

Keamanan Informasi & AI

Onno W. Purbo
onno@indo.net.id
onno@itts.ac.id
Twitter onnowpurbo

**Institut Teknologi Tangerang Selatan (ITTS)
OnnoCenter**

Hak Cipta

Karya ini dilisensikan di bawah lisensi **Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)**.

Artinya, **siapapun bebas untuk menggunakan, menyalin, membagikan, dan mengadaptasi** materi ini, **dengan syarat**:

- **Memberikan atribusi yang sesuai** (menyebut sumber asli),
- **Menyertakan lisensi yang sama** jika dimodifikasi atau dikembangkan lebih lanjut.

Lihat detail lisensi di sini <https://creativecommons.org/licenses/by-nd/4.0/legalcode.id>

Disclaimer

Materi dalam buku *Keamanan Informasi & AI* ini disusun untuk tujuan pendidikan, pembelajaran, dan pengembangan kompetensi akademik di bidang keamanan informasi dan keamanan kecerdasan buatan. Seluruh konsep, contoh, studi kasus, kode, dan simulasi yang disajikan dimaksudkan untuk meningkatkan pemahaman, kesadaran risiko, serta kemampuan mitigasi secara etis dan bertanggung jawab, dan hanya boleh digunakan pada lingkungan pembelajaran, penelitian, atau pengujian yang sah dan berizin. Pembahasan mengenai teknik serangan dan kelemahan sistem bertujuan untuk kepentingan defensif dan peningkatan keamanan, bukan untuk mendorong penyalahgunaan atau aktivitas ilegal; oleh karena itu, penulis tidak bertanggung jawab atas penggunaan materi yang melanggar hukum, etika, kebijakan institusi, atau peraturan perundang-undangan yang berlaku. Dataset yang digunakan bersifat dummy, sintetis, atau telah dianonimkan, dan buku ini tidak dimaksudkan sebagai pengganti nasihat hukum atau kebijakan resmi organisasi.

Daftar Isi

Hak Cipta.....	2
Disclaimer.....	2
Daftar Isi.....	3
Kata Pengantar.....	16
PENDAHULUAN.....	17
Capaian Pembelajaran Mata Kuliah (CPMK).....	17
Perangkat Pembelajaran dan Lingkungan Teknis.....	18
Perspektif Pendidikan.....	18
BAB 1: Fondasi Keamanan Informasi.....	19
Keamanan Informasi Bukan Lagi Pilihan.....	19
Keamanan yang Terlambat Selalu Mahal.....	20
Mengapa CIA Triad Menjadi Fondasi Keamanan Informasi.....	20
CIA Triad sebagai Fondasi Keamanan Informasi.....	20
Confidentiality (Kerahasiaan).....	22
Definisi Confidentiality.....	22
Mengapa Confidentiality Penting.....	22
Contoh Pelanggaran Confidentiality.....	23
Mekanisme Perlindungan Confidentiality.....	23
Integrity (Keutuhan Data).....	24
Definisi Integrity.....	24
Mengapa Integritas Sangat Penting.....	24
Contoh Pelanggaran Integrity.....	25
Mekanisme Perlindungan Integrity.....	25
Availability (Ketersediaan).....	26
Definisi Availability.....	26
Makna Availability dalam Keamanan Informasi.....	26
Contoh Gangguan Availability.....	27
Mekanisme Perlindungan Availability.....	27
Hubungan dan Trade-off dalam CIA Triad.....	27
Ancaman Modern: Dunia Digital yang Semakin Kompleks.....	28
Mengapa Keamanan Informasi Terus Mengalami Kegagalan.....	28
Keamanan sebagai Proses Berpikir Berkelanjutan.....	29
Membangun Mentalitas Keamanan.....	29
PRAKTEK 1: Identifikasi Aset & Ancaman.....	30
Tujuan Pembelajaran.....	30
Konsep Inti yang Digunakan.....	30
Tools yang Digunakan.....	30
Langkah Praktikum (Step-by-Step).....	30
Langkah 1 — Memilih Sistem Nyata.....	30
Langkah 2 — Identifikasi Aset.....	31
A. Aset Data.....	31
B. Aset Sistem.....	31

C. Aset Pengguna.....	31
Langkah 3 — Identifikasi Ancaman.....	32
Langkah 4 — Pemetaan Ancaman ke CIA Triad.....	32
Langkah 5 — Analisis Dampak.....	33
Output yang Harus Dikumpulkan.....	33
1. Diagram Aset & Ancaman.....	33
2. Tabel Analisis CIA Impact.....	33
Evaluasi Penilaian.....	34
Refleksi Penutup Modul.....	34
BAB 2: Manajemen Risiko & Threat Modeling.....	35
Dari Kesadaran ke Analisis.....	35
Konsep Risiko dalam Keamanan Informasi.....	35
Manajemen Risiko sebagai Proses Berkelanjutan.....	35
Threat Modeling: Memetakan Cara Sistem Bisa Gagal.....	36
Praktik: Threat Modeling pada Aplikasi Fiktif.....	36
Penilaian Risiko dan Risk Scoring.....	36
Output dan Keterampilan yang Dihasilkan.....	37
Keamanan sebagai Keputusan yang Disadari.....	37
STRIDE: Kerangka Klasifikasi Ancaman Keamanan.....	37
Tujuan Penggunaan STRIDE.....	37
Enam Kategori Ancaman STRIDE.....	38
1. Spoofing (Pemalsuan Identitas).....	38
2. Tampering (Perubahan Data Tanpa Izin).....	38
3. Repudiation (Penyangkalan Aktivitas).....	38
4. Information Disclosure (Kebocoran Informasi).....	39
5. Denial of Service (Gangguan Layanan).....	39
6. Elevation of Privilege (Peningkatan Hak Akses).....	39
Hubungan STRIDE dengan CIA Triad.....	40
Contoh Penggunaan STRIDE dalam Threat Modeling.....	40
Contoh Tabel Risk Scoring (Manual).....	41
Skala Penilaian (Disepakati di Awal).....	41
Tabel Analisis Risiko.....	41
Klasifikasi Level Risiko (Contoh).....	42
Contoh Interpretasi (Untuk Laporan Mahasiswa).....	42
Versi Ringkas (Untuk Kelas Non-Teknis).....	42
Skill yang Dilatih dengan Tabel Ini.....	42
PRAKTEK 2: Risk Assessment & Threat Modeling.....	43
Tujuan.....	43
Konsep yang Digunakan.....	43
Tools yang Digunakan.....	43
Langkah Praktek (Step-by-Step).....	44
Langkah 1 — Menentukan Aplikasi Fiktif.....	44
Langkah 2 — Memetakan Sistem Secara Sederhana.....	44
Langkah 3 — Menentukan Aset Utama.....	44

Langkah 4 — Identifikasi Ancaman dengan STRIDE.....	45
Langkah 5 — Menilai Likelihood dan Impact.....	45
Langkah 6 — Menghitung Risk Score.....	45
Langkah 7 — Menentukan Prioritas Mitigasi.....	46
Output yang Harus Dikumpulkan.....	46
Template Risk Register.....	47
A. Informasi Umum (Diisi di Bagian Atas Spreadsheet).....	47
B. Skala Penilaian (Disepakati di Awal).....	47
Likelihood (Kemungkinan Terjadi).....	47
Impact (Dampak Jika Terjadi).....	47
C. Tabel Risk Register Utama.....	48
D. Klasifikasi Level Risiko (Contoh).....	48
E. Contoh Pengisian (Sebagai Referensi).....	49
F. Ringkasan Prioritas Risiko (Opsional tapi Disarankan).....	49
Keterampilan yang Dilatih.....	49
Catatan Penting untuk Mahasiswa.....	49
Catatan Tambahan.....	50
Penutup.....	50
BAB 3: Kriptografi untuk Praktisi.....	51
Kriptografi dalam Dunia Nyata (Tanpa Ribet).....	51
Hashing — Bukan untuk Disimpan Balik.....	51
Encryption — Saat Data Harus Bisa Dibaca Lagi.....	52
Key Management — Bagian yang Paling Sering Diabaikan.....	53
Kesalahan Umum Kriptografi (Ini yang Paling Sering Terjadi).....	53
PRAKTEK 3: Kriptografi Praktis.....	54
Tujuan.....	54
Konsep Kunci (Wajib Dipahami Sebelum Koding).....	54
Hash ≠ Enkripsi.....	54
Golden Rule Password Security.....	54
Tools & Environment.....	55
Sistem.....	55
Install Dependency.....	55
Praktek 1 — Hash Password dengan Benar.....	55
Contoh SALAH (Sering Terjadi di Sistem Nyata).....	55
Cara BENAR — Password Hashing dengan bcrypt.....	55
Prinsip bcrypt.....	55
Script Aman (Production-Ready).....	55
Penjelasan.....	56
Real Case.....	56
Praktek 2 — Enkripsi Data Pribadi (AES).....	56
Konsep AES (Symmetric Encryption).....	57
Enkripsi AES dengan Python (cryptography).....	57
Step 1 — Generate Key.....	57
Step 2 — Enkripsi Data.....	57

