

# Keamanan Aplikasi Berbasis Web

Aplikasi berbasis web memiliki sejumlah kerentanan umum yang dapat di eksploitasi. Open Web Application Security Project (OWASP) Foundation telah merilis sepuluh jenis kerentanan keamanan aplikasi web paling kritis dan umum terjadi. Daftar ini dibuat berdasarkan penelitian dan analisis mendalam terhadap data yang dikumpulkan dari berbagai sumber di seluruh dunia.

Badan Siber dan Sandi Negara (BSSN) juga telah mengeluarkan pedoman teknis keamanan aplikasi berbasis web yang termuat dalam Peraturan BSSN Nomor 4 Tahun 2021 tentang Pedoman Manajemen Keamanan Informasi Sistem Pemerintahan Berbasis Elektronik dan Standar Teknis dan Prosedur Keamanan Sistem Pemerintahan Berbasis Elektronik.

- [Website Vulnerability](#)
  - [A01:2021-Broken Access Control](#)
  - [A02:2021 - Cryptographic Failures](#)
  - [A03:2021 - Injection](#)
  - [A04:2021 - Insecure Design Vulnerabilities](#)
  - [A05:2021 - Security Misconfiguration](#)
  - [A06:2021 - Vulnerable and Outdated Components](#)
  - [A07:2021 - Identification and Authentication Failures](#)
  - [A08:2021 - Software and Data Integrity Failures](#)
  - [A09:2021 - Security Logging and Monitoring](#)
  - [A10:2021 - Server-Side Request Forgery](#)
- [Pengukuran Tingkat Kerentanan \(Severity\)](#)
  - [Common Vulnerability Scoring System \(CVSS\)](#)

- [Metrik Eksploitasi \(Exploitation Metrics\)](#)
- [Metrik Dampak \(Impact Metrics\)](#)
- [Metrik Ancaman \(Threat Metrics\)](#)
- [Metrik Lingkungan \(Environmental Metric\)](#)
- [Standar Teknis Keamanan Aplikasi Berbasis Web](#)
  - [a. Autentikasi](#)
  - [b. Manajemen Sesi](#)
  - [c. Persyaratan Kontrol Akses](#)
  - [d. Validasi Input](#)
  - [e. Kriptografi pada Verifikasi Statis](#)
  - [f. Penanganan Eror dan Pencatatan Log](#)
  - [g. Proteksi Data](#)
  - [h. Keamanan Komunikasi](#)
  - [i. Pengendalian Kode Berbahaya](#)
  - [j. Logika Bisnis](#)
  - [k. Keamanan File](#)
  - [l. Keamanan API dan Web Service](#)
  - [m. Keamanan Konfigurasi](#)

# Website Vulnerability

Open Web Application Security Project (OWASP) Foundation telah merilis daftar kerentanan umum pada aplikasi berbasis web Top 10 adalah daftar yang dibuat oleh Open Web Application Security Project (OWASP) yang berisi sepuluh jenis kerentanan keamanan aplikasi web paling kritis dan umum terjadi. Daftar ini dibuat berdasarkan penelitian dan analisis mendalam terhadap data yang dikumpulkan dari berbagai sumber di seluruh dunia. Dari daftar ini, kita bisa mengetahui ancaman yang paling memiliki dampak besar atau serius pada keamanan aplikasi web.

Referensi:

<https://www.owasp.org>.

# A01:2021-Broken Access Control

## Deskripsi

Broken Access Control adalah salah satu ancaman keamanan aplikasi/web yang paling serius dan menduduki peringkat pertama dalam OWASP Top 10. Kontrol akses adalah proses yang menentukan siapa yang memiliki hak untuk melihat atau menggunakan sumber daya dalam sistem. Kerentanan ini umumnya terjadi ketika sistem gagal menerapkan mekanisme kontrol akses yang memadai diantaranya:

- Tidak terdapat pengecekan akses kontrol dengan memodifikasi URL, *internal application state*, atau *HTML page*, atau menggunakan *custom API attack tool*.
- Mengizinkan *primary key* untuk dapat diganti ke record user lain, yang memungkinkan untuk melihat atau melakukan perubahan pada akun lain.
- Peningkatan sebuah privilege (*Elevation Privilege*), pengguna memiliki akses sebagai admin menggunakan akun *user standard*.
- Manipulasi metadata
- Konfigurasi yang salah pada CORS sehingga menyebabkan API mengakses sistem yang tidak diizinkan

## Dampak

- Pengguna yang tidak berwenang dapat mengakses atau mengubah data yang seharusnya dilindungi.
- Pengambilalihan sistem
- Kerugian finansial dan kerusakan reputasi
- Konsekuensi hukum dan regulasi

## Mitigasi

- Menolak semua akses kecuali ke direktori yang dapat diakses publik
- Melakukan implementasi mekanisme one time password (OTP) pada seluruh aplikasi sehingga meminimalisir penggunaan CORS.
- Menerapkan kontrol akses minimal pada pengguna (*least user privilege*)
- Menonaktifkan *directory listing web server* dan memastikan file metadata (contohnya .git) dan *file backup* tidak ada di dalam web roots.
- Mencatat kegagalan akses kontrol dan alert admin jika diperlukan (seperti adanya kegagalan yang terjadi berulang - ulang).

- Membatasi ukuran API dan akses ke kontroler untuk meminimalisir kerusakan dari *automated attack tooling*.
- *JWT tokens* harus langsung di hilangkan validasinya pada server setelah logout.
- Melakukan pengujian kerentanan keamanan

# A02:2021 - Cryptographic Failures

## Deskripsi

Cryptographic Failures, atau kegagalan kriptografi terjadi ketika mekanisme kriptografi yang digunakan untuk melindungi data tidak berfungsi dengan baik. Kegagalan kriptografi disebabkan antara lain:

- Tidak menerapkan enkripsi / pemilihan algoritma yang tidak tepat.
- Kesalahan dalam implementasi
- Manajemen kunci kriptografi yang kurang memadai.
- Pengguna tidak memverifikasi sertifikat elektronik dari server

## Dampak

Kegagalan kriptografi dapat menimbulkan risiko keamanan siber yang serius, seperti kebocoran data sensitif, manipulasi data, kerugian finansial, dan kerusakan reputasi bisnis

## Mitigasi

- Mengklasifikasikan data yang diproses, disimpan, atau dikirim oleh aplikasi sesuai dengan ketentuan perundang-undangan, persyaratan peraturan, atau kebutuhan bisnis.
- Menetapkan kontrol sesuai klasifikasi.
- Jangan menyimpan data sensitif yang tidak perlu.
- Mengenkripsi semua data sensitif pada *database*
- Menggunakan standar algoritma, protokol yang mutakhir dan kuat, serta menerapkan manajemen kunci yang tepat.
- Mengenkripsi semua data saat transmisi dengan protokol aman seperti TLS dengan *cipher perfect forward secrecy* (PFS), prioritas cipher oleh server, dan parameter yang aman. Menerapkan enkripsi seperti HTTP *Strict Transport Security* (HSTS).
- Menonaktifkan *caching* untuk respons yang berisi data sensitif.
- Menyimpan kata sandi (*password*) menggunakan fungsi *hashing* adaptif dan salted yang kuat.
- Verifikasi secara independen efektivitas konfigurasi dan pengaturan.
- Melakukan audit dan pemantauan keamanan.

