan pada *host* maupun jaringan internal tidak dapat dihindari dan harus dihadapi. Langkah pertama dalam tahap pemulihan dari suatu pembobolan adalah membuat dan mendokumentasikan kebijakan-kebijakan dan

prosedur-prosedur yang diperlukan untuk menanggapi keberhasilan penyerangan *sebelum* kekacauan terjadi. Prosedur respon berisi tindakan-tindakan yang diperlukan untuk merespon insiden pembobolan yang berhasil terhadap suatu Web server dan urutan yang tepat dari tindakan tersebut, urutan dapat menjadi sangat penting. Dengan adanya Petugas Keamanan Sistem Informasi dan Manajer Program Keamanan Sistem Informasi dan Manajer Program Keamanan Sistem Informasi, diharapkan dapat mengatasi dengan cepat ketika ada kecurigaan atau konfirmasi mengenai suatu pembobolan.

Seorang administrator Web server harus mengikuti kebijakan dan prosedur yang telah dibuat untuk penanganan insiden dan Petugas Keamanan Sistem Informasi adalah yang pertama dihubungi untuk mendapatkan petunjuk sebelum mengambil langkah apapun setelah muncul kecurigaan atau konfirmasi adanya pembobolan keamanan. Contoh lain dari langkah-langkah yang biasanya dilakukan setelah penemuan suatu pembobolan adalah sebagai berikut:

1. Melaporkan insiden tersebut kepada Petugas Keamanan Sistem Informasi .
2. Petugas Keamanan Sistem Informasi berkonsultasi secara cepat diinternal dengan Manajer Program Keamanan Sistem Informasi atau denganpihak-pihak terkait yang terdapat pada Lampiran C, dan jika diperlukan dapat ditindaklanjuti secara hukum dengan berkonsultasi dengan pihak manajemen konsultan hukum dan penegakanhukum.Dalam hal Manajer Program Keamanan Sistem Informasi belumterbentuk, maka Petugas Keamanan Sistem Informasi akan melaporkan insiden yang terjadi pada pimpinan untuk diputuskan apakah akan diatasi secara internal atau ditindaklanjuti secara hukum (dapat dilihat pada Lampiran C).
3. Petugas Keamanan Sistem Informasi mengisolasi sistem yang bobol atau mengambil langkah-langkah lain untuk mengisolasi

serangan sehingga informasi tambahan dapat dikumpulkan. Mengisolasikan sistem harus diselesaikan dengan sangat hati-hati, jangan sampai terjadi perubahan terhdap sistem karena akan dijadikan sebagai barang bukti (*incident evidence*). Beberapa penyusup melakukan konfigurasi sistem-sistem yang bobol untuk menghapus bukti jika suatu sistem yang bobol diputuskan hubungannya dari jaringan atau di *boot* kembali. Salah satu metode untuk mengisolasi suatu sistem dapat berupa mengkonfigurasikan kembali switch atau router terdekat.

4. Dalam hal keputusan pimpinan menyatakan bahwa pemulihan akan diatasi secara internal, maka dapat dilakukan oleh Petugas Keamanan Informasi yang berkompeten hal-hal sebagai berikut:

   a) Setelah isolasi, dilakukan backup untuk keperluan analisa audit internal.

   b) Menyelidiki *host-host* yang serupa untuk menentukan apakah penyerang juga membobol sistem-sistem lainnya. *Host* serupa akan mengikutsertakan *host-host* yang berada di kisaran alamat IP yang sama, memiliki *password* sama menjalin suatu koneksi yang terpercaya, dan/atau memiliki OS dan/atau aplikasi sama.

   c) Melakukan analisis terhadap intrusi tersebut, termasuk :

      (1) Keadaan server pada saat itu, dimulai dengan data yang berlangsung paling dekat, misalnya koneksi jaringan, *dump* memori, *file* time-stamp, dan para pengguna yang *log in* pada saat itu.

      (2) Modifikasi yang dibuat terhadap perangkat lunak dan konfigurasi sistem.

      (3) Modifikasi yang dibuat terhadap data.

      (4) *Tools* atau data yang ditinggalkan oleh penyerang.

(5) *File*-fie *log* sistem, deteksi intrusi, dan *firewall.*

d) Mengembalikan sistem

(1) Menginstal dengan baik suatu versi OS, aplikasi-aplikasi, *patch-patch* yang dibutuhkan, dan konten Web; maupun mengembalikan sistem dengan membuat salinan dari *backup*, pilihan ini dapat menjadi lebih beresiko karena *backup* bisa saja dibuat setelah adanya pembobolan, dan mengembalikan sistem dengan membuat salinan dari suatu*backup* yang bobol masih memungkinkan penyerang untuk mengakses ke sistem).

(2) Menonaktifkan layanan yang tidak dibutuhkan.

(3) Menerapkan seluruh *patch.*

(4) Mengubah seluruh *password,* bahkan jika perlu pada *host-host* yang tidak bobol.

(5) Mengkonfigurasi ulang elemen keamanan jaringan untuk membuat tambahan perlindungan dan notifikasi, seperti: *firewall, router,* dan IDPS.

e) Menguji sistem untuk memastikan keamanan.

f) Menghubungkan kembali sistem ke jaringan.

g) Memonitor sistem dan jaringan dalam hal tanda-tanda bahwa penyerang sedang berusaha untuk mengakses lagi sistem atau jaringan.

h) Mendokumentasikan pembelajaran yang diperoleh.

Berdasarkan kebijakan dan prosedur yang telah dibuat, para administrator sistem sebaiknya memutuskan apakah menginstalasi kembali OS dari suatu sistem yang bobol atau menyalinnya dari suatu*back-up.* Faktor-faktor yang sering kali dipertimbangkan termasuk hal-hal berikut:

1. Level akses yang diperoleh penyerang (contoh: *root,* pengguna, *guest,* sistem).
2. Tipe penyerang (internal atau eksternal).
3. Tujuan pembobolan (misalnya, *defacement* halaman Web, tempat penyimpanan perangkat lunak, *platform* untuk serangang-serangan lainnya).
4. Metoda yang digunakan untuk pembobolan sistem.
5. Tindakan dari penyerang selama dan setelah pembobolan (misalnya, *file log*, laporan deteksi penyerangan).
6. Durasi dari pembobolan.
7. Luasnya area pembobolan pada jaringan (misalnya, jumlah *host* yang dibobol).
8. Hasil dari konsultasi dengan manajemen dan konsultan hukum.

Semakin rendah level akses yang diperoleh oleh penyusup dan semakin dapat dideteksi tindakan penyusupan tersebut oleh administrator Web server, semakin berkurang resiko dalam hal memulihkan sistem dengan menyalin dari suatu*backup* dan melakukan *patch* kerawanan. Untuk insiden dimana tindakan-tidakan penyerang kurang dapat dideteksi dan/atau dimana penyerang mendapatkan akses level tinggi, direkomendasikan untuk menginstal ulang OS dan aplikasi-aplikasi dari media yang terpercaya, serta data Web server disalin dari suatu *backup* yang otoritatif.

8.4. Pengujian Keamanan Web Server

Pengujian keamanan secara periodik terhadap Web server sangat penting. Tanpa pengujian secara periodik, tidak ada jaminan terhadap tindakan protektif yang dilakukan atau *patch* pengamanan yang diterapkan oleh administrator Web server berfungsi sebagaimana yang mestinya. Walaupun ada beragam teknik

pengujian pengamanan, namun *vulnerability scanning* adalah merupakan teknik yang paling banyak digunakan. *Vulnerability scanning* memberikan bantuan pada seorang administrator Web server dalam mengidentifikasi kelemahan serta memverifikasi apakah tindakan pengamanan yang ada sudah efektif. *Penetration testing* juga digunakan, akan tetapi frekuensi tidak sebanyak *vulnerability scanning* dan biasanya hanya merupakan bagian dari keseluruhan pengujian penetrasi dari jaringan organisasi.

8.4.1 *Scanning* Kerawanan

Vulnerability scanning adalah upaya mengidentifikasi kerawanan dalam *host-host* yang dipindai. *Vulnerability scanner* bergantung pada *update* periodik dari *database* kerawanan untuk dapat mengenali kerawanan terkini. Sebelum menjalankan alat pemindai apapun, para administrator W*eb server* harus menginstal *update* terbaru ke *database* kerawanan mereka.