Step 3 — Dekripsi Data.....	57
Perbedaan Hash vs Enkripsi (Nyata).....	57
Praktek 3 — Simulasi Kesalahan Fatal (Plaintext Storage).....	58
Sistem Salah (TAPI SERING ADA).....	58
Sistem Benar.....	58
Praktek 4 — GnuPG (GNU Privacy Guard).....	58
Tujuan Pembelajaran.....	58
Konsep Dasar GnuPG.....	58
Kriptografi Asimetris.....	58
Kapan GnuPG Dipakai?.....	59
Setup Awal (Ubuntu 24.04).....	59
BAGIAN 1 — Membuat Key Pair GnuPG.....	59
Step 1 — Generate Key.....	59
Step 2 — Lihat Key yang Dibuat.....	60
BAGIAN 2 — Enkripsi File (Public Key).....	60
Contoh File Sensitif.....	60
Enkripsi File.....	60
BAGIAN 3 — Dekripsi File (Private Key).....	60
BAGIAN 4 — Enkripsi & Dekripsi Message (Text).....	60
Enkripsi Pesan.....	60
Dekripsi Pesan.....	61
BAGIAN 5 — Enkripsi Simetris (Tanpa Public Key).....	61
Enkripsi File.....	61
Dekripsi.....	61
BAGIAN 6 — Proteksi AES Key (Best Practice).....	61
Contoh Kasus Nyata.....	61
Saat Dibutuhkan:.....	61
BAGIAN 7 — Export & Import Public Key (Distribusi Aman).....	62
Export Public Key.....	62
Import Public Key.....	62
BAGIAN 8 — Kesalahan Fatal (WAJIB DIHINDARI).....	62
BAGIAN 9 — Backup & Recovery Key.....	62
Integrasi dengan Sistem Nyata.....	62
Output Skill Mahasiswa.....	62
Analisis Kesalahan Umum (Wajib Masuk Laporan).....	63
Penutup.....	63
BAB 4: Keamanan Jaringan & Traffic Analysis.....	64
Network Attack Basics.....	64
Konsep IDS & IPS.....	64
Cara IDS/IPS Mengenali Serangan.....	65
Penutup.....	65
PRAKTEK 4: Analisis Trafik Jaringan.....	66
Tujuan.....	66
Konsep Dasar (Wajib Dipahami).....	66

Packet.....	66
Protocol.....	66
Payload.....	67
Tools Utama.....	67
Persiapan Lingkungan (Ubuntu 24.04).....	67
Install Wireshark.....	67
Cek Interface Jaringan.....	67
Praktik 1 — Capture Trafik HTTP (Tidak Aman).....	68
Tujuan.....	68
Langkah 1 — Jalankan Wireshark.....	68
Langkah 2 — Akses Website HTTP.....	68
Langkah 3 — Filter HTTP di Wireshark.....	68
Praktik 2 — Simulasi Data Sensitif Bocor.....	69
Kirim Data Login Palsu (HTTP).....	69
Praktik 3 — Capture Trafik HTTPS (Aman).....	69
Akses HTTPS.....	69
Perbandingan HTTP vs HTTPS.....	70
Praktik 4 — Analisis Metadata HTTPS.....	70
Praktik 5 — Capture & Analisis dengan Python (Advanced).....	70
Install Library.....	70
Script Capture Packet Sederhana.....	70
Output yang Wajib Dikumpulkan.....	71
Screenshot.....	71
Laporan Singkat (1–2 halaman).....	71
Refleksi Penting untuk Mahasiswa.....	71
BAB 5: Deteksi Serangan (Rule-Based & AI-Based).....	72
Teori: Signature Detection vs Anomaly Detection.....	72
Signature-Based Detection (Rule-Based).....	72
Anomaly-Based Detection (AI-Based / Behavior-Based).....	73
Perbandingan Singkat: Signature vs Anomaly.....	73
Kelemahan Rule-Based Detection (Kenapa Tidak Cukup?).....	74
1. Tidak Bisa Melihat Serangan Baru (Zero-Day).....	74
2. Rule Explosion (Ledakan Aturan).....	74
3. Mudah Diakali (Bypassable).....	74
4. Tidak Memahami Konteks.....	75
Mengapa Kita Harus Bergerak ke Sistem yang Lebih Cerdas?.....	75
Mindset Penting untuk Mahasiswa.....	75
PRAKTEK 5: Intrusion Detection Dasar.....	76
Tujuan.....	76
Konsep Inti (Wajib Dipahami).....	76
1. Apa itu Intrusion Detection System (IDS).....	76
2. Signature-based vs Anomaly-based Detection.....	76
Signature-based Detection.....	76
Anomaly-based Detection.....	77

Tools yang Digunakan (Open Source).....	77
BAGIAN A — IDS Signature-Based dengan Snort.....	77
Step 1 — Instalasi Snort (Ubuntu 24.04).....	77
Step 2 — Menjalankan Snort dalam Mode IDS.....	78
Step 3 — Membuat Rule IDS Sederhana (Signature).....	78
Step 4 — Simulasi Serangan Sederhana (Ping Flood).....	78
Step 5 — Simulasi Serangan Port Scanning.....	79
Output Snort.....	79
BAGIAN B — IDS Sederhana dengan Python (Anomaly-Based).....	79
Tujuan.....	79
Step 1 — Contoh Log yang Dianalisis.....	79
Step 2 — Script Python IDS Sederhana.....	80
Interpretasi Output.....	80
Diskusi Keamanan (Penting).....	80
Output Praktikum (Yang Dikumpulkan).....	80
Kesimpulan.....	81
BAB 6: Perlindungan Data Pribadi.....	82
Pendahuluan: Dari Hukum ke Keyboard.....	82
Teori 1 — UU Perlindungan Data Pribadi Indonesia (UU PDP).....	82
Prinsip Kunci UU PDP (Versi Teknis).....	82
Teori 2 — Data Lifecycle: Hidup, Digunakan, Lalu Dihapus.....	83
Tahap 1 — Data Collection (Pengumpulan).....	83
Tahap 2 — Consent (Persetujuan).....	84
Tahap 3 — Data Processing & Storage.....	84
Tahap 4 — Data Sharing (Jika Ada).....	84
Tahap 5 — Data Retention & Deletion.....	85
Penutup: PDP adalah Soal Etika Teknis.....	85
PRAKTEK 6: Identifikasi & Klasifikasi Data Pribadi (PDP).....	86
Tujuan.....	86
Konsep Inti (Wajib Dipahami).....	86
Data Pribadi Umum vs Sensitif.....	86
Data Lifecycle (Siklus Hidup Data).....	87
Tools yang Digunakan (100% Open Source).....	87
Praktek Step-by-Step.....	87
STEP 1 — Siapkan Dataset Dummy.....	87
STEP 2 — Identifikasi Kolom Data Pribadi (Python).....	88
STEP 3 — Klasifikasi Tingkat Sensitivitas.....	88
STEP 4 — Tentukan Kontrol Pengamanan.....	89
STEP 5 — Buat Data Classification Matrix.....	90
Output Praktikum (Wajib Dikumpulkan).....	90
Kesalahan Umum (Dan Harus Dihindari).....	90
Skill Nyata yang Didapat.....	91
BAB 7: Audit Kebocoran Data Pribadi.....	92
Tujuan.....	92