# A03:2021 - Injection

## Deskripsi

Kerentanan muncul ketika data yang tidak terpercaya dikirimkan ke interpreter, yang dapat menyebabkan eksekusi perintah yang tidak diinginkan. Kerentanan disebabkan antara lain:

- Kurangnya validasi input
- Sanitasi data yang tidak cukup
- Penggunaan metode pengolahan data yang tidak aman
- Kueri secara dinamis atau permintaan yang tidak diberikan parameter tanpa konteks-peringatan pengalihan

## Dampak

- Penyerang bisa menyisipkan kode berbahaya yang akan dijalankan oleh sistem secara ilegal.
- Kebocoran data sensitif.
- Kerusakan sistem dan data
- Kerugian Finansial
- Kerusakan reputasi
- Konsekuensi hukum dan kepatuhan

## Mitigasi

- Penyimpanan data terpisah dari perintah dan *query*.
- Menggunakan API yang aman yang mencegah penggunaan mesin penerjemah secara keseluruhan, menyediakan sebuah tatap muka berparameter, atau migrasi ke perangkat pemetaan relasi objek.
- Menggunakan daftar (*whitelist*) pada validasi input di sisi server.
- Menerapkan validasi dan sanitasi terhadap input pengguna.
- Menggunakan LIMIT dan kontrol SQL lainnya di antara *query* untuk mencegah terungkapnya data jika terjadi injection.
- Melakukan pengujian keamanan

# A04:2021 - Insecure Design Vulnerabilities

## Deskripsi

*Insecure design vulnerabilities* adalah celah keamanan yang muncul akibat kurangnya perhatian atau perencanaan yang memadai pada aspek keamanan selama fase desain aplikasi atau sistem. Kerentanan ini terjadi ketika arsitektur dan perencanaan sistem tidak memperhitungkan ancaman potensial atau tidak mengikuti praktik keamanan yang terbaik.

Terdapat perbedaan antara desain tidak aman dan implementasi tidak aman. Sebuah desain aman masih bisa memiliki kerusakan implementasi yang mengarah ke kerentanan yang dapat dieksploitasi. **Suatu desain tidak aman tidak dapat diperbaiki oleh sebuah implementasi yang sempurna**. Satu dari faktor yang berkontribusi terhadap desain tidak aman adalah ketiadaan pembuatan profil risiko bisnis yang inheren dalam perangkat lunak atau sistem yang sedang dikembangkan, maka menjadi kegagalan untuk menentukan desain keamanan level yang diperlukan.

## Dampak

Sistem menjadi rentan terhadap eksploitasi bahkan sebelum proses implementasi dan pengujian dimulai.

## Mitigasi

- Membuat dan menggunakan prosedur pengembangan secara aman untuk mengevaluasi dan mendesain kontrol keamanan.
- Menggunakan permodelan ancaman untuk autentikasi darurat, kontrol akses, *business logic*, dan *key flows*.
- Mengintegrasikan kendali dan bahasa keamanan ke dalam *use case*.
- Mengintegrasikan pengujian untuk *frontend* dan *backend*.
- Mensegregasikan lapisan tier pada sistem dan lapisan jaringan berdasarkan kebutuhan eksposur dan proteksi
- Mensegregasikan tenant secara robust dengan desain pada seluruh tier
- Membatasi konsumsi sumber daya oleh pengguna atau layanan



# A05:2021 - Security Misconfiguration

## Deskripsi

*Security misconfiguration* merupakan kondisi sistem komputer, aplikasi, atau infrastruktur IT tidak dikonfigurasi dengan baik untuk melindungi informasi sensitif dan sumber daya organisasi dari ancaman keamanan. Konfigurasi yang salah atau kurangnya pembaruan pada sistem dan aplikasi dapat menciptakan celah keamanan yang dapat dimanfaatkan oleh penyerang. Hal ini sering disebabkan oleh:

- Kelalaian dalam pengelolaan konfigurasi keamanan atau kurangnya pemahaman mengenai praktik keamanan yang terbaik.
- Tidak memiliki pertahanan yang sesuai atau *security hardening* yang diperlukan
- Fitur - fitur yang tidak digunakan masih di enable atau diinstall
- Menggunakan akun dan password default atau tidak pernah diubah.
- Cara menghandle error terlalu informatif kepada user
- Tidak menggunakan fitur keamanan terbaru.
- Pengaturan security pada server aplikasi, framework aplikasi tidak diatur secara aman.
- Server tidak mengirim *security header* atau directives, atau tidak diatur secara aman..
- Menggunakan software yang tidak *update* atau rentan.

## Dampak

Eksploitasi terhadap celah konfigurasi.

## Mitigasi

- Melakukan otomatisasi terhadap proses *hardening* untuk meminimalisir usaha yang diperlukan untuk mengatur *environment* baru yang aman.
- Menghapus atau tidak menginstall fitur dan framework yang tidak digunakan.
- Meninjau dan memperbarui konfigurasi yang sesuai dengan standar keamanan.
- Menerapkan segmentasi antar komponen dengan segmentasi, containerization, atau cloud security groups (ACLs).
- Menerapkan *security headers*.
- Melakukan automasi untuk memverifikasi keefektifan dari konfigurasi dan setting di semua environments.

# A06:2021 - Vulnerable and Outdated Components

## Deskripsi

*Vulnerable and Outdated Components* merupakan kerentanan perangkat lunak dalam aplikasi web yang memiliki celah keamanan atau tidak lagi mendapatkan dukungan berupa pembaruan atau *patch* dari pengembangnya. Komponen dapat mencakup *library*, *framework*, modul, atau bagian lain dari perangkat lunak yang digunakan dalam pengembangan aplikasi web.