Vulnerability scanner mempunyai kemampuan sebagai berikut :

1. Mengidentifikasi *host* yang sedang aktif pada jaringan
2. Mengidentifikasi layanan aktif (*port*) pada komputer dan yang rentan
3. Mengidentifikasi program aplikasi
4. Mengidentifikasi sistem operasi
5. Mengidentifikasi kerawanan yang terkait dengan sistem operasi dan aplikasi yang ditemukan
6. Melakukan uji kepatuhan sesuai dengan petunjuk penggunaan aplikasi kebijakan pengamanan.

Pemindaian terhadap kerawanan ini merupakan suatu kegiatan yang harus dilakukan secara intensif serta sangat memerlukan keterlibatan manusia dalam menginterpretasikan hasilnya. Kegiatan ini juga dapat mengganggu kegiatan operasional lainnya seperti

menghabiskan *bandwidth*, memperlambat waktu respon jaringan dan juga berpotensi mempengaruhi ketersediaan dari *server* yang dipindai atau aplikasinya yang sedang dipindai. Akan tetapi, pemindaian kerawanan ini sangat penting untuk memastikan bahwa kerawanan tersebut dimitigisikan secepat mungkin sebelum ditemukan dan dieksploitasi oleh pihak-pihak yang tidak diinginkan. Pemindaian kerawanan sebaiknya dilakukan pada basis mingguan hingga bulanan. Hasil pemindaian sebaiknya didokumentasikan dan kekurangan yang ditemukan sebaiknya dikoreksi.

Sebaiknya juga digunakan lebih dari satu jenis *vulnerability scanner*. Karena tidak ada *vulnerability scanner* yang dapat mendeteksi seluruh kerawanan yang ada; namun dengan menggunakan dua scanner umumnya dapat meningkatkan jumlah kerawanan yang terdeteksi. Dapat digunakan satu scanner komersil dan satu scanner *freeware (*gratis) baik untuk jaringan maupun *host*.

8.4.2 Penetration Testing

Penetration Testing adalah uji keamanan dimana para evaluator berupaya memeriksa fitur-fitur keamanan dari suatu sistem berdasarkan pemahamannya terhadap desain sistem dan implementasinya. Tujuan dari Penetration Testing adalah untuk menguji sistem proteksi, khususnya respon terhadap indikasi serangan, dengan menggunakan perangkat dan teknik yang biasa dikembangkan oleh para penyerang. Pengujian ini sangat direkomendasikan untuk sistem yang kompleks dan kritis (penting).

Penetration testing memerlukan keahlian tinggi untuk memilimalisir resiko bagi system yang menjadi sasaran, selain itu juga dapat memperlambat waktu respon jaringan yang disebabkan oleh pemetaan jaringan (network mapping), pemindaian kerawanan dan lebih jauh lagi jika dilakukan secara tidak hati-hati maka dapat mengakibatkan kerusakan system. Hal-hal yang perlu diperhatikan

dalam Penetration Testing adalah sebagai berikut :

1. Menguji jaringan menggunakan metodo*log*i dan *tools* yang sama dengan yang digunakan oleh para penyerang
2. Memverifikasi apakah kerawanan tersebut benar-benar ada.
3. Pengujian dilakukan secara lebih mendalam, tidak hanya pada tingkat permukaan, dan mendemonstrasikan bagaimana kerawanan ini dapat dieksploitasi berulang-ulang untuk memperoleh akses yang lebih besar
4. Dapat mendemonstrasikan bahwa kerawanan tidak hanya bersifat teoritis
5. Memberikan rekomendasi yang diperlukan untuk ditangani pokok-pokok persoalan keamanan
6. Jika memungkinkan adanya pengujian prosedur dan kerawanan faktor manusia terhadap rekayasa sosial.

8.5. *Remote Administration* Pada Web Server

Adminstrasi dan *update* konten suatu W*eb server* dari jarak jauh (*remote*) dapat dilakukan hanya setelah dilakukannya pertimbangan secara cermat terhadap resiko-resiko yang mungkin timbul. Konfigurasi yang paling aman adalah tidak diperbolehkannya administrasi dan *update* konten secara remote.Resiko-resiko yang mungkin ditimbulkan apabila dilakukannya administrasi dan *updating* secara remote bervariasi tergantung lokasi *Web server* pada jaringan. Untuk suatu W*eb server* yang lokasinya di belakang *firewall*, administrasi dan *update* secara remote dapat diimplementasikan secara relatif aman dari jaringan internal, namun bukan tanpa resiko. Administrasi dan *updating* secara remote tidak diperbolehkan untuk dilakukan dari suatu*host* yang lokasinya terletak di luar jaringan komputer organisasi kecuali dilakukan dari komputer yang dikontrol misalnya melalui suatu VPN.

Apabila suatu organisasi memutuskan untuk dapat melakukan administrasi dan *updating* konten pada suatu W*eb server* secara remote, harus dengan mengikuti langkah-langkah berikut dapat memastikan bahwa konten diimplementasikan dengan cara seaman mungkin:

1. Gunakan mekanisme otentikasi yang kuat (misalnya pasangan kunci publik*/private*, otentikasi dua faktor (*two factor authentication*) atau lebih
2. Batasi/tentukan komputer (*host*) mana yang digunakan untuk melakukan administrasi dan *updating* isi *web* secara remote.Pembatasan dapat dilakukan berdasarkan :
    a) Pengguna yang terotorisasi
    b) Alamat IP
    c) Penggunaan komputer pada jaringan internal melalui mekanisme akses jarak jauh (misal : VPN)
3. Penggunaan protokol keamanan yang menyediakan enkripsi pada masing-masing *password* dan datanya (misal : SSH, HTTPS). Jangan menggunakan protokol jaringan yang tidak dirancang untuk mengirimkan data secara aman (misal : telnet, FTP, NFS, HTTP) kecuali dalam kondisi sangat dibutuhkan dan melalui mekanisme *tunnelling* pada protokol terenkripsi, seperti SSH, SSL, atau IPSec.
4. Gunakan konsep kewenangan terbatas (*least privilege*) dalam melakukan administrasi dan *updating* isi *web* secara jarak jauh untuk meminimalisir hak akses administrasi dan *updating* isi *web*.
5. Tidak mengizinkan administrasi jarak jauh dari internet melalui *firewall* kecuali dilakukan melalui mekanisme yang aman, seperti VPN.
6. Ganti nama *akun* dan *password default* untuk setiap aplikasi atau program yang digunakan untuk administrasi secara jarak

jauh.

7. Tidak melakukan *file sharing* (berbagi *file*) pada jaringan internal dari *web server* atau sebaliknya.

9. Checklist Keamanan Web server

Untuk memudahkan proses penerapan keamanan Web server maka terdapat hal-hal yang perlu menjadi perhatian untuk diketahui dan/atau dilakukan. Checklist dari substansi yang telah dijelaskan pada bab-bab sebelumnya diberikan pada Bab berikut ini.

9.1. Checklist Merencanakan dan Mengelola Web Server

Dipenuhi	Tindakan
	Perencanaan konfigurasi dan deployment Web server
☐	Mengidentifikasi fungsi- fungsi dari Web server
☐	Mengidentifikasi kategori informasi yang akan disimpan, diproses, dan dikirim melalui Web server
☐	Mengidentifikasi kebutuhan keamanan informasi
☐	Mengidentifikasi bagaimana informasi dipublikasikan ke Web server
☐	Mengidentifikasi kebutuhan keamanan *host* lain yang terlibat (misalnya backend database atau Web server)
☐	Mengidentifikasi *host* yang ditunjuk untuk menjalankan Web server
☐	Mengidentifikasi layanan jaringan yang akan diberikan atau didukung oleh Web server
☐	Mengidentifikasi kebutuhan keamanan dari suatu layanan tambahan yang diberikan atau didukung