Teori Inti.....	92
Data Breach Lifecycle.....	92
Incident Response Basics.....	93
Tahapan Incident Response.....	93
Pendekatan Audit: Investigatif & Defensif.....	94
Kesalahan Umum yang Harus Dihindari.....	95
Mengapa Bab Ini Penting?.....	95
PRAKTEK 7: Audit Kebocoran Data.....	96
Tujuan.....	96
Konsep Inti.....	96
Incident Handling.....	96
Digital Footprint.....	96
Tools (100% Open Source).....	97
Skenario Praktikum (Realistis).....	97
STEP 1 — Siapkan Data Kebocoran (Simulasi).....	97
STEP 2 — OSINT Simulasi: Deteksi Data yang Terekspos.....	98
STEP 3 — Klasifikasi Dampak Kebocoran.....	99
STEP 4 — Amankan Bukti Audit (Chain of Custody).....	99
STEP 5 — Susun Incident Report (Template Nyata).....	100
1. Identitas Insiden.....	100
2. Kronologi Singkat.....	100
3. Jenis Data Terekspos.....	100
4. Dampak Potensial.....	100
5. Tindakan Darurat.....	100
6. Rekomendasi.....	100
Output yang Dikumpulkan.....	101
Kesalahan Umum (Sering Terjadi).....	101
Skill Nyata yang Didapat.....	101
BAB 8: Keamanan AI sebagai Sistem (AI sebagai Attack Surface).....	102
Pendahuluan: AI Tidak Berdiri Sendiri.....	102
AI sebagai Attack Surface Baru.....	102
Apa itu Attack Surface?.....	102
Perubahan Fundamental yang Dibawa AI.....	103
Komponen Sistem AI yang Perlu Diamankan.....	103
1. Data (Training, Validation, dan Input).....	103
2. Model AI.....	104
3. Pipeline & Infrastruktur.....	104
4. Output & Keputusan AI.....	104
Ancaman Keamanan Khas Sistem AI.....	105
AI dan Perubahan Paradigma Keamanan.....	105
Dari “Secure the System” ke “Secure the Decision”.....	105
Hubungan dengan CIA Triad.....	106
Mindset yang Harus Dibangun.....	106
Transisi ke Bab Selanjutnya.....	106

PRAKTEK 8: Analisis Keamanan Sistem AI (Pre-Attack Lab).....	107
Tujuan.....	107
Konsep Inti.....	107
Tools yang Digunakan (Ringan & Open Source).....	107
Skenario Praktikum (Wajib Dipilih 1).....	107
Pilihan Skenario:.....	107
Langkah Praktikum (Step-by-Step).....	108
STEP 1 — Memahami Sistem AI Secara Utuh.....	108
STEP 2 — Identifikasi Komponen Sistem AI.....	108
STEP 3 — Membuat Diagram AI sebagai Sistem.....	109
STEP 4 — Identifikasi Attack Surface.....	109
STEP 5 — Pemetaan CIA Triad pada Sistem AI.....	109
STEP 6 — Analisis Risiko Awal (Tanpa Angka Rumit).....	110
STEP 7 — Refleksi Pra-Serangan (WAJIB).....	110
Output yang Wajib Dikumpulkan.....	110
Evaluasi Penilaian.....	110
Kesalahan Umum (Wajib Dihindari).....	111
Insight Penting.....	111
BAB 9: Keamanan AI (AI sebagai Target Serangan).....	112
Fokus: AI tidak otomatis aman.....	112
Mengubah Cara Pandang: AI sebagai Attack Surface.....	112
Adversarial Attack — Menipu AI Tanpa Merusak Sistem.....	112
Apa itu Adversarial Attack?.....	112
Mengapa Ini Berbahaya?.....	113
Data Poisoning — Meracuni AI dari Dalam.....	113
Apa itu Data Poisoning?.....	113
Model Leakage — Ketika Model “Bocor” Tanpa Disadari.....	114
Apa itu Model Leakage?.....	114
Refleksi Penting untuk Mahasiswa.....	115
Mindset yang Ingin Dibangun.....	115
PRAKTEK 9: Serangan terhadap AI.....	116
Tujuan.....	116
Konsep Inti (Fundamental yang Wajib Dipahami).....	116
Adversarial Example.....	116
Data Poisoning.....	116
Tools yang Digunakan.....	117
Praktikum Lengkap (Step-by-Step).....	117
Bagian A — Setup Lingkungan.....	117
Bagian B — Train Model AI Sederhana.....	117
Dataset & Model.....	117
Bagian C — Adversarial Example (Serangan Input).....	118
Tujuan.....	118
Ambil Satu Data Uji.....	118
Manipulasi Input Secara Halus.....	118

Diskusi Kritis.....	118
Bagian D — Data Poisoning (Serangan Data Training).....	119
Tujuan.....	119
Racuni Dataset.....	119
Train Model dengan Data Rusak.....	119
Bagian E — Analisis & Refleksi.....	119
Output Praktikum (Wajib Dikumpulkan).....	120
Insight Penutup (Inspiratif).....	120
BAB 10: AI untuk Keamanan Informasi.....	121
Fokus: AI sebagai Alat Pertahanan.....	121
Mengapa Keamanan Modern Butuh AI?.....	121
Teori 1 — Machine Learning untuk Anomaly Detection.....	122
Apa itu Anomaly Detection?.....	122
Pendekatan Rule-Based vs ML-Based.....	122
Jenis Pendekatan ML untuk Anomaly Detection.....	123
Teori 2 — Security Analytics.....	123
Apa itu Security Analytics?.....	123
Perbedaan Log Monitoring vs Security Analytics.....	123
Peran AI dalam Security Analytics.....	124
Mindset Penting: AI adalah Copilot, Bukan Pengganti.....	124
Kenapa Bab Ini Penting untuk Mahasiswa & Praktisi?.....	125
Ringkasan Kunci (Takeaways).....	125
PRAKTEK 10: AI untuk Deteksi Anomali.....	126
Tujuan.....	126
Konsep Inti.....	126
Tools (Open Source).....	126
Skenario Data Log yang Real (Pilih salah satu atau gabungkan).....	127
Tahap Praktikum (Step-by-step).....	127
0. Setup Environment di Ubuntu 24.04.....	127
1. Ambil Dataset Log.....	128
Opsi A — pakai log asli mesin (paling real).....	128
Opsi B — bikin dataset latihan (biar kontrol).....	128
2. Prinsip Feature Engineering (Supaya “AI” mengerti).....	128
Implementasi: Pipeline Lengkap (Python).....	128
A. scripts/parse_authlog.py — parse auth.log → dataset fitur.....	129
B. scripts/train_models.py — latih Isolation Forest + KMeans.....	132
C. scripts/detect_anomalies.py — scoring, ranking, export.....	134
Cara Membaca Hasil.....	136
Simulasi Serangan Ringan (Aman untuk Lab).....	137
(Optional) Amankan Model & Output dengan GnuPG.....	137
1. Generate key (sekali saja).....	137
2. Enkripsi output CSV.....	137
3. Enkripsi model.....	137
Output yang Wajib Dikumpulkan.....	138