Komponen yang tidak diperbarui dapat menjadi sasaran potensial bagi penyerang yang mencari celah keamanan untuk dieksploitasi. Kejadian ini terjadi antara lain karena:

- Komponen tersebut mungkin mempunyai bug atau celah keamanan yang telah diketahui dan dipublikasikan, tetapi belum diperbaiki.
- Tidak teridentifikasinya komponen yang digunakan secara langsung maupun dependensinya.
- Perangkat lunak rentan, tidak didukung, atau sudah usang.
- Tidak memindai kerentanan secara teratur terkait dengan komponen yang digunakan.
- Tidak memperbaiki atau mengupdate platform, kerangka kerja, dan dependensi yang digunakan.
- Tidak menguji kompatibilitas pustaka-pustaka yang diperbarui, ditingkatkan, atau di-patch.
- Tidak mengkonfigurasi komponen secara aman.

## Dampak

Terdapat kerentanan/bug pada komponen yang dapat dieksploitasi

## Mitigasi

- Menghapus dependensi, fitur, komponen, file, dan dokumentasi yang tidak digunakan.
- Inventarisasi versi komponen sisi klien dan sisi server secara terus menerus dan dependensinya
- Memantau secara terus menerus sumber-sumber informasi terkait kerentanan komponen seperti Common Vulnerability and Exposures (CVE) dan National Vulnerability Database (NVD) untuk kerentanan dalam komponen.
- Hanya mengunduh dan menginstal komponen dari sumber resmi melalui tautan yang aman. P

- Memantau pustaka dan komponen yang tidak dirawat atau tidak membuat patch keamanan untuk versi yang lebih lama.

# A07:2021 - Identification and Authentication Failures

## Deskripsi

*Identification and authentication failures* merupakan kelemahan atau kegagalan dalam sistem identifikasi dan autentikasi yang dapat menyebabkan akses tidak sah atau kebocoran informasi sensitif. Hal ini dapat terjadi ketika aplikasi tidak berhasil memverifikasi identitas pengguna atau tidak mengelola akses dengan efektif. Kerentanan ini disebabkan antara lain karena:

- Mengizinkan bruteforce / serangan otomatis seperti isian kredensial, di mana penyerang memiliki daftar nama pengguna dan kata sandi yang valid.
- Mengizinkan penggunaan kata sandi bawaan atau lemah.
- Menggunakan pemulihan (*restore*) kredensial yang lemah atau tidak efektif dan proses lupa kata sandi yang tidak aman.
- Menggunakan password dengan teks biasa, terenkripsi, atau dengan hash yang lemah
- Memiliki otentikasi multi-faktor yang hilang atau tidak efektif.
- Mengekspos ID Sesi di URL
- Tidak membatalkan ID Sesi dengan benar. Sesi pengguna atau token autentikasi (terutama token single sign-on (SSO)) tidak divalidasi dengan benar selama logout atau periode tidak aktif.

## Dampak

Kegagalan dalam aspek ini dapat menciptakan celah bagi penyerang untuk mendapatkan akses yang tidak sah, menyusup ke dalam sistem, atau mengeksploitasi data pribadi.

## Mitigasi

- Menerapkan otentikasi multi-faktor untuk mencegah pengisian kredensial otomatis, brute force, dan serangan penggunaan kembali kredensial yang hilang / dicuri.
- Tidak menggunakan kredensial bawaan apa pun, terutama untuk pengguna admin.
- Menerapkan pemeriksaan kata sandi (password) yang lemah, seperti menguji kata sandi baru.
- Menerapkan kata sandi (password) dengan mengatur panjang sandi, kompleksitas, dan kebijakan rotasi sesuai pedoman yang berlaku (misal NIST 800-63b)
- Pastikan pendaftaran, pemulihan (*restore*) kredensial, dan jalur API diperkuat terhadap serangan enumerasi akun dengan menggunakan pesan yang sama untuk semua hasil.
- Mencatata dan membatasi upaya login yang gagal.

- Menggunakan pengelola sesi built-in sisi server yang aman, menghasilkan ID sesi acak baru dengan entropi tinggi setelah login.
- ID sesi tidak boleh ada di URL, disimpan dengan aman, dan tidak valid setelah keluar, *idle*, dan waktu tunggu absolut.

# A08:2021 - Software and Data Integrity Failures

## Deskripsi

*Software and Data Integrity Failures* merupakan kondisi keaslian, konsistensi, dan keamanan perangkat lunak serta data tidak dapat dipastikan. Kegagalan integritas ini dapat disebabkan oleh berbagai faktor diantaranya:

- Kerentanan dalam source code, serangan siber, dan penggunaan komponen perangkat lunak yang tidak aman
- Kode dan infrastruktur yang tidak mencegah terjadinya pelanggaran integritas
- Aplikasi yang bergantung pada *plugins*, *libraries*, atau *modules* dari sumber yang tidak dipercaya dan *pipeline* yang tidak aman.
- Pembaharuan otomatis yang diunduh tanpa adanya verifikasi integritas.

## Dampak

Akses ilegal/tidak sah, kode yang berbahaya, atau kerusakan sistem.

## Mitigasi

- Menggunakan tanda tangan digital atau mekanisme yang sama untuk memverifikasi bahwa perangkat lunak atau data berasal dari sumber yang diharapkan dan tidak dimanipulasi.
- Memastikan *libraries* dan dependensi seperti npm atau maven menggunakan repositori yang terpercaya.
- Memastikan keamanan rantai pasokan perangkat lunak dengan memverifikasi bahwa komponen tersebut tidak memiliki kerentanan yang sudah diketahui.
- Memastikan adanya proses review ketika mengubah kode dan konfigurasi untuk meminimalisir kemungkinan kode atau konfigurasi berbahaya masuk ke dalam *pipeline* perangkat lunak anda.
- Memastikan *CI/CD pipeline* anda memiliki metode pemisahan, konfigurasi dan akses kontrol yang tepat untuk memastikan integritas kode yang masuk mulai dari proses *build* / pembangunan hingga proses *deployment* / penyebaran perangkat lunak.
- Memastikan data yang belum di tandatangani atau tidak terenkripsi ini tidak terkirim ke klien yang tidak dipercaya tanpa adanya pengecekan integritas atau tanda tangan digital untuk mendeteksi apakah data telah di manipulasi atau pemutaran ulang data yang telah di serialisasi.