Dipenuhi	Tindakan
	oleh Web server
☐	Mengidentifikasi bagaimana Web server akan dikelola
☐	Mengidentifikasi para pengguna dan kategori para pengguna dari Web server dan tentukan *priviledge*dari setiap kategori pengguna
☐	Identifikasi metode otentikasi pengguna pada Web server dan bagaimana data otentikasi akan diproteksi
☐	Mengidentifikasi bagaimana akses ke sumber informasi akan diberlakukan
☐	Mengidentifikasi mekanisme keamanan fisik yang tepat
☐	Mengidentifikasi mekanisme ketersediaan yang tepat
	Pemilihan OS yang tepat untuk Web server
☐	Mengidentifikasi penyingkapan kerawanan yang paling minimal
☐	Kemampuan untuk membatasi aktifitas tingkat administratif atau root hanya untuk para pengguna yang sah
☐	Kemampuan untuk mengontrol akses terhadap data pada server
☐	Kemampuan untuk menonaktifkan layanan jaringan yang tidak diperlukan dalam perangkat lunak OS atau server
☐	Kemampuan untuk mengontrol akses ke berbagai

Dipenuhi	Tindakan
	bentuk program yang dapat dieksekusi, seperti CGI script dan server plug-in
☐	Kemampuan untuk merekam setiap aktifitas server untuk mendeteksi penyerangan dan usaha penyerangan
☐	Ketersediaan kapabilitas firewall *host-based*
☐	Ketersediaan staf yang berpengalaman untuk menginstal, mengkonfigurasi, mengamankan, dan memelihara OS
	Pemilihan platform yang tepat untuk Web server
☐	*General purpose OS* *Trusted OS* *Web server appliance* *Pre-hardened OS and Web server* *Virtualized Platform*

9.2. Checklist Mengamankan Sistem Operasi Web Server

Dipenuhi	Tindakan
	Patch dan upgrade OS
☐	Melakukan dan mendokumentasikan proses *patching*
☐	Menjaga server agar tidak terhubung ke jaringan atau terisolasi yang membatasi koneksi sampai seluruh *patch* telah dipasang
☐	Mengidentifikasi dan menginstalasi seluruh *patch* dan *upgrade* yang dibutuhkan ke OS
☐	Mengidentifikasi dan menginstalasi seluruh *patch* dan *upgrade* yang dibutuhkan pada aplikasi dan layanan yang terkait dengan OS
☐	Mengidentifikasi dan meminimalkan kerawanan yang tidak di-*patch*
	Hilangkan atau nonaktifkan layanan dan aplikasi yang tidak diperlukan
☐	Non-aktifkan atau hilangkan layanan dan aplikasi yang tidak diperlukan
☐	**Konfigurasikan otentikasi pengguna OS**
☐	Hilangkan atau nonaktifkan akun default dan grup yang tidak dibutuhkan
☐	Non-aktifkan akun yang tidak interaktif

Dipenuhi	Tindakan
☐	Menciptakan grup-grup pengguna untuk komputer-komputer tertentu
☐	Menciptakan akun-akun pengguna untuk computer-komputer tertentu
☐	Memeriksa kebijakan password organisasi dan atur akun password secara tepat (misal panjang, kompleksitas dan sebagainya)
☐	Mencegah penebakan password (misalnya memberikan jeda waktuantara percobaan *log*in password, tolak *log*in setelah sejumlah percobaan gagal yang telah ditentukan)
☐	Menginstalasi dan mengkonfigurasi mekanisme keamanan lain untuk memperkuat otentikasi
	Konfigurasi kontrol sumber daya secara tepat
☐	Menolak akses untuk membaca *file-file* dan direktori-direktori yang tidak diperlukan
☐	Menolak akses untuk menulis pada *file-file* dan direktori-direktori yang tidak diperlukan
☐	Membatasi priviledge dalam hal eksekusi dari tool sistem hanya pada administrator sistem
	Instalasi dan Konfigurasi kontrol keamanan tambahan
☐	Memilih, menginstalasi pasang dan mengkonfigurasikan perangkat lunak tambahan untuk menyediakan kontrol-kontrol yang dibutuhkan yang tidak termasuk dalam OS

Dipenuhi	Tindakan
	Uji keamanan dari OS
☐	Mengidentifikasikan sistem identik yang terpisah
☐	Uji OS dilakukan setelah instalasi awal untuk menentukan kerawanan
☐	Uji OS secara berkala (misal triwulan) untuk menentukan kerawanan baru

9.3. Checklist Mengamankan Web Server

Dipenuhi	Tindakan
	Instalasi Web server secara aman
☐	Menginstalasi perangkat lunak Web server pada suatu *host* yang telah ditetapkan atau guest OS yang divirtualkan dan ditetapkan
☐	Mengaplikasikan suatu *patches* atau *upgrade* untuk memperbaiki kerawanan yang diketahui
☐	Membuat suatu *phisical disc* atau *logical portion*(terpisah dari OS dan aplikasi Web server) untuk konten Web
☐	menghilangkan atau nonaktifkan seluruh layanan yang dipasang oleh aplikasi Web server namun tidak dibutuhkan (misal gopher, FTP, administrasi remote)
☐	menghilangkan atau nonaktifkan seluruh akun *log*in default yang tidak dibutuhkan yang dibuat pada saat instalasi Web server
☐	Menghilangkan seluruh dokumentasi manufaktur dari server
☐	Menghilangkan *file* contoh atau *file* uji apapun dari server, termasuk *script* dan kode yang dapat dijalankan
☐	Mengaplikasikan *template* keamanan yang sesuai atau *hardening* script ke server
☐	Mengkonfigurasikan kembali banner layanan HTTP (dan layanan lain yang dibutuhkan) untuk tidak melaporkan tipe dan versi dari Web server dan OS

Dipenuhi	Tindakan
	Konfigurasikan OS dan *Access control*Web server
☐	Mengkonfigurasikan proses Web server untuk dapat dijalankan oleh pengguna dengan priviledge yang dibatasi dengan ketat
☐	Mengkonfigurasikan Web server sedemikian hingga *file* konten Web dapat dibaca namun tidak dapat ditulis oleh proses layanan
☐	Mengkonfigurasikan Web server sedemikian hingga proses layanan tidak dapat menulis ke direktori dimana konten Web disimpan
☐	Mengkonfigurasikan Web server sedemikian hingga hanya proses yang berhak untuk administrasi Web server yang dapat menulis *file* konten Web
☐	Mengkonfigurasikan OS *host* sedemikian hingga Web server dapat menulis *file log* namun tidak dapat membacanya
☐	Mengkonfigurasikan *host* OS sehingga *file* temporer yang dibuat oleh aplikasi Web server terbatas untuk subdirektori yang khusus dan dilindungi dengan baik.
☐	mengkonfigurasikan *host* OS sehingga akses untuk *file* temporer manapun yang dibuat oleh aplikasi Web server terbatas untuk proses layanan yang telah membuat *file*.
☐	menginstalasikan konten Web pada suatu hard drive atau *log*ical partition yang berbeda dari OS dan aplikasi Web server
☐	Jika *upload* ke Web server diperbolehkan, konfigurasikan sedemikian hingga ada suatu batasan tentang jumlah ruang hard drive yang ditetapkan

Dipenuhi	Tindakan
	untuk tujuan ini; *upload* sebaiknya ditempatkan dalam partisi yang terpisah
☐	Pastikan bahwa *file log* tersimpan dalam suatu lokasi yang diukur secara tepat; *file log* sebaiknya ditempatkan pada suatu partisi terpisah
☐	Konfigurasi jumlah maksimum proses Web server dan/atau koneksi jaringan yang sebaiknya diperbolehkan oleh Web server
☐	Pastikan bahwa guest OS virtual apapun mengikuti checklist ini
☐	Pastikan bahwa para pengguna dan administrator dapat mengubah password
☐	Non-aktifkan para pengguna setelah tidak-aktif untuk periode tertentu
☐	Pastikan para pengguna dan administrator yang memiliki ID yang unik
	Konfigurasikan suatu direktori konten Web yang aman
☐	Tetapkan suatu hard drive atau *log*ical partition tunggal untuk konten Web dan bangun subdirektori terkait khusus untuk *file* konten Web server, termasuk grafik namun tidak termasuk script dan program lain
☐	Tentukan suatu direktori tunggal khusus untuk seluruh script atau program-program eksternal yang dijalankan sebagai bagian dari konten Web server (misal CGI, ASP)
☐	Non aktifkan eksekusi script yang tidak secara khusus dibawah kontrol akun administratif. Tindakan ini