Challenge Upgrade (Kalau Mau Naik Level).....	138
BAB 11: AI untuk Audit Data & Privasi.....	139
Tujuan.....	139
Gambaran Besar: Mengapa Audit Data Butuh AI?.....	139
Data Sensitivity Detection.....	139
PII Detection (Personally Identifiable Information).....	140
Pendekatan AI dalam Audit Privasi.....	141
Studi Kasus Mini (Untuk Dipikirkan).....	141
Kesalahan Umum dalam Audit Data.....	142
Penutup: Dari Audit ke Budaya Privasi.....	142
PRAKTEK 11: AI untuk Audit Data Pribadi.....	143
Tujuan.....	143
Konsep: PII Detection (Realistic).....	143
Tools (Open Source).....	144
1. Python (Utama).....	144
2. GnuPG (GNU Privacy Guard).....	144
Tahap Praktikum (Step-by-step).....	144
Tahap 0 — Setup Environment (Ubuntu 24.04).....	144
Tahap 1 — Siapkan Dataset Contoh (Realistic).....	144
Tahap 2 — Bangun “AI Data Scanner” Versi 1 (Regex Engine).....	145
Tahap 3 — “Tandai Risiko” + Output yang Bisa Dipakai (CSV Ringkas).....	148
Tahap 4 — Upgrade ke “Regex + ML ringan” (Opsional tapi Keren).....	149
Tahap 5 — Aksi Setelah Audit dengan GnuPG.....	150
1. Generate Key (untuk kelas, pakai mode cepat).....	150
2. Enkripsi Laporan Audit.....	150
3. Sign Laporan (Integritas).....	150
Output yang Wajib Dikumpulkan (Sesuai Modul).....	151
Challenge (Fun & Menantang).....	151
Bonus: Redaction (Contoh Kode yang Real Dipakai).....	151
BAB 12 — Privacy-Preserving AI.....	153
Gambaran Besar Privacy-Preserving AI.....	153
Differential Privacy (DP).....	153
Federated Learning (FL) — Konsep Dasar.....	154
Perbandingan Singkat.....	155
Mindset yang Harus Dibawa Pulang.....	156
PRAKTEK 12: Privacy-Preserving AI (Differential Privacy).....	157
Tujuan.....	157
Konsep Inti yang Perlu Dipahami.....	157
1. Apa itu Differential Privacy?.....	157
2. Kenapa “Noise Injection” masuk akal?.....	157
3. Istilah penting (biar tidak bingung).....	157
Tools (Open Source).....	158
Skenario Real yang Bisa Diimplementasi.....	158
LANGKAH PRAKTIKUM (Ubuntu 24.04).....	158

Step 0 — Setup Environment.....	158
Step 1 — Siapkan Dataset Contoh (atau pakai data Anda).....	159
Step 2 — Implementasi Differential Privacy (Noise Injection + Clipping).....	159
Step 3 — Bandingkan Utility Secara “Data Scientist”.....	163
Output Praktikum (Yang Dikumpulkan).....	163
BONUS: Amankan File Rilis dengan GnuPG (Enkripsi + Tanda Tangan).....	163
1. Generate key (sekali saja).....	163
2. Enkripsi file untuk penerima (recommended).....	164
3. Decrypt (di sisi penerima).....	164
4. Tanda tangan file (integrity).....	164
Diskusi Kelas.....	164
BAB 13: Proyek Akhir (AI Security Project).....	165
Tujuan.....	165
Kerangka Umum Proyek.....	165
Pilihan Proyek.....	166
AI Deteksi Phishing.....	166
AI Audit Data Pribadi.....	166
AI IDS Sederhana.....	167
AI Privacy Checker.....	167
Output yang Wajib Dikumpulkan.....	168
Catatan Penting untuk Mahasiswa.....	168
Penutup.....	168
PRAKTEK 13 — Proyek Akhir AI Security.....	169
Tujuan.....	169
Pilihan Proyek (Pilih 1).....	169
Struktur Wajib Proyek (Sama untuk semua).....	169
Tahap 0 — Setup Environment (Ubuntu 24.04).....	169
Tahap 1 — Keamanan Data Proyek (Wajib): GnuPG untuk Dataset & Output..	170
1. Buat key GPG (untuk proyek).....	170
2. Enkripsi dataset (contoh).....	170
Tahap 2 — Pilih Proyek + Jalankan Step-by-step.....	171
OPSI A — AI Phishing Detector (Recommended).....	171
1. Siapkan dataset (realistis tapi aman).....	171
2. Buat training script (ML sederhana + explainable).....	171
3. Buat detector + alasan (top keywords).....	172
OPSI B — AI Audit PDP (Privacy Audit Tool).....	174
1. Dataset contoh.....	174
2. Tool audit (regex + scoring).....	174
OPSI C — AI IDS Sederhana (Anomaly Detection dari Log).....	176
1. Buat dataset log sederhana (contoh realistis).....	176
2. Anomaly detection dengan IsolationForest.....	176
Tahap 3 — Output Wajib: Tool + Laporan + Demo.....	178
1. Tool (Wajib).....	178
2. Laporan (Template singkat tapi kuat).....	178

3. Demo (Wajib).....	178
Rubrik Penilaian.....	178
Checklist Final (Sebelum Submit).....	179
BAB 14: Review, Etika & Masa Depan.....	180
Teori 1 — Ethics of AI Security.....	180
Teori 2 — AI Abuse & Governance.....	180
Teori 3 — Masa Depan Profesi Security.....	181
Review & Refleksi (Bagian Fun & Menantang).....	182
Penutup Inspiratif.....	182
PRAKTEK 14: Evaluasi & Etika.....	183
Tujuan.....	183
Aktivitas Utama (Flow Praktikum).....	183
Tahap 1 — Freeze Versi Proyek (Supaya Hasil Evaluasi Konsisten).....	183
Tahap 2 — Uji Kualitas Model/Detektor (Metrics yang “security-minded”).....	184
Tahap 3 — Uji Robustness (Input “Nakal” ala Attacker).....	185
1. Untuk phishing detector.....	185
2. Untuk audit PDP / PII detector.....	185
3. Untuk IDS log detector.....	185
Tahap 4 — Uji Privasi & Data Handling (PDP-friendly).....	185
Contoh: Masking PII sebelum masuk log.....	186
Contoh: Enkripsi laporan/dataset dengan GnuPG (Open Source).....	186
Tahap 5 — Etika AI & Keamanan (Diskusi yang “Real”, bukan normatif).....	186
5 Studi Kasus yang Cocok untuk Diskusi Kelas.....	186
Output Akhir: Template “Refleksi Tertulis”.....	187
1. Ringkasan Proyek (1 paragraf).....	187
2. Dataset & Metodologi Evaluasi.....	187
3. Hasil Evaluasi (wajib angka).....	187
4. Uji Robustness & Temuan.....	187
5. Privasi & Keamanan Data.....	187
6. Risiko Etika & Mitigasi.....	187
7. Rencana Perbaikan (Next steps).....	188
Checklist Penilaian (Rubrik Sederhana).....	188
Mini-Challenge (Supaya Menantang).....	188
Penutup.....	189
LAMPIRAN: Membuat Data Training Set.....	190
Source Code.....	190
Isi Dataset (di dalam ZIP).....	199
Skenario Latihan (Lengkap & Siap Dipakai).....	199
Skenario A — “Kebocoran Data Portal UMKM”.....	199
Skenario B — “Deteksi Phishing Berbasis ML”.....	200
Skenario C — “Audit PII di Tiket Helpdesk”.....	201
Skenario D — “Eksperimen Data Poisoning”.....	201
Skenario E — “Differential Privacy untuk Laporan Statistik”.....	202
Mapping Cepat: Dataset → Pertemuan.....	202

Glossary.....	203
Referensi.....	207
Tentang Penulis.....	208

Kata Pengantar

Alhamdulillah, puji dan syukur penulis panjatkan ke hadirat Allah SWT atas rahmat dan karunia-Nya sehingga buku **Keamanan Informasi & AI** ini dapat disusun dan diselesaikan. Buku ini disiapkan sebagai **buku ajar dan referensi utama** untuk mendukung proses pembelajaran mata kuliah **Keamanan Informasi dan Keamanan AI** yang digunakan dalam kegiatan perkuliahan di **Institut Teknologi Tangerang Selatan (ITTS)**. Materi dalam buku ini dirancang untuk menjembatani konsep fundamental keamanan informasi dengan tantangan nyata yang muncul akibat pesatnya adopsi kecerdasan buatan, sehingga mahasiswa tidak hanya memahami teori, tetapi juga mampu menerapkannya secara praktis, kritis, dan bertanggung jawab.

Penulis menyampaikan **terima kasih** kepada seluruh civitas akademika ITTS—pimpinan institusi, rekan dosen, mahasiswa, serta komunitas pembelajar—atlas dukungan, diskusi, masukan, dan dinamika kelas yang sangat berkontribusi dalam penyempurnaan materi buku ini. Apresiasi juga disampaikan kepada komunitas open-source dan para peneliti yang karyanya menjadi landasan penting dalam pengembangan pendekatan praktikum dan studi kasus yang digunakan. Buku ini disusun dengan pendekatan **Outcome-Based Education (OBE)**, mengutamakan keseimbangan antara teori dan praktik, serta menekankan etika, profesionalisme, dan kesadaran risiko sebagai bagian integral dari kompetensi keamanan.