# A09:2021 - Security Logging and Monitoring

- **Deskripsi**

Security Logging and Monitoring Failures terjadi ketika sistem tidak mampu mencatat (logging) dan memantau (monitoring) aktivitas keamanan dengan efektif

## **Dampak**

- Kegagalan login dan transaksi dengan nilai yang tinggi tidak di catat.
- Peringatan dan Error tidak menghasilkan pencatatan yang memadai atau catatan pesan yang tidak jelas.
- Log dari aplikasi dan API tidak di monitor untuk aktifitas mencurigakan.
- Log hanya disimpan secara lokal.
- Threshold peringatan yang sesuai dan proses dari respon eskalasi tidak efektif
- Sistem tidak dapat melacak, menganalisis, dan merespons ancaman keamanan dengan cepat dan akurat.

## **Mitigasi**

- Memastikan semua kesalahan login, kontrol akses dan validasi input dari server-side dapat di catat dan disimpan dengan waktu yang cukup untuk analisis forensik.
- Memastikan semua catatan dihasilkan dalam format dimana solusi pengelola catatan dapat dengan mudah digunakan.
- Memastikan data catatan telah di encode dengan benar untuk mencegah injeksi atau serangan pada pencatatan atau sistem monitor.
- Memastikan transaksi dengan nilai yang tinggi memiliki jejak audit dengan kontrol integritas untuk mencegah gangguan dan penghapusan, seperti hanya dapat ditambahkan ke database atau yang mirip seperti itu.
- Melakukan monitoring secara efektif dan memberikan peringatan terhadap aktifitas mencurigakan yang terdeteksi dan merespon secara cepat.
- Membuat atau adopsi sebuah respon insiden dan rencana pemulihan



# A10:2021 - Server-Side Request Forgery

## Deskripsi

Server-Side Request Forgery (SSRF) adalah jenis kerentanan keamanan yang terjadi ketika aplikasi web mengambil remote resource tanpa memvalidasi URL yang disediakan pengguna. Hal ini memungkinkan penyerang untuk memaksa aplikasi mengirimkan permintaan yang telah dimanipulasi ke tujuan yang tidak diharapkan, bahkan ketika aplikasi telah dilindungi oleh firewall, VPN, atau access control list (ACL) lainnya.

## Dampak

- Penyerang dapat memaksa aplikasi untuk mengirim *crafted request* ke destinasi yang tidak diharapkan.
- Kebocoran data sensitif.
- Akses ke jaringan internal
- Serangan lateral dan eskalasi
- Gangguan / kerusakan sistem
- Pengungkapan infrastruktur dan arsitektur
- Pelanggaran kepatuhan

## Mitigasi

- Melakukan segmentasi fungsionalitas akses ke remote resource dalam jaringan terpisah untuk mengurangi dampak SSRF.
- Menerapkan kebijakan firewall “deny by default” atau aturan kontrol akses jaringan yang hanya mengizinkan lalu lintas intranet penting.
- Sanitasi dan validasi semua data input pengguna.
- Menerapkan skema URL, port, dan tujuan dengan positive allow list.
- Menonaktifkan *HTTP redirections*.
- Perhatikan konsistensi URL untuk menghindari serangan seperti DNS rebinding dan kondisi race TOCTOU (time of check, time of use).
- Tidak mengembangkan layanan yang berhubungan dengan keamanan pada sistem yang berada di barisan depan,
- Khusus untuk *frontends* dengan pengguna/grup pengguna yang loyal atau berdedikasi serta dapat dikelola gunakanlah enkripsi jaringan (VPN) pada sistem independen mengingat adanya kebutuhan proteksi yang sangat tinggi.

# Pengukuran Tingkat Kerentanan (Severity)

Kerentanan yang telah diidentifikasi perlu dinilai dan dikelompokkan tingkat kerentanannya (severity). Kerentanan ditangani secara prioritas berdasarkan tingkat kerentanannya.

Referensi:

<https://www.first.org/>

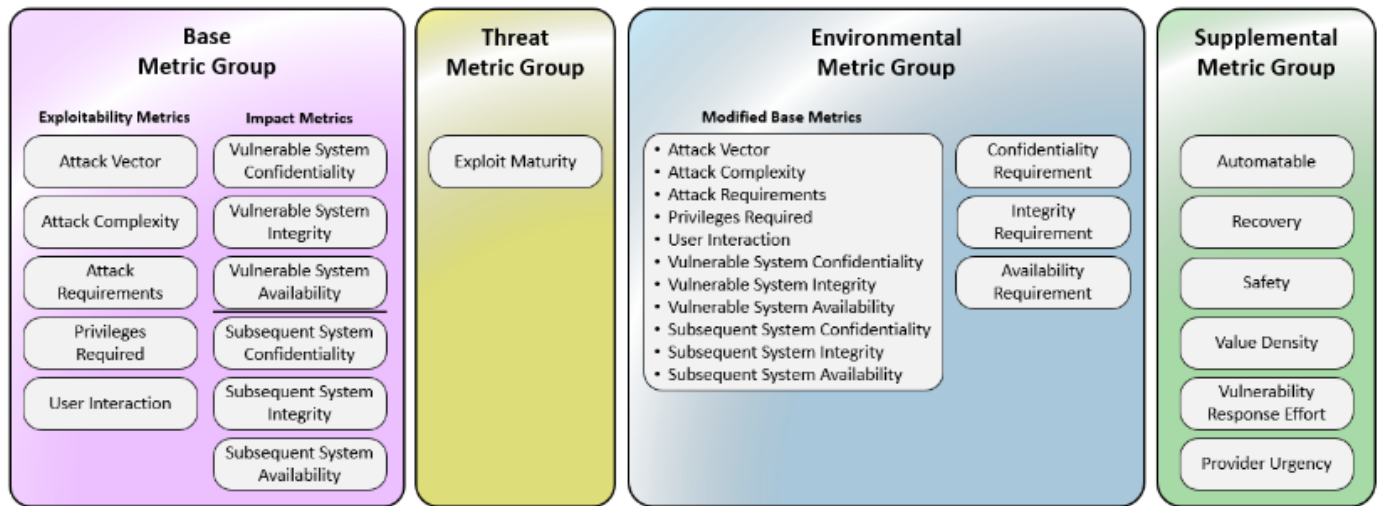
# Common Vulnerability Scoring System (CVSS)

Common Vulnerability Scoring System (CVSS) adalah standar industri yang bebas dan terbuka untuk menilai tingkat keparahan (*severity*) kerentanan keamanan sistem. CVSS menetapkan skor *severity* dari sebuah kerentanan, memungkinkan responden untuk memprioritaskan object mana yang harus diperbaiki terlebih dahulu sesuai dengan tingkat resikonya. CVSS dikembangkan dan dikelola oleh FIRST.Org, organisasi yang bertujuan membantu Tim Tanggap Insiden Siber (*Computer Security Incident Response Team /CSIRT*).