Dipenuhi	Tindakan
	dicapai dengan pembuatan dan pengontrolan akses ke suatu direktori terpisah yang dimaksudkan untuk memuat script yang berhak
☐	Non aktifkan penggunaan hard atau symbolic link (misal, *shortcut* untuk Windows)
☐	Identifikasikan folder dan *file* mana dalam dokumen Web server yang harus dibatasi dan mana yang sebaiknya dapat diakses (dan oleh siapa)
☐	Cek kebijakan password dari organisasi dan atur akun password secara tepat (misal, panjang, kompleksitas)
☐	Gunakan *file* robots.txt, jika sesuai
☐	Konfigurasikan perlindungan anti spambot, jika ada (misal, *CAPTCHA, nofollow*, atau *keyword filtering*)

9.4. Checklist Mengamankan Konten Web

Dipenuhi	Tindakan
	Pastikan tidak ada tipe-tipe informasi berikut ini yang terdapat pada atau melalui suatu Web server
☐	Rekaman (*Record*) yang diklasifikasikan
☐	Aturan dan prosedur personil lingkup internal
☐	Informasi sensitif atau berklasifikasi
☐	Informasi pribadi tentang personil suatu organisasi
☐	Nomor telepon, alamat email, atau daftar umum staf kecuali jika diperlukan untuk memenuhi persyaratan yang bersifat organisasi
☐	Jadwal pimpinan organisasi atau lokasi tepat mereka (apakah ada atau tidak di lokasi kantor)
☐	Informasi komposisi, persiapan atau penggunaan dari materi berbahaya.
☐	Informasi sensitif yang berkaitan dengan keamanan negara
☐	Catatan investigasi
☐	Catatan keuangan (diluar yang telah tersedia secara publik)
☐	Catatan medis
☐	Prosedur keamanan fisik dan informasi organisasi
☐	Informasi tentang jaringan organisasi dan infrastruktur sistem informasi

Dipenuhi	Tindakan
☐	Informasi yang mengkhususkan atau mengimplikasikan kerawanan keamanan fisik
☐	Rencana, peta, diagram, foto udara, dan rencana arsitektural organisasi gedung, properti, atau instalasi gedung
☐	Materi hak cipta tanpa ijin tertulis dari pemilik
☐	Kebijakan privasi atau keamanan yang menunjukkan tipe adanya tindakan keamanan hingga tingkat yang mungkin berguna bagi penyerang
	Tetapkan suatu kebijakan formal dan proses terdokumentasi dalam lingkup organisasi (*organizational-wide documented)* mengenai konten Web yang ditampilkan
☐	Informasi teridentifikasi yang sebaiknya dipublikasikan pada Web
☐	Sasaran audiens teridentifikasi
☐	Identifikasi efek negatif yang mungkin muncul akibat publikasi informasi
☐	Penanggung jawab yang jelas untuk pembuatan, publikasi, dan pemeliharaan informasi khusus.
☐	Menyediakan pedoman dalam hal gaya dan bentuk yang sesuai untuk publikasi Web
☐	Menyediakan tinjauan ulang yang sesuai terhadap informasi dalam hal sensifitas dan distribusi/kontrol peluncuran (termasuk sensifitas informasi dalam suatu kumpulan)
☐	Menentukan kontrol akses dan keamanan yang sesuai

Dipenuhi	Tindakan
☐	Menyediakan pedoman tentang informasi yang dimuat dalam source code dari konten Web
	Kelola privasi pengguna Web
☐	Kelola suatu kebijakan privasi yang dipublikasikan
☐	Lakukan larangan pengumpulan data identifikasi secara pribadi tanpa ijin eksplisit dari pengguna dan hanya kumpulkan data yang diperlukan
☐	Lakukan larangan penggunaan cookies yang "menetap "
☐	Gunakan *session cookie* hanya jika diidentifikasikan secara jelas dalam kebijakan privasi yang dipublikasikan
	Kurangi serangan tidak langsung pada konten
☐	Pastikan para pengguna situs waspada terhadap bahaya serangan phishing dan pharming dan bagaimana menghindarinya
☐	Validasi komunikasi resmi dengan membuat emails yang khas (*personilized email)* dan menyediakan informasi identifikasi yang unik (tetapi tidak rahasia) yang sebaiknya hanya organisasi dan pengguna yang tahu
☐	Gunakanan *signature*pada email jika sesuai
☐	Jalankan validasi konten dalam aplikasi Web untuk menghindari serangan *phishing* yang lebih rumit (misal serangan berbasis *scripting* antar-situs)
☐	Buatlah konten Web (*personalize*) untuk mengidentifikasi situs Web yang sah

Dipenuhi	Tindakan
☐	Gunakan otentikasi berbasis token atau otentikasi mutual jika dapat diaplikasikan
☐	Sarankan penggunaan Web browser atau browser toolbars dengan perlindungan terhadap phishing/pharming
☐	Gunakan versi terkini dari software DNS dengan *patch* keamanan versi terakhir
☐	Instalasi mekanisme perlindungan DNS server-side
☐	Monitor domain organisasi dan domain yang serupa
☐	Sederhanakan struktur nama domain organisasi
☐	Gunakan koneksi yang aman untuk *log*in
☐	Jika perlu, ikutsertakan suatu vendor untuk menyediakan tindakan anti-phishing/anti-pharming yang lebih kuat
	Pertimbangan keamanan active content *client-side*
☐	Pertimbangkan resiko dan keuntungan active content*client-side*
☐	Jangan ambil tindakan tanpa pernyataan ijin dari pengguna
☐	Jika mungkin, gunakan hanya active content yang di-adopsi secara luas seperti JavaScript, PDF, dan Flash
☐	Jika mungkin, sediakan beberapa alternatif (misal, HTML yang disediakan bersama PDF)
	Kelola keamanan active content server-side
☐	Hanya kode yang sederhana, mudah untuk

Dipenuhi	Tindakan
	dimengerti yang sebaiknya digunakan
☐	Batasi atau tidak ada pembacaan atau penulisan kepada *file* sistem yang sebaiknya diijinkan
☐	Batasi atau tidak ada interaksi dengan program-program lain (misal, sendmail) yang semestinya diijinkan
☐	Sebaiknya tidak ada persyaratan untuk menjalankan dengan suid privileges pada UNIX atau Linux
☐	Nama path yang jelas sebaiknya digunakan (yaitu tidak bergantung pada variabel path)
☐	Tidak ada direktori yang memiliki ijin untuk menulis dan eksekusi
☐	Seluruh *file* yang dapat dijalankan ditempatkan dalam suatu folder yang telah ditetapkan
☐	SSI di non-aktifkan atau fungsi eksekusi di nonaktifkan
☐	Seluruh input pengguna di validasi
☐	Kode pembangkit konten Web harus di scan atau di audit
☐	Halaman yang dibuat secara dinamis tidak menghasilkan *metacharacters* yang berbahaya
☐	Pengkodean himpunan karakter harus diatur dengan jelas dalam setiap halaman
☐	Data pengguna harus di scan untuk menjamin hanya mengandung input yang diharapkan, (misal a-z, A_Z, 0-9); *care should be taken* dengan karakter khusus atau HTML tags

Dipenuhi	Tindakan
☐	Cookies harus diperiksa dalam hal karakter khusus apapun
☐	Mekanisme enkripsi harus digunakan untuk mengenkripsi password yang dimasukkan melalui bentuk script
☐	Untuk aplikasi Web yang dibatasi oleh nama pengguna dan password, tidak ada halaman Web dalam aplikasi yang semestinya dapat di akses tanpa mengeksekusi proses *log*in yang sesuai
☐	Seluruh script contoh dihilangkan
☐	Tidak ada script pihak ketiga atau kode yang dapat di eksekusi yang digunakan tanpa memverifikasi *source code*

9.5. Checklist Penggunaan Tekno*log*i Otentikasi dan Enkripsi untuk Web Server

Dipenuhi	Tindakan
	Lindungi terhadap serangan *brute force*
☐	Gunakan otentikasi yang kuat jika memungkinkan
☐	Gunakan delay setelah usaha *log*in yang gagal
☐	Lock-out suatu akun setelah satu set usaha *log*in yang gagal
☐	Berlakukan suatu kebijakan password
☐	Blacklist IP address atau domain yang diketahui untuk usaha serangan *brute force*
☐	Gunakan software pengawasan *log* untuk mendeteksi serangan *brute force*
	Konfigurasikan tekno*log*i otentikasi dan enkripsi Web
☐	Untuk sumber daya Web yang memerlukan proteksi minimal dan terdiri dari peserta yang sedikit dan jelas, konfigurasikan otentikasi yang berdasarkan alamat
☐	Untuk sumber daya Web yang memerlukan perlindungan tambahan akan tetapi terdiri dari peserta yang sedikit dan jelas, konfigurasikan otentikasi berdasarkan alamat sebagai garis pertahanan kedua
☐	Untuk sumber daya Web yang memerlukan