Penulis menyadari bahwa buku ini masih memiliki keterbatasan dan terbuka untuk terus disempurnakan seiring perkembangan teknologi, regulasi, dan praktik keamanan. Oleh karena itu, kritik dan saran yang konstruktif sangat diharapkan. Semoga buku ini dapat memberikan manfaat nyata bagi mahasiswa, dosen, dan praktisi, serta menjadi kontribusi kecil dalam membangun ekosistem pembelajaran keamanan informasi dan keamanan AI yang relevan, aplikatif, dan berkelanjutan.

Jakarta, Januari 2026

Penulis

PENDAHULUAN

Transformasi digital yang berlangsung secara masif telah mengubah cara informasi diciptakan, diproses, dan dilindungi. Ketergantungan yang semakin tinggi terhadap sistem berbasis data dan *artificial intelligence (AI)* memperluas ruang lingkup keamanan informasi secara signifikan. Keamanan tidak lagi terbatas pada jaringan dan basis data konvensional, tetapi juga mencakup **perlindungan data pribadi, model pembelajaran mesin, sistem pengambilan keputusan otomatis, serta integritas dan keandalan algoritma cerdas.**

Mata kuliah dan buku referensi **Keamanan Informasi dan Keamanan AI** dirancang untuk memberikan landasan konseptual yang kuat sekaligus keterampilan praktis yang relevan dengan kebutuhan dunia akademik dan industri. Materi mengintegrasikan prinsip-prinsip klasik keamanan informasi dengan tantangan baru yang muncul akibat adopsi AI, sehingga keamanan dipahami tidak hanya sebagai mekanisme pertahanan, tetapi juga sebagai bagian strategis dari desain sistem, tata kelola teknologi, dan tanggung jawab profesional.

Mata kuliah ini berbobot **3 SKS** dan diselenggarakan selama **14 pertemuan**, dilengkapi dengan **Ujian Tengah Semester (UTS)** dan **Ujian Akhir Semester (UAS)**. Struktur pembelajaran dirancang dengan komposisi sekitar **40% teori** dan **60% praktik serta proyek**, sejalan dengan pendekatan **Outcome-Based Education (OBE)**. Dalam kerangka OBE, keberhasilan pembelajaran diukur melalui capaian kompetensi yang terukur dan dapat dievaluasi, bukan semata-mata berdasarkan cakupan materi.

Mata kuliah ini ditujukan bagi mahasiswa yang telah memiliki dasar **Teknologi Informasi, Sistem Informasi, atau pemrograman dasar**, sehingga proses pembelajaran dapat langsung difokuskan pada analisis, eksperimen, dan pemecahan masalah keamanan secara aplikatif.

Capaian Pembelajaran Mata Kuliah (CPMK)

Berdasarkan prinsip OBE, capaian pembelajaran dirumuskan dalam bentuk kemampuan yang dapat diamati dan dinilai. Setelah menyelesaikan mata kuliah ini, mahasiswa diharapkan mampu:

1. **Menjelaskan dan menerapkan prinsip keamanan informasi modern**, termasuk kerahasiaan, integritas, ketersediaan, dan akuntabilitas, pada sistem digital.
2. **Mengidentifikasi, mengklasifikasikan, dan melindungi data pribadi serta data sensitif**, dengan mempertimbangkan aspek teknis dan regulasi yang berlaku.
3. **Mendeteksi dan mengaudit potensi kebocoran data dan kelemahan keamanan** menggunakan pendekatan yang sistematis dan berbasis bukti.
4. **Menganalisis ancaman terhadap sistem berbasis AI**, termasuk *adversarial attack*, *data poisoning*, dan kebocoran model.
5. **Memanfaatkan AI sebagai alat bantu keamanan**, misalnya untuk deteksi anomali, analitik keamanan, dan audit data otomatis.
6. **Melakukan audit keamanan berbasis AI** serta menafsirkan hasilnya secara kritis dan bertanggung jawab.

7. **Merancang solusi keamanan yang praktis, etis, dan berkelanjutan**, dengan memperhatikan aspek teknis, profesional, dan dampak sosial.

Capaian pembelajaran ini dipetakan secara langsung ke aktivitas pembelajaran, praktikum, proyek, dan metode evaluasi untuk memastikan keterpaduan antara **tujuan pembelajaran, proses pembelajaran, dan penilaian**, sebagaimana dituntut dalam kerangka OBE.

Perangkat Pembelajaran dan Lingkungan Teknis

Untuk mendukung pembelajaran yang berorientasi pada capaian dan pengalaman nyata, mata kuliah ini memanfaatkan berbagai **perangkat lunak open-source** yang umum digunakan dalam riset dan praktik profesional, antara lain:

- **Python** dan **Jupyter Notebook** sebagai lingkungan utama untuk eksperimen, analisis, dan pengembangan prototipe
- Pustaka *machine learning* dasar seperti **Scikit-learn**, serta pengantar **PyTorch** atau **TensorFlow**
- Perangkat analisis jaringan dan deteksi intrusi seperti **Wireshark** dan **Snort / Suricata**
- Berbagai **alat OSINT** (misalnya *Have I Been Pwned* dan *Maltego*) yang digunakan dalam skenario simulasi terkontrol dan etis
- **Dataset publik** dari repositori terbuka serta *synthetic data* untuk kebutuhan eksperimen keamanan
- Penggunaan **Linux Virtual Machine** atau **Docker** (opsional) untuk menjamin isolasi, reproduisibilitas, dan keamanan lingkungan praktik

Perspektif Pendidikan

Dengan mengkombinasikan teori fundamental, praktik keamanan, dan pendekatan berbasis AI, buku ini bertujuan membentuk **kompetensi teknis, kemampuan berpikir kritis**, serta **kesadaran etika dan profesional** mahasiswa. Keamanan diposisikan sebagai disiplin yang dinamis dan multidisipliner, yang harus terus beradaptasi dengan perkembangan teknologi, pola ancaman, dan tuntutan masyarakat.

Buku referensi ini ditujukan bagi mahasiswa sarjana dan pascasarjana awal, dosen, serta praktisi yang membutuhkan kerangka pembelajaran dan referensi yang sistematis, aplikatif, dan **berorientasi pada capaian pembelajaran** dalam bidang **keamanan informasi dan keamanan AI**.

BAB 1: Fondasi Keamanan Informasi

Pada awal perkembangan teknologi informasi, fokus utama manusia adalah **bagaimana membangun sistem agar dapat berfungsi dengan baik**. Namun seiring meningkatnya kompleksitas dan ketergantungan terhadap teknologi, muncul pertanyaan yang jauh lebih krusial: “**apa yang terjadi jika sistem tersebut gagal?**”. Pergeseran cara berpikir inilah yang melahirkan disiplin **keamanan informasi**.

Di era digital modern, **hampir seluruh aktivitas manusia menghasilkan dan bergantung pada data**. Mahasiswa melakukan pendaftaran kuliah secara daring, dosen mengelola nilai melalui sistem akademik, rumah sakit menyimpan rekam medis elektronik, dan perusahaan mengambil keputusan strategis berbasis analitik data serta *kecerdasan buatan*. Data tidak lagi sekadar catatan administratif, melainkan telah berubah menjadi **aset strategis bernilai tinggi**.

Semakin bernilai suatu data, semakin besar pula risiko yang menyertainya.

Keamanan Informasi Bukan Lagi Pilihan

Secara umum, **keamanan informasi** merupakan **upaya yang terencana dan sistematis** untuk melindungi informasi dari:

- **akses yang tidak sah,**
- **perubahan atau manipulasi yang tidak diinginkan,** serta
- **gangguan terhadap ketersediaan informasi.**

Tujuan utama keamanan informasi **bukanlah menciptakan sistem yang sepenuhnya kebal terhadap serangan**, melainkan **mengelola risiko** sehingga dampak dari ancaman dapat ditekan ke tingkat yang masih dapat diterima oleh organisasi.

Dalam praktiknya, keamanan informasi selalu berangkat dari **tiga pertanyaan mendasar**:

1. **Siapa yang berhak mengakses informasi?**
2. **Apakah informasi tersebut tetap benar dan utuh?**
3. **Apakah informasi tersedia saat dibutuhkan?**

Ketiga pertanyaan ini dirangkum dalam sebuah kerangka konseptual yang dikenal sebagai *CIA Triad*.