CVSS terdiri dari 4 (empat) metric grup berikut:

- Base score (CVSS-B)
  - Base Score (CVSS-B) dirancang untuk mengukur tingkat keparahan (*severity*) kerentanan, dan tidak digunakan untuk mengukur risiko karena hanya mewakili karakteristik intristik dari kerentanan dan tidak bergantung pada faktor apapun yang terkait dengan ancaman / lingkungan sistem.
- Threat Metric (CVSS-BT)
  - Threat metric mengukur tingkat keparahan (*severity*) berdasarkan beberapa faktor, diantaranya ketersediaan pembuktian (*proof of concept*), atau eksploitasi aktif.
- Environmental Metric (CVSS-BE)
  - Environmental metric mengukur tingkat keparahan (*severity*) yang dihasilkan pada lingkungan komputasi tertentu, diantara bergantung pada faktor mitigasi dan kritikalitas sistem.
- Supplemental Metric (CVSS-BTE)
  - Supplemental metric menggambarkan dan mengukur atribut ekstrinsik tambahan dari kerentanan, dan untuk menambahkan konteks.

Keempat metric tersebut dapat digambarkan sebagai berikut.



Gambar 1. CVSS Metrics

CVSS adalah representasi numerik dari tingkat keparahan kerentanan yang juga dikenal sebagai "**Skor dasar (base score)**" dengan nilai skor dasar bervariasi dari 0 sampai 10. CVSS base score harus dilengkapi dengan analisis lingkungan (environmental metric) dan atribut yang dapat berubah dari waktu ke waktu (threat metric).

Versi terakhir yang dirilis adalah CVSS v.4.0.

Tingkat keparahan (severity) kerentanan dikelompokkan menjadi 4 (empat) sebagaimana ditunjukkan pada tabel berikut.

Tabel 1. CVSS V 3.x dan 4.0 Severity Rating

Kategori Kerentanan	Nilai CVSS
Kritikal (Critical)	9.0 s.d 10.0
Tinggi (High)	7.0 s.d. 8.9
Sedang (Medium)	4.0 s.d 6.9
Rendah (Low)	0.1 s.d. 3.9
None*	0.0

# Metrik Eksploitasi

## (Exploitation Metrics)

### Vektor Serangan (Attack Vector / AV)

Vektor serangan menggambarkan konteks kemungkinan eksploitasi serangan. Asumsinya, serangan melalui jaringan kemungkinannya lebih besar dari pada serangan yang memerlukan akses fisik terhadap perangkat sehingga juga akan berdampak lebih besar. Distribusi metrik vektor serangan ditunjukkan pada Tabel 2 berikut.

Tabel 1. Matrik Vektor Serangan

Nilai	Deskripsi
Network (N)	Sistem yang rentan terhubung dengan jaringan, sehingga dapat dieksploitasi dari jarak jauh
Adjacent (A)	Sistem yang rentan dibatasi pada protokol tertentu, sehingga serangan harus dilakukan pada jaringan yang sama (misal bluetooth, NFC, wifi), jaringan logic lainnya (satu subnet) atau dari dalam domain yang terbatas.
Local (L)	Sistem yang rentan terhubung pada jaringan lokal, sehingga serangan harus dilakukan pada sistem target secara lokal (keyboard, konsol) atau atau melalui emulasi terminal (SSH), atau menggunakan teknis social engineering untuk mengelabui pengguna agar membuka dokumen yang telah disisipi malware.
Physical (P)	Serangan mengharuskan adanya akses secara fisik terhadap sistem yang rentan, misalnya serangan harus dilakukan menggunakan USB.

### Kompleksitas Serangan (Attack Complexity / AC)

Matrik menggambarkan tindakan yang harus dilakukan penyerang agar eksploitasi berhasil dilakukan.

Tabel 2. Kompleksitas Serangan

Nilai	Deskripsi
-------	-----------

Low (L)	Penyerang tidak perlu melakukan tindakan tertentu untuk melakukan eksploitasi terhadap sistem yang rentan. Serangan dapat dilakukan secara berulang.
High (H)	Serangan bergantung pada keamanan pada pertahanan sistem yang rentan. Penyerang harus melakukan metode tambahan untuk melewati keamanan yang ada. Penyerang harus memiliki informasi kredensial sistem.

Persyaratan Serangan (Attack Requirement / AT)

Matrik menggambarkan persyaratan pengembangan dan eksekusi atau variabel yang diperlukan untuk menjalankan serangan.

Tabel 3. Persyaratan Serangan

Nilai	Deskripsi
None (N)	Keberhasilan serangan tidak bergantung pada kondisi pengembangan dan eksekusi pada sistem yang rentan. Eksploitasi terhadap kerentanan dapat dilakukan pada kondisi apapun.
Present (PR)	Keberhasilan serangan bergantung pada kondisi implementasi dan eksekusi tertentu dari sistem yang rentan untuk menjalankan serangan.

Hak Akses yang Diperlukan (Privileges Required/ PR)

Privileges Required menggambarkan tingkat kewenangan yang harus dimiliki oleh penyerang sebelum mengeksploitasi kerentanan, misalnya harus memperoleh kredensial sistem sebelum melakukan serangan. Nilai tertinggi adalah ketika penyerang tidak memerlukan hak akses tertentu untuk mengeksploitasi sistem.

Tabel 4. Matrik Kebutuhan Hak Akses

Nilai	Deskripsi
None (N)	Penyerang tidak diotentikasi sebelum melakukan serangan, sehingga tidak memerlukan akses tertentu terhadap konfigurasi / file pada sistem yang rentan.
Low (L)	Penyerang memerlukan hak akses dengan kemampuan minimal yang dimiliki oleh user biasa yang dapat mengakses file tidak sensitif.
High (H)	Penyerang memerlukan hak akses yang memiliki kontrol signifikan (misal admin) yang memungkinkan akses ke seluruh konfigurasi dan file pada sistem.