Dipenuhi	Tindakan
	perlindungan minimal tetapi tidak terdiri atas peserta yang didefinisikan dengan jelas, konfigurasikan otentikasi dasar atau *digest* (lebih baik)
☐	Untuk sumber daya Web yang memerlukan perlindungan dari *malicious bots*, konfigurasikan otentikasi dasar atau *digest* (lebih baik) atau implementasikan teknik-teknik mitigasi
☐	Untuk sumber daya Web yang memerlukan perlindungan maksimal, konfigurasikan SSL/TLS dengan cipher suite yang kuat
	Konfigurasikan SSL/TLS
☐	Pastikan implementasi SSL/TLS telah dilakukan *patch* sepenuhnya
☐	Gunakan suatu *issued certificate* pihak ketiga untuk otentikasi server (kecuali seluruh sistem yang menggunakan server di atur secara organisasi dapat digunakan *self-signed certificate.*
☐	Untuk konfigurasi yang membutuhkan otentikasi klien tingkat menengah, konfigurasikan server untuk membutuhkan nama pengguna dan password melalui SSL/TLS
☐	Untuk konfigurasi yang membutuhkan otentikasi klien tingkat tinggi, konfigurasikan server untuk membutuhkan sertifikat klien melalui SSL/TLS
☐	Pastikan *cipher suit* yang lemah dinonaktifkan sesuai dengan ketentuan yang berlaku.
☐	Konfigurasikan pengecek keutuhan *file* untuk mengawasi sertifikat Web server

Dipenuhi	Tindakan
☐	Jika hanya SSL/TLS yang digunakan dalam Web server, pastikan akses melalui port TCP daripada 443 di nonaktifkan
☐	Jika sebagian besar lalu lintas pada Web server akan melalui SSL/TLS yang terenkripsi, pastikan bahwa mekanisme *log*ging dan deteksi yang sesuai digunakan dalam Web server (karena pengawasan jaringan tidak efektif menghadapi sesi SSL/TLS)

9.6. Checklist Penerapan Infrastruktur Jaringan yang Aman

Dipenuhi	Tindakan
	Identifikasi lokasi jaringan
☐	Web server diletakkan pada jaringan internal dan dilindungi oleh suatu *firewall*, Web server terletak dalam suatu DMZ atau *host* Web server adalah pihak luar
	Lakukan penilaian terhadap konfigurasi *firewall*
☐	Web server dilindungi oleh suatu *firewall*
☐	Web server, jika berhadapan dengan ancaman yang lebih tinggi atau jika lebih rentan, dilindungi oleh suatu *firewall layer* aplikasi
☐	*Firewall* mengendalikan semua lalu lintas antara Internet dan Web server
☐	*Firewall* memblokir semua *inbound traffic* ke Web server kecuali TCP ports 80 (HTTP) dan/atau 443 (HTTP), jika dibutuhkan
☐	*Firewall* memblokir (dalam hubungannya dengan IDPS) alamat IP atau *subnet* yang dilaporkan IDPS sedang menyerang jaringan organisasi
☐	*Firewall* memberitahukan jaringan atau administrator Web server mengenai kegiatan yang mencurigakan melalui suatu cara yang sesuai
☐	*Firewall* menyediakan penyaring konten (*firewall layer* aplikasi)

Dipenuhi	Tindakan
☐	*Firewall* dikonfigurasi untuk melindungi dari serangan DoS
☐	*Firewall* mendeteksi permintaan URL yang salah format atau serangan terhadap permintaan URL yang dikenal
☐	*Firewall* me-*log* kejadian kritis
☐	*Firewall* dan OS *firewall* di *patch* hingga tingkat aman yang termutakhir atau yang paling tinggi
	Lakukan evaluasi terhadap deteksi intrusi dan sistem pencegahan
☐	IDPS berbasis *host* digunakan untuk Web server yang beroperasi terutama menggunakan SSL/TLS
☐	IDPS dikonfigurasikan untuk memonitor lalu intas jaringan dari dan ke Web server setelah *firewall*
☐	IDPS dibentuk untuk memonitor perubahan atas *file* kritis pada Web server (IDPS berbasis *host* atau pengecek integritas *file*)
☐	IDPS memblokir (dalam hubungannya dengan firewall) alamat IP atau *subnet* yang menyerang jaringan organisasi
☐	IDPS memberitahu para administrator IDPS atau administrator Web server mengenai serangan melalui cara yang sesuai
☐	IDPS dikonfigurasikan untuk memaksimalkan deteksi dengan suatu tingkat penerimaan dari kesalahan yang positif
☐	IDPS dikonfigurasikan untuk me-*log* kejadian

Dipenuhi	Tindakan
☐	IDPS di-update dengan tanda serangan baru secara berkala (misal berdasarkan harian)
☐	IDPS berbasis *host* dikonfigurasikan untuk memonitor sumber daya sistem yang ada pada Web server *host*
	Lakukan penilaian terhadap switch jaringan
☐	*Switch* jaringan digunakan untuk memproteksi jaringan dari penyadap jaringan
☐	*Switch* jaringan dikonfigurasikan pada model keamanan yang tinggi untuk mengalahkan ARP spoofing dan serangan-serangan ARP *poisoning*
☐	*Switch* jaringan dikonfigurasikan untuk mengirimkan semua lalu lintas pada segmen jaringan ke IDPS berbasis jaringan
	Lakukan evaluasi terhadap *load balancer*
☐	*Load balancer* digunakan untuk meningkatkan ketersediaan Web server
☐	*Load balancer* diperbesar dengan Web *cache,* jika dapat diaplikasikan
	Lakukan evaluasi terhadap *reverse proxy*
☐	***Reverse proxy*** digunakan sebagai suatu gerbang pengamanan untuk meningkatkan ketersediaan Web server
☐	***Reverse proxy*** diperbesar dengan penambahan

Dipenuhi	Tindakan
	kecepatan enkripsi, otentikasi pengguna, dan kemampuan penyaring konten, jika dapat diaplikasikan

9.7. Checklist Mengelola Web Server

Dipenuhi	Tindakan
	Lakukan *Log*ging
☐	Gunakan format *log* kombinasi untuk menyimpan *Log* Transfer atau mengkonfigurasi secara manual informasi yang dijabarkan oleh format *log* kombinasi menjadi format standar untuk *Log* Transfer
☐	Aktifkan *Log* Pengacu atau *Log* Agent jika format *log* kombinasi tidak tersedia
☐	Buatlah nama-nama *file log* yang berbeda untuk situs Web virtual yang berbeda yang mungkin di implementasikan sebagai bagian dari suatu Web server fisik tunggal
☐	Gunakan identitas pengguna remote sebagaimana yang ditentukan dalam RFC 1413
☐	Simpanlah *log* pada suatu *host* yang terpisah (sys*log*)
☐	Pastikan terdapat kapasitas yang cukup untuk *log*
☐	Arsipkan *log* berdasarkan pada persyaratan organisasi
☐	Periksa kembali *log* harian
☐	Periksa kembali *log* mingguan (untuk trend long-term yang lebih banyak)
☐	Gunakan *tool* analisis *file log* yang otomatis
	Lakukan pembuatan backup Web server
☐	Buatlah suatu kebijakan backup Web server

Dipenuhi	Tindakan
☐	Lakukan back up Web server secara differential atau incremental berbasis harian hingga mingguan
☐	Buatlah back up Web server secara penuh berbasis mingguan hingga bulanan
☐	Buatlah back up arsip secara periodik
☐	Pelihara suatu copy otoritatif situs Web
	Lakukan pemulihan dari suatu kebobolan
☐	Laporkan insiden kepada kapabilitas tanggap insiden komputer dari organisasi
☐	Isolasi sistem yang bobol atau ambil langkah-langkah lain untuk memagari serangan sehingga informasi tambahan dapat dikumpulkan
☐	Periksa *host-host* yang serupa untuk menentukan jika penyerang juga membobol sistem-sistem lain
☐	Konsultasikan, sebagaimana mestinya, dengan manajemen, konsultan hukum, dan kantor penegak hukum secara cepat.
☐	Analisa intrusi yang terjadi
☐	Kembalikan sistem seperti semula
☐	Ujilah sistem untuk memastikan keamanan
☐	Hubungkan kembali sistem ke jaringan
☐	Monitor sistem dan jaringan terhadap tanda-tanda bahwa penyerang berusaha mengakses kembali sitem atau jaringan kembali.
☐	Dokumentasikan pembelajaran yang diperoleh