Keamanan yang Terlambat Selalu Mahal

Banyak organisasi baru menyadari pentingnya keamanan informasi **setelah insiden terjadi**—setelah data bocor, setelah sistem lumpuh, dan setelah kepercayaan publik runtuh. Pada tahap ini, keamanan seringkali **datang terlambat dan dengan biaya yang jauh lebih besar**.

Kesalahan paling mendasar adalah **menganggap keamanan informasi sebagai fitur tambahan**. Padahal, **keamanan adalah fondasi utama**. Sistem yang canggih namun tidak aman ibarat **gedung megah tanpa pondasi**: tampak kuat di permukaan, tetapi sangat rapuh ketika menghadapi tekanan nyata.

Lebih jauh lagi, **keamanan informasi tidak hanya melindungi komputer atau jaringan**, tetapi juga:

- **identitas manusia,**
- **keputusan dan keberlangsungan organisasi,**
- **stabilitas sosial,** serta
- **kepercayaan publik.**

Mengapa CIA Triad Menjadi Fondasi Keamanan Informasi

Di era digital saat ini, **data telah menjadi aset paling berharga** bagi individu, organisasi, maupun negara. Data akademik mahasiswa, data pelanggan UMKM, data kesehatan, hingga data biometrik kini disimpan dan diproses dalam sistem informasi yang saling terhubung melalui jaringan global. **Ketergantungan yang tinggi terhadap sistem digital secara langsung meningkatkan risiko keamanan informasi.**

Karena itu, keamanan informasi **bukan semata persoalan teknis** seperti firewall atau antivirus. Ia merupakan **isu strategis** yang menyentuh aspek kepercayaan, keberlangsungan layanan, pengambilan keputusan, hingga stabilitas sosial.

Untuk memahami **apa yang sebenarnya harus dilindungi**, dibutuhkan sebuah kerangka dasar yang **sederhana, jelas, dan kuat**. Kerangka inilah yang dikenal sebagai *CIA Triad*—fondasi konseptual dalam keamanan informasi yang menjadi titik awal bagi seluruh kebijakan, teknologi, dan praktik keamanan.

CIA Triad sebagai Fondasi Keamanan Informasi

Dalam menghadapi kompleksitas keamanan informasi, diperlukan sebuah **kerangka dasar yang sederhana namun komprehensif** untuk memahami apa yang sebenarnya harus dilindungi. Kerangka tersebut dikenal sebagai **CIA Triad**, sebuah model konseptual yang telah lama digunakan dan **tetap relevan hingga saat ini** dalam praktik profesional keamanan informasi.

Istilah **CIA** dalam konteks ini **bukan merujuk pada lembaga intelijen**, melainkan singkatan dari tiga prinsip fundamental keamanan informasi:

- *Confidentiality* (**Kerahasiaan**)
- *Integrity* (**Integritas atau Keutuhan Data**)
- *Availability* (**Ketersediaan**)

Ketiga prinsip ini membentuk **fondasi utama** dalam perancangan, pengelolaan, dan evaluasi sistem informasi yang aman. **Hampir seluruh insiden keamanan informasi** dapat ditelusuri sebagai kegagalan pada **satu atau lebih** dari ketiga aspek tersebut.

Secara konseptual, **CIA Triad digunakan sebagai kerangka berpikir** dalam berbagai aktivitas keamanan informasi, antara lain:

- **Perancangan sistem dan arsitektur keamanan,**
- **Analisis serta manajemen risiko,**
- **Audit dan evaluasi keamanan informasi,**
- **Investigasi insiden keamanan,**
- **Penyusunan kebijakan, standar, dan regulasi keamanan.**

Pada dasarnya, setiap bentuk **kontrol keamanan**—baik yang bersifat **teknis** (seperti enkripsi dan otentikasi), **administratif** (kebijakan dan prosedur), maupun **prosedural** (alur kerja dan tata kelola)—dirancang untuk **melindungi satu atau lebih pilar dalam CIA Triad**.

CIA Triad juga berfungsi sebagai **alat analisis insiden keamanan**, misalnya:

- **Kebocoran data** menunjukkan kegagalan pada *confidentiality*,
- **Perubahan atau manipulasi data tanpa izin** mencerminkan pelanggaran *integrity*,
- **Gangguan layanan atau sistem tidak dapat diakses** merupakan masalah pada *availability*.

Dalam banyak kasus, **satu insiden dapat melanggar lebih dari satu aspek CIA secara bersamaan**. Contohnya adalah serangan *ransomware*, yang dapat menyebabkan hilangnya ketersediaan sistem, merusak integritas data, serta berpotensi mengancam kerahasiaan informasi.

Dengan demikian, **CIA Triad menyediakan cara pandang yang terstruktur dan mudah dipahami** untuk menjelaskan keamanan informasi. Kerangka ini membantu menjawab pertanyaan mendasar: **apa yang dilindungi, mengapa perlindungan itu penting, dan bagaimana kegagalan keamanan dapat terjadi**, sehingga menjadi **titik awal yang esensial** dalam membangun sistem informasi yang aman, andal, dan berkelanjutan.

Confidentiality (Kerahasiaan)

Definisi Confidentiality

Confidentiality adalah **prinsip keamanan informasi** yang menjamin bahwa **informasi hanya dapat diakses oleh pihak yang berwenang**. Setiap data harus terlindungi agar **tidak bocor, tidak terlihat, dan tidak disalin** oleh pihak yang tidak memiliki hak akses.

Prinsip ini menjadi **sangat krusial** ketika sistem mengelola **data pribadi, rahasia bisnis, atau informasi strategis organisasi**. Pada dasarnya, **tidak semua orang berhak mengetahui seluruh informasi**—ini adalah inti dari *confidentiality*.

Mengapa Confidentiality Penting

Berbagai jenis data memiliki **tingkat kerahasiaan yang berbeda**, seperti:

- **data pribadi** (NIK, alamat, nomor telepon),
- **nilai akademik dan data mahasiswa**,
- **informasi gaji dan kepegawaian**,
- **rekam medis pasien**,
- hingga **dataset pelatihan AI**.

Ketika data-data tersebut **diakses oleh pihak yang tidak berwenang**, maka **prinsip confidentiality telah dilanggar**.

Menariknya, **kebocoran data modern jarang disebabkan oleh serangan teknis yang sangat canggih**. Sebaliknya, sebagian besar insiden justru terjadi karena **kesalahan dasar dan kelalaian manusia**, antara lain:

- **database yang tidak dilindungi autentikasi**,
- **penyimpanan cloud yang dibiarkan bersifat publik**,
- **kredensial yang dibagikan atau digunakan ulang secara sembarangan**,
- **tautan internal yang tidak sengaja tersebar ke publik**.

Dari sini dapat ditarik satu kesimpulan penting:

keamanan informasi sering gagal bukan karena serangan, melainkan karena kelalaian.

Contoh Pelanggaran Confidentiality

Beberapa contoh **pelanggaran confidentiality** yang sering ditemukan dalam praktik antara lain:

- database berisi NIK dan nomor telepon dapat diakses tanpa autentikasi,
- file *backup* disimpan dalam bentuk *plain text*,
- email *phishing* yang berhasil mencuri kredensial pengguna,
- akun internal masih dapat diakses oleh mantan pegawai karena hak akses tidak dicabut.

Pelanggaran kerahasiaan **sering menjadi pintu masuk bagi insiden yang lebih serius**, seperti **penipuan**, **pemerasan**, dan **pencurian identitas**.

Mekanisme Perlindungan Confidentiality

Perlindungan *confidentiality* dilakukan melalui kombinasi **kontrol teknis**, **administratif**, dan **prosedural**, di antaranya:

- enkripsi data, baik saat *data at rest* maupun *data in transit*,
- autentikasi pengguna yang kuat,
- kontrol akses berbasis peran (*role-based access control*),
- klasifikasi data berdasarkan tingkat sensitivitas,
- kebijakan penggunaan, penyimpanan, dan pembagian data.

Dalam konteks **perlindungan data pribadi**, kegagalan menjaga *confidentiality* tidak hanya berdampak teknis, tetapi juga dapat menimbulkan **konsekuensi hukum dan etika yang serius** bagi organisasi maupun individu yang bertanggung jawab.