**Interaksi Pengguna (User Interaction / UI)**

Matrik menggambarkan persyaratan interaksi pengguna selain penyerang untuk berhasil melakukan serangan ke sistem yang rentan.

Tabel 5. Interaksi Pengguna

Nilai	Deskripsi
None (N)	Sistem yang rentan dapat dieksploitasi tanpa interaksi pengguna, selain penyerang. Misalnya penyerang dari jarak jauh dapat mengirimkan exploit pada sistem dan mengeksekusi kode untuk meningkatkan hak akses.
Passive (P)	Serangan memerlukan interaksi terbatas dari pengguna untuk melakukan eksploitasi.
Active (A)	Serangan memerlukan interaksi pengguna tertentu untuk melakukan eksploitasi, misal pengguna harus menyetujui peringatan terhadap tindakan tertentu.

# Metrik Dampak (Impact Metrics)

Metrik menggambarkan dampak kerentanan yang berhasil dieksploitasi. Namun demikian, analis harus dapat menentukan batasan terhadap dampak akhir yang dapat dicapai oleh penyerang. Ketika mengidentifikasi nilai metrik dampak, perlu diperhitungkan dampak terhadap sistem yang rentan (*vulnerable system impact*) dan dampak diluar sistem yang rentan (*subsequent system impact*) yang ditentukan oleh dua hal yaitu dampak sistem rentan dan dampak selanjutnya yang muncul. Jika kerentanan tidak mempunyai dampak yang terjadi diluar sistem yang rentan, maka *subsequent metric* akan memiliki nilai NONE (N).

Metrik dampak terdiri atas beberapa kategori berikut.

## Kerahasiaan / Confidentiality (VC/SC)

Metrik mengukur dampak kerahasiaan terhadap informasi karena keberhasilan eksploitasi kerentanan. Nilai dampak terhadap kerahasiaan ditunjukkan pada Tabel berikut.

Tabel. Metrik Dampak Kerahasiaan

Nilai Metrik	Dampak pada <i>Vulnerable System</i> (VC)	Dampak pada <i>Subsequent System</i> (SC)
High (H)	Semua informasi didalam sistem dapat diakses penyerang, atau akses terhadap informasi yang terbatas namun berdampak serius, misalnya kredensial sistem.	Semua informasi didalam <i>subsequent system</i> dapat diakses penyerang, atau akses terhadap informasi yang terbatas namun berdampak serius, misalnya kredensial sistem.
Low (L)	Penyerang dapat mengakses terhadap informasi terbatas, namun tidak memiliki kendali atas informasi tersebut sehingga tidak berdampak serius atau tidak menimbulkan kerugian secara langsung terhadap sistem.	Penyerang dapat mengakses terhadap informasi terbatas, namun tidak memiliki kendali atas informasi tersebut sehingga tidak berdampak serius atau tidak menimbulkan kerugian secara langsung terhadap <i>subsequent system</i> .
None (N)	Tidak terdapat informasi yang terungkap / hilang	Tidak terdapat informasi yang terungkap / hilang pada <i>subsequent system</i> .

## Integritas / Integrity (VI/SI)



Metrik mengukur dampak integritas terhadap informasi karena keberhasilan eksploitasi kerentanan. Integritas sistem terdampak ketika penyerang dapat melakukan modifikasi terhadap data / informasi di dalam sistem. Nilai dampak terhadap integritas sistem ditunjukkan pada Tabel berikut.

Tabel. Metrik Dampak Integritas

Nilai Metrik	Dampak pada <i>Vulnerable System</i> (VC)	Dampak pada <i>Subsequent System</i> (SC)
High (H)	Hilangnya perlindungan integritas sistem. Penyerang dapat memodifikasi seluruh file yang dilindungi didalam sistem, atau hanya dapat melakukan modifikasi pada file tertentu, namun modifikasi yang berbahaya dapat berdampak serius pada sistem.	Hilangnya perlindungan integritas sistem. Penyerang dapat memodifikasi seluruh file yang dilindungi didalam sistem, atau hanya dapat melakukan modifikasi pada file tertentu, namun modifikasi yang berbahaya dapat berdampak serius pada <i>subsequent system</i> .
Low (L)	Penyerang dapat melakukan modifikasi, namun tidak memiliki kendali atas akibat dari modifikasi tersebut atau jumlah modifikasi dibatasi sehingga tidak berdampak serius atau tidak menimbulkan kerugian secara langsung terhadap sistem.	Penyerang dapat melakukan modifikasi, namun tidak memiliki kendali atas akibat dari modifikasi tersebut atau jumlah modifikasi dibatasi sehingga tidak berdampak serius atau tidak menimbulkan kerugian secara langsung terhadap <i>subsequent system</i> .
None (N)	Tidak terdapat integritas sistem yang hilang.	Tidak terdapat integritas <i>subsequent system</i> yang hilang.

### Ketersediaan / Availability (VA/SA)

Metrik mengukur dampak ketersediaan terhadap informasi karena keberhasilan eksploitasi kerentanan. Ketersediaan sistem terdampak ketika penyerang dapat mengganggu ketersediaan akses / membuat sistem tidak dapat diakses oleh pengguna. Nilai dampak terhadap ketersediaan sistem ditunjukkan pada Tabel berikut.

Tabel. Metrik Dampak Ketersediaan

Nilai Metrik	Dampak pada <i>Vulnerable System</i> (VC)	Dampak pada <i>Subsequent System</i> (SC)
High (H)	Hilangnya perlindungan ketersediaan sistem. Penyerang dapat menolak seluruh akses kedalam sistem, baik bersifat sementara maupun terus menerus (persistent). Atau penyerang dapat menolak beberapa akses, namun berdampak serius terhadap ketersediaan sistem.	Hilangnya perlindungan ketersediaan layanan <i>subsequent system</i> . Penyerang dapat menolak seluruh akses kedalam <i>subsequent system</i> , baik bersifat sementara maupun terus menerus (persistent). Atau penyerang dapat menolak beberapa akses, namun berdampak serius terhadap ketersediaan <i>subsequent system</i> .