Dipenuhi	Tindakan
	Uji Keamanan
☐	Lakukan secara berkala scan kerawanan pada Web server, konten yang dihasilkan secara dinamis, dan jaringan pendukung
☐	Lakukan update scanner kerawanan sebelum pengujian
☐	Perbaiki kekurangan yang teridentifiikasi oleh scanner kerawanan
☐	Lakukan Penetration Testing pada Web server dan infrastruktur jaringan pendukung
☐	Perbaiki kekurangan apapun yang teridentifikasi oleh uji penetrasi
	Melakukan administrasi dari jarak jauh (*remote)* dan *update* konten
☐	Gunakan suatu mekanisme otentikasi yang kuat (misal, pasangan kunci publik/privat, otentikasi dua-faktor)
☐	Batasi *host* yang dapat digunakan untuk mengelola secara remote atau meng*update* konten pada Web server dengan IP address dan ke jaringan internal
☐	Gunakan protokol yang aman (misal, SSH, HTTPS)
☐	Terapkan konsep least privilege pada administrasi yang dilakukan secara remote dan update konten (misalkan meminimalkan hak akses untuk kedua hal ini)
☐	Rubah akun atau password *default* dari perangkat atau aplikasi administrasi yang dilakukan secara

Dipenuhi	Tindakan
	remote
☐	Tidak membolehkan administrasi dilakukan secara remoter melalui internet kecuali dengan menggunakan*Secure Shell* atau VPN
☐	Tidak melakukan sharing *file* Web server pada jaringan internal

Lampiran A Resource Online Keamanan Web Server

Lampiran ini berisi daftar *Resource Online* yang dapat membantu administrator Web server dan lainnya dalam mendapatkan informasi yang lebih banyak tentang keamanan Web server.

Pengamanan Active Content

Judul	URL
Active Software Professionals (ASP) Alliance	http://www.aspalliance.com/
Department of Homeland Security (DHS) Build Security In Portal	https://buildsecurityin.us-cert.gov/
Exploiting Common Vulnerabilities in PHP Hypertext Preprocessor (PHP) Applications	http://www.securereality.com.au/studyinscarlet.txt
Java Security	http://java.sun.com/security
Java Security Frequently Asked Questions	http://www.cs.princeton.edu/sip/faq/java-faq.php3
Open Web Application Security Project (OWASP)	http://www.owasp.org/
OWASP Top Ten	http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
PHP Manual, Chapter 4 Security	http://www.securereality.com.au/studyinscarlet.txt
PHP Security Guide	http://phpsec.org/php-security-guide.pdf

Judul	URL
Web Application Security Consortium	http://www.webappsec.org
www.asp.net	http://www.asp.net
www.cgisecurity .com	http://www.cgisecurity.com/lib

Application Assessment and Code Review Resources

Judul	URL
Detecting Web Application Security Vulnerabilities	http://www.oreillynet.com/pub/a/sysadmin/2006/11/02/webapp_security_scan.html
DHS Build Security In Portal	https://buildsecurityin.us-cert.gov/
OWASP WebGoat	http://www.owasp.org/index.php/OWASP_WebGoat_Project
OWASP WebScarab	http://www.owasp.org/index.php/OWASP_WebScarab_Project
A Process for Performing Securtity Code Reviews	http://www.computer.org/portal/site/security/index.jsp?pageID=security_level1_article&TheCat=1001&path=security/2006/v4n4&*file*=basic.xml
SPI Dynamics	http://www.spidynamics.com
Wapiti	http://wapiti.sourceforge.net/
Watchfire	http://www.watchfire.com
Web Application Security Consortium Articles	http://www.webappsec.org/project/articles/

Digital Certificate Providers (Third-Party Certificate Authorities)

Judul	URL
CertiSign Certificadora Digital Ltda	http://www.certisign.com.br/
Deutsches Forschungsnetz	http://www.pca.dfn.de/
Entrust.net.Ltd	http://www.entrust.net/products/index.htm
GeoTrust Inc.	http://www.geotrust.com/
GlobalSign NV/SA	http://www.globalsign.net/
GoDaddy	http://www.godaddy.com/
IKS GmbH	http://www.iks-jena.de/produkte/ca/
Lanechange.net	http://www.lanechange.net/
Register.com	http://www.register.com/
TC TrustCenter	http://www.trustcenter.da/
Thawte Consulting	http://www.thawte.com/certs/server/request.html

General Web Server Security Resources

Judul	URL
A Look into Web Server and Web Application Attack Signatures	http://www.cgisecurity.com/papers/fingerprint-port80.txt
Center for Education and	http://www.cerias.purdue.edu/

Judul	URL
Research in Information Assurance and Security (CERIAS)	
Computer Emergency Response Team Coordination Center (CERT/CC) Securing Public Web Servers	http://www.sei.cmu.edu/pub/documents/sims/pdf/sim011.pdf
CERT	http://www.cert.org/
Department of Defense (DoD) Web Site Administration Policies and Prosedures	http://www.defenselink.mil/webmaster/policy/dod_web_policy_12071998_with_amendments_and_corrections.htmlhttp://csrc.nist.gov/
National Information Assurance Partnership	http://www.niap.nist.gov
National Institute of Standards and Techno*log*y (NIST) Computer Security Resource Center	http://csrc.nist.gov/
NIST National Vulnerability Database	http://nvd.nist.gov
Office of Management and Budget Circular No. A-130	
Open Source Vulnerability Database	http://www.whitehouse.gov/omb/circulars/a130/
RISKS Forum	http://catless.ncl.ac.uk/Risks/
SANS Institute	http://www.sans.org/
SANS Twenty Most Critical Internet Security	http://www.sans.org/top20.htm

Judul	URL
Vulnerabilities	
Security Configuration Checklists Program for IT Products	http://checklists.nist.gov
Trust Management on the World Wide Web	http://www.firstmonday.dk/issues/issue3_6/khare/
U.S. Department of Energy Computer Incident Advisory Capability (CIAC)	http://www.ciac.org/ciac/
United States Computer Emergency Response Team (US-CERT)	http://www.us-cert.gov
World Wide Web Security Frequently Asked Questions	http://www.w3.org/Security/Faq/www-security-faq.html

Internet Information Services (IIS) Web Server Security Resources

Judul	URL
eEye Advisories and Alerts	http://research.eeye.com/html/advisories/published/index.html
IIS 5.0 Security Checklist	http://www.microsoft.com/technet/prodtechnol/windows2000serv/technologies/iis/tips/iis5chk.mspx
IIS 6 Security	http://www.securityfocus.com/infocus/1765
IIS 6 Security Best Practices	http://technet2.microsoft.com/WindowsServer/en/library/ace052a0-a713-423e8e8c-

Judul	URL
	4bf198f597b81033.mspx
IIS 6 Security Overview	http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/f8f81568-31f2-4210-9982-b9391afc30eb.mspx?mfr=true
IIS Lockdown Tool	http://www.microsoft.com/technet/security/tools/locktool.mspx
National Security Agency (NSA) Guide to the Secure Configuration and Administration of Microsoft IIS 5.0	http://www.nsa.gov/notices/notic00004.cfm?Address=//snac/os/win2k/iis_5_v1_4.pdf
Security in IIS 6.0	http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/f8f81568-31f2-4210-9982-b9391afc30eb.mspx?mfr=true

Miscellaneous Web Security Resources

Title	URL
Dominosecurity.org	http://www.dominosecurity.org/
Honeynet Project	http://project.honeynet.org/
Lotus Domino Security Page	http://www-128.ibm.com/developerworks/lotus/security/
Microsoft Internet Explorer Security Page	http://www.microsoft.com/windows/ie/security/default.asp

Title	URL
Mozilla Security Center	http://www.mozilla.org/security/
Netcraft	http://www.netcraft.com/
Netscape Security Page	http://browser.netscape.com/ns8/security/