Integrity (Keutuhan Data)

Definisi Integrity

Integrity adalah **prinsip keamanan informasi** yang menjamin bahwa **data tetap utuh, akurat, dan konsisten**, serta **tidak mengalami perubahan tanpa izin**. Informasi yang terlindungi dengan baik tetapi **tidak lagi benar** tetap berbahaya, karena dapat **menghasilkan keputusan yang keliru dan merugikan**.

Prinsip *integrity* **tidak hanya berkaitan dengan serangan teknis**, tetapi juga mencakup **kesalahan manusia, kegagalan proses, dan kelemahan tata kelola data**. Jika *confidentiality* menjawab pertanyaan “**siapa yang berhak mengakses data?**”, maka *integrity* menjawab pertanyaan yang tidak kalah penting: “**apakah data ini masih benar dan dapat dipercaya?**”.

Mengapa Integritas Sangat Penting

Data yang bocor memang berbahaya, namun **data yang diam-diam diubah tanpa terdeteksi sering kali jauh lebih merusak**. Perubahan yang tidak sah dapat terjadi tanpa disadari dan berdampak langsung pada keputusan kritis.

Contohnya antara lain:

- **nilai mahasiswa yang diubah tanpa jejak,**
- **laporan keuangan yang dimanipulasi,**
- **log keamanan yang dihapus untuk menutupi serangan,**
- **atau dataset pelatihan AI yang sengaja dirusak.**

Dalam konteks **kecerdasan buatan**, *integrity* menjadi **isu yang semakin krusial**. Sistem AI **belajar dari data**, sehingga **kualitas dan keutuhan data secara langsung menentukan kualitas keputusan yang dihasilkan**. Jika data bersifat **rusak, bias, atau dimanipulasi**, maka sistem AI akan **mewarisi dan memperbesar kesalahan tersebut dalam skala besar**.

Dengan kata lain, **AI tidak memperbaiki kesalahan—AI justru mengamplifikasinya**.

Contoh Pelanggaran Integrity

Beberapa contoh **pelanggaran *integrity*** yang sering terjadi dalam sistem informasi meliputi:

- nilai akademik diubah tanpa otorisasi,
- transaksi keuangan dimodifikasi,
- log keamanan dihapus atau diubah untuk menghilangkan jejak,
- dataset pelatihan AI dimanipulasi sehingga menghasilkan model yang bias atau menyesatkan.

Dalam sistem berbasis AI, **pelanggaran *integrity* pada data memiliki dampak yang jauh lebih luas**, karena kesalahan tersebut direplikasi dan disebarakan secara otomatis oleh model.

Mekanisme Perlindungan Integrity

Untuk menjaga *integrity*, digunakan berbagai **mekanisme pengamanan**, antara lain:

- **hash dan *checksum*** untuk mendeteksi perubahan data,
- **tanda tangan digital (*digital signature*)** untuk menjamin keaslian dan keutuhan,
- **audit trail** untuk mencatat setiap perubahan data,
- **kontrol versi (*version control*)**,
- **validasi input dan output**,
- serta **monitoring perubahan data secara berkelanjutan**.

Secara keseluruhan, ***integrity* merupakan fondasi kepercayaan terhadap sistem informasi**. Tanpa keutuhan data yang terjaga, **hasil analisis, keputusan organisasi, dan keluaran sistem—termasuk sistem AI—tidak dapat dipercaya**.

Availability (Ketersediaan)

Definisi Availability

Availability adalah prinsip keamanan informasi yang menjamin bahwa **sistem, layanan, dan data selalu tersedia dan dapat diakses** oleh **pengguna yang sah** pada saat dibutuhkan. **Sistem yang aman secara teknis tetapi sering tidak dapat diakses tetap dianggap gagal**, karena tidak mampu memenuhi tujuan utamanya.

Prinsip *availability* berkaitan erat dengan **keandalan sistem, ketahanan terhadap gangguan**, serta **kemampuan layanan untuk terus beroperasi dalam berbagai kondisi**.

Makna Availability dalam Keamanan Informasi

Keamanan sering kali **disalah artikan sebagai upaya mengunci sistem seketat mungkin**. Padahal, **sistem yang terlalu tertutup hingga sulit digunakan bukanlah sistem yang aman**. *Availability* menegaskan bahwa **keamanan harus tetap memungkinkan sistem berfungsi sebagaimana mestinya**.

Secara praktis, *availability* memastikan bahwa:

- **layanan dapat diakses saat dibutuhkan,**
- **data tidak hilang atau tidak dapat digunakan,**
- **sistem tetap berjalan meskipun terjadi gangguan atau serangan.**

Banyak serangan keamanan modern **tidak berfokus pada pencurian data**, melainkan pada **melumpuhkan layanan**. Serangan seperti *ransomware* dan *Distributed Denial of Service (DDoS)* secara langsung menargetkan *availability* dengan membuat sistem **tidak dapat digunakan**.

Dalam konteks **layanan publik**, kegagalan *availability* dapat berdampak langsung pada kehidupan masyarakat, seperti:

- **pendaftaran layanan yang gagal,**
- **terhentinya layanan kesehatan,**
- **terganggunya distribusi bantuan sosial.**

Oleh karena itu, **keamanan sejati bukan sekadar membuat sistem tertutup**, melainkan **memastikan sistem tetap berjalan bahkan dalam kondisi terburuk**.

Contoh Gangguan Availability

Beberapa contoh **gangguan availability** yang sering terjadi antara lain:

- serangan ***Distributed Denial of Service (DDoS)***,
- **server down** pada periode kritis (misalnya masa KRS atau penggajian),
- serangan ***ransomware*** yang mengenkripsi sistem,
- kegagalan listrik atau perangkat keras tanpa mekanisme ***backup***.

Gangguan terhadap **availability** tidak hanya menimbulkan kerugian finansial, tetapi juga **merusak reputasi organisasi dan menurunkan kepercayaan publik**.

Mekanisme Perlindungan Availability

Untuk menjaga **availability**, organisasi perlu menerapkan kombinasi **pendekatan teknis dan manajerial**, antara lain:

- mekanisme ***backup dan recovery***,
- **redundansi sistem dan infrastruktur**,
- **failover dan *high availability***,
- **monitoring dan *alerting* secara real-time**,
- **rencana pemulihan bencana (*disaster recovery plan*)**.

Availability menuntut **perencanaan yang matang**, karena kegagalan sering kali terjadi bukan akibat satu masalah, melainkan **rantai gangguan yang saling berkaitan**.

Hubungan dan Trade-off dalam CIA Triad

Ketiga elemen dalam **CIA Triad** tidak selalu dapat dimaksimalkan secara bersamaan. Dalam praktik nyata, sering terjadi **trade-off**, misalnya:

- **enkripsi yang sangat kuat** dapat menurunkan performa sistem,
- **kontrol akses yang terlalu ketat** dapat mengurangi kemudahan penggunaan,
- **redundansi sistem** meningkatkan **availability** tetapi menambah kompleksitas dan biaya.

Oleh karena itu, **keamanan informasi** bukanlah soal menciptakan sistem yang “**paling aman**”, melainkan **sistem yang paling sesuai dengan risiko, kebutuhan, dan konteks penggunaannya**.

Ancaman Modern: Dunia Digital yang Semakin Kompleks

Pada masa awal, **sistem informasi cenderung tertutup dan terpusat**. Namun, sistem modern kini berkembang menjadi lingkungan yang:

- **terdistribusi,**
- **berbasis *cloud*,**
- **terhubung melalui *API*,**
- **serta semakin bergantung pada *kecerdasan buatan*.**

Perubahan ini secara langsung **meningkatkan kompleksitas sistem** dan menciptakan **permukaan serangan yang jauh lebih luas**. *Cloud* memang menawarkan kemudahan dan skalabilitas, tetapi sering kali menghadirkan **ilusi keamanan**. *AI* memberikan kemampuan cerdas, namun sekaligus **membuka vektor serangan baru**. Sementara itu, *data economy* menjadikan data sebagai komoditas bernilai tinggi, sering kali **tanpa mempertimbangkan perlindungan jangka panjang**.