Low (L)	Terdapat gangguan terhadap ketersediaan sistem, namun tidak dapat sepenuhnya menolak layanan kepada pengguna yang sah.	Terdapat gangguan terhadap ketersediaan layanan <i>subsequent system</i> , namun tidak dapat sepenuhnya menolak layanan kepada pengguna yang sah.
None (N)	Tidak terdapat gangguan terhadap ketersediaan sistem .	Tidak terdapat gangguan terhadap ketersediaan <i>subsequent system</i>

# Metrik Ancaman (Threat Metrics)

Metrik ancaman (*threat metrics*) mengukur kondisi teknik eksploitasi / ketersediaan kode untuk mengeksploitasi kerentanan.

Metrik ancaman diukur berdasarkan kematangan exploit / *exploit maturity* (E)

## Kematangan Exploit / Exploit Maturity (E)

Metrik mengukur kemungkinan serangan terhadap kerentanan berdasaeakan kondisi exploit saat ini, ketersediaan kode exploit, atau aktif "in the wild". Ketersediaan exploit untuk publik atau kemudahan instruksi penggunaan meningkatkan kemungkinan serangan, termasuk bagi penyerang yang tidak terampil. Ketersediaan exploit atau instruksi juga dapat berkembang bergantung pada tingkat keberhasilan pembuktian eksploitasi kerentanan (*proof of concept*). Pada beberapa kasus, eksploitasi dapat dijalankan otomatis menggunakan tools serangan tertentu. Informasi terkait teknik eksploitasi / instruksi teknis lainnya selanjutnya akan disebut dengan intelijen ancaman (*threat intelligence*). Pada operasional organisasi disarankan menggunakan banyak sumber threat intelligence.

Daftar kemungkinan nilai kematangan exploit ditunjukkan pada Tabel berikut.

Tabel. Kematangan Exploit (Exploit Maturity)

Nilai Metrik	Deskripsi
Not Defined (X)	<i>Threat intelligence</i> yang handal tidak tersedia untuk menentukan karakteristik kematangan eksploitasi. Not Defined (X) merupakan nilai default.
Attacked (A)	Tersedia <i>threat intelligence</i> : melaporkan serangan terhadap percobaan / keberhasilan eksploitasi, atau terdapat tools untuk memudahkan eksploitasi kerentanan.
Proof of Concept (P)	Terdapat <i>threat intelligence</i> : pembuktian eksploitasi ( <i>proof of concept</i> ) dapat diakses publik, tidak terdapat laporan percobaan eksploitasi kerentanan, tidak terdapat pengetahuan /tools untuk memudahkan eksploitasi yang dapat diakses publik.

Unreported (U)	Terdapat <i>threat intelligence</i> : Tidak terdapat pembuktian eksploitasi ( <i>proof of concept</i> ) dapat diakses publik, tidak terdapat laporan percobaan eksploitasi kerentanan, tidak terdapat pengetahuan /tools untuk memudahkan eksploitasi yang dapat diakses publik.
----------------	--

# Metrik Lingkungan (Environmental Metric)

Metrik memungkinkan pengguna menganalisis nilai berdasarkan tingkat kritikalitas dari aset TI yang terdampak terhadap aspek kerahasiaan, integritas, dan ketersediaan, Metrik merupakan modifikasi dari *base metric*.

## Persyaratan Keamanan Kerahasiaan (*Confidentiality*), Integritas (*Integrity*), dan Ketersediaan (*Availability*)

Setiap aspek memiliki 3 (tiga) kemungkinan nilai: Low, Medium, High, atau nilai defaultnya Not Defined (X). Dampak penuh terhadap nilai metrik lingkungan ditentukan oleh metrik dasar yang telah disesuaikan dengan kebutuhan. Nilai default X, setara dengan nilai metrik High (H). Nilai metrik digambarkan pada Tabel berikut.

Tabel. Metrik Persyaratan Keamanan (*Security Requirements*)

Nilai Metrik	Deskripsi
Not Defined (X)	Nilai default. Nilai ini menggambarkan tidak tercukupinya informasi untuk memilih salah satu nilai yang lain.
High (H)	Hilangnya [kerahasiaan integritas ketersediaan] memiliki dampak buruk paling besar terhadap organisasi atau individu dalam organisasi.
Medium (M)	Hilangnya [kerahasiaan integritas ketersediaan] memiliki dampak serius terhadap organisasi atau individu dalam organisasi.
Low (L)	Hilangnya [kerahasiaan integritas ketersediaan] memiliki dampak yang terbatas terhadap organisasi atau individu dalam organisasi.

# Standar Teknis Keamanan Aplikasi Berbasis Web

Badan Siber dan Sandi Negara (BSSN) telah menerbitkan standar teknis keamanan aplikasi berbasis web yang tertuang dalam Peraturan BSSN Nomor 4 Tahun 2021 tentang Pedoman Manajemen Keamananan Informasi Sistem Pemerintahan Berbasis Elektronik dan Standar Teknis dan Prosedur Keamanan Sistem Pemerintahan Berbasis Elektronik.

# a. Autentikasi

Autentikasi dilakukan dengan:

1. menggunakan manajemen kata sandi untuk proses autentikasi;
2. menerapkan verifikasi kata sandi pada sisi server;
3. mengatur jumlah karakter, kombinasi jenis karakter, dan masa berlaku dari kata sandi;
  - jumlah karakter minimal 12 karakter;
  - kombinasi jenis karakter minimal terdiri dari huruf besar, huruf kecil, simbol dan angka, tidak ada perulangan karakter;
  - masa berlaku dari kata sandi minimal 6 bulan, dan tidak ada perulangan kata sandi
4. mengatur jumlah maksimum kesalahan dalam pemasukan kata sandi;
5. mengatur mekanisme pemulihan kata sandi;
6. menjaga kerahasiaan kata sandi yang disimpan melalui mekanisme kriptografi; dan
7. menggunakan jalur komunikasi yang diamankan untuk proses autentikasi.

## b. Manajemen Sesi

Manajemen sesi dilakukan dengan:

1. menggunakan pengendali sesi untuk proses manajemen sesi;
2. menggunakan pengendali sesi yang disediakan oleh kerangka kerja aplikasi;
3. mengatur pembuatan dan keacakan token sesi yang dihasilkan oleh pengendali sesi;
4. mengatur kondisi dan jangka waktu habis sesi;
5. validasi dan pencantuman session id;
6. perlindungan terhadap lokasi dan pengiriman token untuk sesi terautentikasi; dan
7. perlindungan terhadap duplikasi dan mekanisme persetujuan pengguna.