Phishing Resources

Title	URL
Anti-Phishing Working Group (APWG)	http://www.antiphishing.org/
Federal Trade Commission (FTC), "How Not to Get Hooked by a 'phishing' Scam"	http://www.ftc.gov/bcp/edu/pubs/consumer/alerts/alt127
Internet Crime Compalint Center (ICCC)	http://www.ic3.gov/
Phish Report Network	http://www.phishreport.net/

WebBot Information

Title	URL
BotSpot	http://www.botspot.com
Configuring the robots.txt *File*	http://www.robotstxt.org/wc/exclusion.html#robotstxt
NIST Mobile Agent Security	http://scrc.nist.gov/mobileagents/project.html
Showing Robots the Door	http://www.ariadne.ac.uk/issue15/

Title	URL
	robots/
University of Maryland Baltimore County (UMBC) AgentWeb	http://agents.umbc.edu/

NIST Publications on System and Network Security

Title	URL
SP 800-18 Revision 1, *Guide for Developing Security Plans for Federal Information Systems*	http://csrc.nist.gov/publications/nistpubs/800-18-Rev1/sp800-18-Rev1-final.pdf
SP 800-26, *Security Self-Assessment Guide for Information Technology Systems*	http://csrc.nist.gov/publications/nistpubs/800-26.pdf
SP 800-27, *Engineering Principles for Information Technology Security*	http://csrc.nist.gov/publications/nistpubs/800-27A/SP800-27-RevA.pdf
SP 800-28, *Guidelines on Active Content and Mobile Code*	http://csrc.nist.gov/publications/nistpubs/800-28/sp800-28.pdf
SP 800-32, *Introduction to Public Key Technology and the government PKI Infrastructure*	http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf
SP 800-34, *Contigency Planning Guide for Information Technology Systems*	http://csrc.nist.gov/publications/nistpubs/800-34/sp800-34.pdf

Title	URL
SP 800-37, *Guide for the Security Certification and Accreditation of Federal Information Systems*	http://csrc.nist.gov/publications/nistpubs/800-37/sp800-37.pdf
SP 800-40 Version 2, *Creating a Patch and Vulnerability Management Program*	http://csrc.nist.gov/publications/nistpubs/800-40-Ver2/sp800-40v2.pdf
SP 800-41, *Guidelines on Firewalls and Firewall Policy*	http://csrc.nist.gov/publications/nistpubs/800-41/sp800-41.pdf
SP 800-42, *Guideline on Network Security Testing*	http://csrc.nist.gov/publications/nistpubs/800-42/sp800-42.pdf
Sp 800-45 Version 2, *Guidelines on Electronic mail Security*	http://csrc.nist.gov/publications/nistpubs/800-45-version2/sp800-45v2.pdf
SP 800-46, *Security for Telecommuting and Broadband Communications*	http://csrc.nist.gov/publications/nistpubs/800-46/sp800-46.pdf
SP 800-52, *Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations*	http://csrc.nist.gov/publications/nistpubs/800-52/sp800-52.pdf
SP 800-53 Revision 1, *Recommended Security Controls for Federal Information Systems*	http://csrc.nist.gov/publications/nistpubs/800-53-Rev1/sp800-53-rev1-final-clean-sz.pdf
SP 800-61, *Computer Security Incident Handling Guide*	http://csrc.nist.gov/publications/nistpubs/800-61/sp800-61.pdf

Title	URL
SP 800-63, *Electronic Authentication Guideline*	http://csrc.nist.gov/publications/nistpubs/800-63/sp800-63V1_0_2.pdf
SP 800-68, Guidance for Securing Microsoft Windows XP Systems for IT Professionals	http://csrc.nist.gov/itsec/download_WinXP.html
SP 800-69, Guidance for Securing Microsoft Windows XP Home Edition: A NIST Security Configuration Checklist	http://csrc.nist.gov/itsec/download_WinXP_Home.html
SP 800-77, Guide to IPsec VPNs	http://csrc.nist.gov/publications/nistpubs/800-77/sp800-77.pdf
SP 800-81, Secure Domain Name System (DNS) Deployment Guide	http://csrc.nist.gov/publications/nistpubs/800-81/sp800-81.pdf
SP 800-83, Guide to Malware Incident Prevention and Handling	http://csrc.nist.gov/publications/nistpubs/800-83/sp800-83.pdf
SP 800-86, Guide to Integrating Forensic Techniques Into Incident Response	http://csrc.nist.gov/publications/nistpubs/800-86/sp800-86.pdf
SP 800-92, Guide to Computer Security *Log* Management	http://csrc.nist.gov/publications/nistpubs/800-92/sp800-92.pdf
SP 800-94, Guide to Intrusion Detection and Prevention System (IDPS)	http://csrc.nist.gov/publications/nistpubs/800-94/sp800-94.pdf

Title	URL
SP 800-95, Guide to Secure Web Services (DRAFT)	http://csrc.nist.gov/publications/drafts.html

Lampiran B Tools dan Aplikasi Keamanan Web

Tools dan aplikasi yang dirujuk pada lampiran ini bukan berarti merupakan daftar tools dan aplikasi yang lengkap digunakan untuk keamanan Web, dan tidak merujuk pada produk tertentu.

Tools Analisis *File*

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Ana*log*	OS yang paling umum digunakan	http://www.ana*log*.cx/intro.html	√	√	Gratis
Deskripsi	Ana*log* merupakan tool analisis *file log* Web server otomatis yang dapat dikompilasi hampir setiap platform yang mendukung bahasa pemrograman C				
Crono*log*	Linux/Unix	http://www.crono*log*.org	√	√	Gratis
Deskripsi	Crono*log* merupakan program yang membaca pesan *log* dari inputnya, yang akan menuliskannya pada suatu *file* output yang dikonstruksi menggunakan suatu template dan data serta waktu terbaru.				
LiveStats 6	Sebagian besar Web Server dan OS	http://www.deepmetrix.com/	√	√	Tidak Gratis
Deskripsi	Livestat6 merupakan suatu tool analisis *log file* Web server yang otomatis				
NetTracker	Sebagian besar Web	http://www.unica.com/	√	√	Tidak Gratis

	server dan OS				
Deskripsi	NetTracker merupakan suatu tool analisis *file log* Web server yang otomatis				
Swatch	Linux/Unix	http://swatch.sourceforge.net	√		Gratis
Deskripsi	Swatch merupakan suatu utility analisis sys*log* Linux/Unix				
Wwwstat	Linux dan Unix dengan Perl telah dipasang	http://ftp.ics.uci.edu/pub/websoft/wwwstat/	√		Gratis
Deskripsi	Wwwstat merupakan suatu tool analisis *file log* Web server otomatis untuk *file-file log berformat access_log*				

Vulnerability Scanning Tools

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Internet Security Systems (IIS) Internet Scanner	Vulnerability Scanner	http://www.iss.net/		√	Tidak Gratis
Deskripsi	*ISS Internet Scanner* merupakan suatu tool untuk *vulnerabily-scanning* berbasis jaringan yang dapat mengidentifikasi lubang keamanan pada *host-host* jaringan				
Metasploit	Vulnerability Scanner	http://www.metasploit.com/	√	√	Gratis

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Deskripsi	Metasploit merupakan suatu tool *freeware vulnerability-scanning* yang dapat mengidentifikasi lubang keamanan pada *host* jaringan				
Nessus	Vulnerability Scanner	http://www.nessus.org/	√	√	Gratis
Deskripsi	Nessus merupakan suatu tool *freeware network-based vulnerability scanning* yang dapat mengidentifikasi lubang keamanan pada *host-host* jaringan				
Retina	Vulnerability Scanner	http://www.eeye.com		√	Tidak Gratis
Deskripsi	Retina merupakan suatu*general-purpose network security scanner* yang dapat mengidentifikasi sejumlah besar kerawanan Web server				
SAINT	Vulnerability Scanner	http://www.saintcorporation.com	√		Tidak Gratis
Deskripsi	SAINT merupakan suatu tool *vulnerability scanning* berbasis jaringan yang dapat mengidentifikasi lubang keamanan pada *host-host* jaringan				
SARA	Vulnerability Scanner	http://www-arc.com/sara/	√		Gratis
Deskripsi	SARA merupakan suatu*freeware network-based vulnerability scanning* yang mengidentifikasi lubang keamanan pada *host* jaringan				