## c. Persyaratan Kontrol Akses

Persyaratan kontrol akses dilakukan dengan prosedur:

1. menetapkan otorisasi pengguna untuk membatasi kontrol akses;
2. mengatur peringatan terhadap bahaya serangan otomatis apabila terjadi akses yang bersamaan atau akses yang terus-menerus pada fungsi;
3. mengatur antarmuka pada sisi administrator; dan
4. mengatur verifikasi kebenaran token ketika mengakses data dan informasi yang dikecualikan

## d. Validasi Input

Validasi input dilakukan dengan prosedur:

1. menerapkan fungsi validasi input pada sisi server;
2. menerapkan mekanisme penolakan input jika terjadi kesalahan validasi;
3. memastikan runtime environment aplikasi tidak rentan terhadap serangan validasi input;
4. melakukan validasi positif pada seluruh input;
5. melakukan filter terhadap data yang tidak dipercaya;
6. menggunakan fitur kode dinamis;
7. melakukan perlindungan terhadap akses yang mengandung konten skrip; dan
8. melakukan perlindungan dari serangan injeksi basis data.

## e. Kriptografi pada Verifikasi Statis

Kriptografi pada verifikasi statis sebagaimana dilakukan dengan prosedur:

1. menggunakan algoritma kriptografi, modul kriptografi, protokol kriptografi, dan kunci kriptografi sesuai dengan ketentuan peraturan perundang-undangan;
2. melakukan autentikasi data yang dienkripsi;
3. menerapkan manajemen kunci kriptografi; dan
4. membuat angka acak yang menggunakan generator angka acak kriptografi.

## f. Penanganan Eror dan Pencatatan Log

Penanganan eror dan pencatatan log dilakukan dengan prosedur:

1. mengatur konten pesan yang ditampilkan ketika terjadi kesalahan;
2. menggunakan metode penanganan eror untuk mencegah kesalahan terprediksi dan tidak terduga serta menangani seluruh pengecualian yang tidak ditangani;
3. mengatur cakupan log yang dicatat untuk mendukung upaya penyelidikan ketika terjadi insiden;
4. mengatur perlindungan log aplikasi dari akses dan modifikasi yang tidak sah;
5. melakukan enkripsi pada data yang disimpan untuk mencegah injeksi log; dan
6. melakukan sinkronisasi sumber waktu sesuai dengan zona waktu dan waktu yang benar.c. tidak mencantumkan informasi yang dikecualikan dalam pencatatan log;
7. mengatur cakupan log yang dicatat untuk mendukung upaya penyelidikan ketika terjadi insiden;
8. mengatur perlindungan log aplikasi dari akses dan modifikasi yang tidak sah;
9. melakukan enkripsi pada data yang disimpan untuk mencegah injeksi log; dan
10. melakukan sinkronisasi sumber waktu sesuai dengan zona waktu dan waktu yang benar.

## g. Proteksi Data

Proteksi data dilakukan dengan prosedur:

1. melakukan identifikasi dan penyimpanan salinan informasi yang dikecualikan;
2. melakukan perlindungan dari akses yang tidak sah terhadap informasi yang dikecualikan yang disimpan sementara dalam aplikasi;
3. melakukan pertukaran, penghapusan, dan audit informasi yang dikecualikan;
4. melakukan penentuan jumlah parameter dengan meminimalkan parameter yang dibutuhkan untuk request kepada server
5. memastikan data disimpan dengan aman;
6. menentukan metode untuk menghapus dan mengekspor data sesuai permintaan pengguna; dan
7. membersihkan memori setelah tidak diperlukan.

## h. Keamanan Komunikasi

Keamanan komunikasi dilakukan dengan prosedur:

1. menggunakan komunikasi terenkripsi;
2. mengatur koneksi masuk dan keluar yang aman dan terenkripsi dari sisi pengguna;
3. mengatur jenis algoritma yang digunakan dan alat pengujiannya; dan
4. mengatur aktivasi dan konfigurasi sertifikat elektronik yang diterbitkan oleh penyelenggara sertifikasi elektronik.

# i. Pengendalian Kode Berbahaya

Pengendalian kode berbahaya dilakukan dengan prosedur:

1. menggunakan analisis kode dalam kontrol kode berbahaya;
2. memastikan kode sumber aplikasi dan pustaka tidak mengandung kode berbahaya dan fungsionalitas lain yang tidak diinginkan;
3. mengatur izin terkait fitur atau sensor terkait privasi;
4. mengatur perlindungan integritas; dan
5. mengatur mekanisme fitur pembaruan.

## j. Logika Bisnis

Logika bisnis dilakukan dengan prosedur:

1. memproses alur logika bisnis dalam urutan langkah dan waktu yang realistis;
2. memastikan logika bisnis memiliki batasan dan validasi;
3. memonitor aktivitas yang tidak biasa;
4. membantu dalam kontrol antiotomatisasi; dan
5. memberikan peringatan ketika terjadi serangan otomatis atau aktivitas yang tidak biasa.



## k. Keamanan File

Keamanan file dilakukan dengan prosedur:

1. mengatur jumlah file untuk setiap pengguna dan kuota ukuran file yang diunggah;
2. melakukan validasi file sesuai dengan tipe konten yang diharapkan;
3. melakukan pelindungan terhadap metadata input dan metadata file;
4. melakukan pemindaian file yang diperoleh dari sumber yang tidak dipercaya; dan
5. melakukan konfigurasi server untuk mengunduh file sesuai ekstensi yang ditentukan.

# I. Keamanan API dan Web Service

Keamanan API dan *web service* dilakukan dengan prosedur:

1. melakukan konfigurasi layanan web;
2. memverifikasi uniform resource identifier API tidak menampilkan informasi yang berpotensi sebagai celah keamanan;
3. membuat keputusan otorisasi;
4. menampilkan metode RESTful hypertext transfer protocol apabila input pengguna dinyatakan valid;
5. menggunakan validasi skema dan verifikasi sebelum menerima input;
6. menggunakan metode perlindungan layanan berbasis web; dan
7. menerapkan kontrol antiotomatisasi.

## m. Keamanan Konfigurasi

Keamanan konfigurasi dilakukan dengan prosedur:

1. mengonfigurasi server sesuai rekomendasi server aplikasi dan kerangka kerja aplikasi yang digunakan;
2. mendokumentasi, menyalin konfigurasi, dan semua dependensi;
3. menghapus fitur, dokumentasi, sampel, dan konfigurasi yang tidak diperlukan;
4. memvalidasi integritas aset jika aset aplikasi diakses secara eksternal; dan
5. menggunakan respons aplikasi dan konten yang aman.