Tool Scanning Aplikasi Web

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Acunetix	Web vulnerability Scanner	http://www.acunetix.com/	√		Tidak Gratis
Deskripsi	Acunetix Web vulnerability scanner merupakan suatu scanner kerawanan aplikasi Web				
AppScan	Web vulnerability Scanner	http://www.watchfire.com/		√	Tidak Gratis
Deskripsi	AppScan merupakan suatu scanner kerawanan aplikasi Web				
Domilock	Vulnerability Scanner	http://domilockbeta.2y.net/		√	Gratis
Deskripsi	Domilock merupakan suatu scanner kerawanan *Web server Lotus Domino* berbasis Web				
Nikto	Common Gateway Interface (CGI) vulnerability scanner	http://www.cirt.net/code/nikto.shtml	√	√	Gratis
Deskripsi	Nikto merupakan suatu scanner yang mengidentifikasi kerawanan dalam scripts CGI				
Paros	Web proxy untuk penguji keamanan aplikasi Web	http://www.paroproxy.org/index.shtml	√	√	Gratis

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Deskripsi	Paros dapat melakukan intersepsi dan modifikasi seluruh *Hypertext Transfer Protocol (HTTP*) dan data *Secure Hypertext Transfer Protocol (HTTPS)* antara server danklien, termasuk cookies dan form field, dan pengujian keamanan aplikasi Web.				
SiteDigger	Suatu scanner kerawanana n Web yang melihat data Google pada situs Anda	http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/sitedigger.htm		√	Gratis
Deskripsi	SiteDigger mencari *Google's cache* untuk menemukan kerawanan, kerusakan, persoalan konfigurasi, informasi *propiertary* dan menarik lainnya pada situs Web				
SSLDigger	SSL cipher interrogator	http://www.foundsome.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/ssldigger.htm		√	Gratis
Deskripsi	SSLDigger merupakan suatu tool yang digunakan untuk mengakses kekuatan server SSL dengan menguji ciphers yang didukung. Terdapat beberapa cipher-cipher dikenal tidak aman				
Wapiti	Web vulnerability scanner	http://wapiti.sourceforge.net/	√	√	Gratis

Tool	Kemampuan	Situs Web	Linux/ Unix	Win 32	Biaya
Deskripsi	Wapiti merupakan suatu scanner kerawanan aplikasi Web yang berbasis *open-source*				
WebInspect	Web vulnerability scanner	http://www.spidynamics.com	√	√	Tidak Gratis
Deskripsi	WebInspect merupakan suatu scanner kerawanan aplikasi Web				
WebScarab	Web application assessment tool	http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project		√	Gratis
Deskripsi	WebScarab merupakan suatu framework untuk menganalisa aplikasi yang berkomunikasi menggunakan HTTP dan protokol HTTP				
Wikto	Web server assessment tool	http://www.sensepost.com/research/wikto/		√	Gratis
Deskripsi	Wikto merupakan suatu scanner kerawanan Web server dan aplikasi Web				

Lampiran D Daftar Kontak

A) Kontak Hukum

1) Polisi

2) Penyidik PPNS Kemkominfo

B) Kontak Insiden

1) Kementerian Kominfo Ditjen APTIKA – Direktorat keamanan Informasi

2) INCIDENT RESPONSE OFFICERINDONESIA COMPUTER EMERGENCY RESPONSE TEAM (ID-CERT)
email:abuse@cert.or.id dan cert@cert.or.id
http://www.cert.or.id/

Lampiran C Daftar Singkatan/Istilah

1. Access Control List **(ACL)**
2. Advanced Encryption Standard **(AES)**
3. Web Adversarial Information Retrieval on the Web **(AIR)**
4. Asynchronous JavaScript and XML (**AJAX)**
5. Application Programming Interface **(API)**
6. Anti-Phishing Working Group **(APWG)**
7. Address Resolution Protocol **(ARP)**
8. Active Server Page **(ASP)**
9. Certificate Authority **(CA)**
10. Completely Automated Public Turing Test to Tell Computers and Humans Apart **(CAPTCHA)**
11. Center for Education and Research in Information Assurance and Security **(CERIAS)**
12. Computer Emergency Response Team Coordination Center **(CERT/CC)**
13. Chief Information Officer **CIO**
14. Computer Gateway Interface **CGI**
15. Common *Log* Format **CLF**
16. Cryptographic Module Validation Program **CMVP**
17. Common Name **CN**
18. Central Processing Unit **CPU**
19. Certificate-Signing Request **CSR**
20. Distributed Denial of Service **DDoS**

21. Demilitarized Zone **DMZ**
22. Domain Name **DN**
23. Domain Name System **DNS**
24. Denial of Service **DoS**
25. Digital Signature Standard **DSS**
26. Federal Information Processing Standard **FIPS**
27. Federal Information Security Management Act **FISMA**
28. *File* Transfer Protocol **FTP**
29. Graphical User Interface **GUI**
30. Hypertext Caching Protocol **HTCP**
31. Hypertext Markup Language **HTML**
32. Hypertext Transfer Protocol **HTTP**
33. Hypertext Transfer Protocol Secure **(HTTPS**) - Transaksi HTTP yang dilindungi melalui protokol Secure Socket Layer (SSL) /Transport Layer Security (TLS)
34. IBM Java Secure Sockets Extension **IBMJSSE**
35. Internet Crime Complaint Center **ICCC**
36. Internet Caching Protocol **ICP**
37. Intrusion Detection and Prevention System **IDPS**
38. Intrusion Detection System **IDS**
39. Internet Engineering Task Force **IETF**
40. Internet Information Server **IIS**
41. Internet Message Access Protocol **IMAP**
42. Internet Protocol **IP**
43. Intrusion Prevention System **IPS**

44. Internet Protocol Security **IPSec**
45. Information System **IS**
46. Internet Service Provider **ISP**
47. Internet Security Systems **ISS**
48. Information System Security Officer **ISSO**
49. Information Systems Security Program Manager **ISSPM**
50. Information Techno*log*y **IT**
51. Information Techno*log*y Laboratory **ITL**
52. Java Runtime Environment **JRE**
53. Java Secure Socket Extension **JSSE**
54. Java Server Page **JSP**
55. Java Virtual Machine **JVM**
56. Local Area Network **LAN**
57. Lightweight Directory Access Protocol **LDAP**
58. Message Authentication Code **MAC**
59. Network Basic Input/Output System **NetBIOS**
60. Network *File* System **NFS**
61. Network Information System **NIS**
62. National Institute of Standards and Techno*log*y **NIST**
63. Network Security Services **NSS**
64. National Vulnerability Database **NVD**
65. Open Database Connectivity **ODBC**
66. Office of Management and Budget **OMB**
67. Open Web Application Security Project **OWASP**

68. Privacy Enhanced Mail **PEM**
69. PHP Hypertext Preprocessor **PHP**
70. Personally Identifiable Information **PII**
71. Public Key Cryptography Standard **PKCS**
72. Public Key Infrastructure **PKI**
73. Redundant Array of Inexpensive Disks **RAID**
74. Robots Exclusion Protocol **REP**
75. Request for Comments **RFC**
76. Senior Agency Information Security Officer **SAISO**
77. Secure Hash Algorithm-1 **SHA-1**
78. Secure Hash Standard **SHS**
79. Security Information and Event Management **SIEM**
80. Simple Mail Transfer Protocol **SMTP**
81. Simple Network Management Protocol **SNMP**
82. Small Office Home Office **SOHO**
83. Special Publication **SP**
84. Structured Query Language **SQL**
85. Secure Shell **SSH**
86. Server Side Includes **SSI**
87. Secure Socket Layer **SSL**
88. Security Support Provider Interface **SSPI**
89. Transmission Control Protocol **TCP**
90. Transport Layer Security **TLS**
91. Trusted Operating System **TOS**

92. User Datagram Protocol **UDP**
93. Uniform Resource Identifier **URI**
94. Uniform Resource Locator **URL**
95. Virtual Local Area Network **VLAN**
96. Virtual Private Network **VPN**
97. Web Cache Coordination Protocol **WCCP**
98. World Wide Web **WWW**
99. Cross-Site Scripting **XSS**

Daftar Pustaka

1. Guidelines on Securing Public Web Servers http://csrc.nist.gov/publications/nistpubs/800-44-ver2/SP800-44v2.pdf