Ancaman modern **jarang muncul sebagai satu serangan besar yang terlihat jelas**. Sebaliknya, ia sering terjadi sebagai **akumulasi dari banyak keputusan kecil yang keliru**.

Mengapa Keamanan Informasi Terus Mengalami Kegagalan

Meskipun **prinsip CIA Triad telah lama dikenal**, kegagalan keamanan informasi **terus berulang**. Penyebab utamanya sering kali **bukan persoalan teknis semata**, melainkan **masalah manusia dan organisasi**.

Keamanan informasi gagal karena:

- **manusia lebih mengutamakan kecepatan dan kenyamanan dibanding kehati-hatian,**
- **organisasi mengorbankan keamanan demi efisiensi jangka pendek,**
- **sistem dibangun tanpa pemahaman risiko yang memadai,**
- **teknologi dipercaya secara berlebihan tanpa memahami batas dan konsekuensinya.**

Banyak organisasi **menginvestasikan dana besar untuk alat keamanan**, tetapi **tidak memahami apa yang sebenarnya mereka lindungi**. Akibatnya, keamanan berubah menjadi **simbol formalitas**, bukan **praktik yang hidup dalam operasional sehari-hari**.

Keamanan sebagai Proses Berpikir Berkelanjutan

Keamanan informasi bukanlah tujuan akhir, melainkan **proses berpikir yang terus berlangsung**. Mempelajari keamanan informasi **bukan berarti menjadi paranoid**, tetapi **menjadi sadar terhadap risiko dan dampak kegagalan**.

Mahasiswa dan praktisi keamanan belajar untuk:

- **melihat sistem tidak hanya dari sisi fungsional,**
- **tetapi juga dari sisi dampak ketika sistem gagal.**

Prinsip paling mendasar dalam keamanan informasi adalah kesadaran bahwa:

- **tidak ada sistem yang sepenuhnya aman,**
- **namun ada sistem yang dikelola dengan tanggung jawab dan kesadaran risiko.**

Membangun Mentalitas Keamanan

Bagian ini **tidak bertujuan mengajarkan alat, konfigurasi, atau kode teknis**, melainkan **membentuk cara berpikir dan mentalitas keamanan**. Sebelum membangun atau menerapkan sistem, pertanyaan-pertanyaan mendasar harus diajukan:

- **Data apa yang paling bernilai dan harus dilindungi?**
- **Siapa yang akan terdampak jika sistem gagal?**
- **Asumsi keamanan apa yang selama ini dianggap “pasti benar” tanpa diuji?**

Jawaban atas pertanyaan-pertanyaan tersebut akan menentukan apakah sebuah sistem sekadar berfungsi, atau benar-benar layak dipercaya.

Bab selanjutnya akan membawa pembaca dari **kesadaran konseptual ini** menuju langkah yang lebih terstruktur, yaitu **manajemen risiko dan *threat modeling***, agar keamanan informasi **tidak berhenti pada niat baik**, tetapi berkembang menjadi **praktik nyata yang dapat dipertanggungjawabkan**.

PRAKTEK 1: Identifikasi Aset & Ancaman

Tujuan Pembelajaran

Setelah mengikuti modul ini, mahasiswa **mampu**:

- **Mengidentifikasi aset informasi dan aset sistem** pada sebuah sistem nyata
- **Mengidentifikasi ancaman keamanan** terhadap aset tersebut
- **Menganalisis dampak keamanan** berdasarkan *CIA Triad*
- **Menyajikan hasil analisis** dalam bentuk diagram dan tabel yang terstruktur

Fokus utama modul ini adalah **membangun cara berpikir keamanan**, bukan konfigurasi teknis.

Konsep Inti yang Digunakan

1. **CIA Triad**
 - *Confidentiality* (**Kerahasiaan**)
 - *Integrity* (**Keutuhan**)
 - *Availability* (**Ketersediaan**)
2. **Asset – Threat – Impact**
 - **Asset**: apa yang bernilai dan harus dilindungi
 - **Threat**: apa yang dapat merusak aset
 - **Impact**: apa akibatnya jika ancaman terjadi

Tools yang Digunakan

Mahasiswa **tidak membutuhkan tools teknis khusus**. Gunakan:

- **Kertas / Whiteboard** (diskusi manual)
- **draw.io / Excalidraw** → membuat diagram aset & ancaman
- **Spreadsheet (Excel / Google Sheets / LibreOffice)** → tabel analisis CIA

Langkah Praktikum (Step-by-Step)

Langkah 1 — Memilih Sistem Nyata

Mahasiswa memilih **satu sistem yang familiar**. Contoh:

- **Sistem Akademik Kampus**
- **WhatsApp**
- **Aplikasi E-Commerce**
- **Sistem Absensi Online**
- **Website UMKM**

Catatan: Sistem tidak harus kompleks—yang penting **nyata dan digunakan sehari-hari**.

Contoh pilihan sistem (digunakan di modul ini):

Sistem Akademik Kampus (SIKAD)

Langkah 2 — Identifikasi Aset

Mahasiswa mengidentifikasi **semua aset yang bernilai**, dibagi menjadi tiga kategori:

A. Aset Data

Contoh pada SIKAD:

- Data mahasiswa (NIM, nama, alamat)
- Nilai akademik
- KRS & KHS
- Data dosen
- Password pengguna
- Log aktivitas sistem

B. Aset Sistem

- Server aplikasi
- Database server
- Website / portal akademik
- Jaringan kampus
- Backup data

C. Aset Pengguna

- Mahasiswa
- Dosen
- Admin akademik
- Operator IT

Prinsip penting:

Jika aset hilang, rusak, atau bocor → ada dampak.

Langkah 3 — Identifikasi Ancaman

Untuk setiap aset, mahasiswa mengidentifikasi **ancaman yang mungkin terjadi**, baik **sengaja maupun tidak sengaja**.

Contoh ancaman umum:

- Human error
- Password lemah
- Akun tidak dicabut
- Salah konfigurasi server
- Serangan *phishing*
- *Ransomware*
- Server down

Langkah 4 — Pemetaan Ancaman ke CIA Triad

Setiap ancaman dipetakan ke **aspek CIA** yang terdampak.

Contoh Tabel CIA Impact

Aset	Ancaman	Confidentiality	Integrity	Availability	Dampak
Data nilai mahasiswa	Akun admin bocor	✗	✗	✓	Nilai dapat diubah
Database akademik	Server down	✓	✓	✗	KRS tidak bisa diakses
Password pengguna	Phishing	✗	✗	✗	Akses ilegal
Sistem akademik	Ransomware	✗	✗	✗	Sistem lumpuh

✓ = aman

✗ = terganggu

Catatan penting:

Satu ancaman bisa merusak lebih dari satu aspek CIA sekaligus.

Langkah 5 — Analisis Dampak

Mahasiswa mendiskusikan:

- **Siapa yang terdampak?**
- **Seberapa serius dampaknya?**
- **Apakah dampaknya teknis, akademik, finansial, atau reputasi?**

Contoh dampak:

- Mahasiswa tidak bisa KRS
- Nilai salah → kelulusan tertunda
- Hilangnya kepercayaan terhadap sistem
- Potensi pelanggaran regulasi data pribadi

Output yang Harus Dikumpulkan

1. Diagram Aset & Ancaman

Menggunakan *draw.io* / *Excalidraw*:

- Aset data
- Aset sistem
- Ancaman utama
- Hubungan antar komponen

Contoh isi diagram:

```
Mahasiswa → Login → SIAKAD → Database Nilai
                        ↑
                Phishing / Ransomware
```

2. Tabel Analisis CIA Impact

Minimal berisi:

- Aset
- Ancaman
- Dampak ke *Confidentiality*
- Dampak ke *Integrity*
- Dampak ke *Availability*

Evaluasi Penilaian

Penilaian **bukan berdasarkan benar-salah teknis**, tetapi pada **kualitas berpikir keamanan**:

Aspek	Bobot
Ketepatan identifikasi aset	30%
Kelengkapan ancaman	30%
Ketepatan pemetaan CIA	30%
Kejelasan presentasi	10%

Refleksi Penutup Modul

Mahasiswa diharapkan menyadari bahwa:

- **Keamanan dimulai dari memahami apa yang bernilai**
- **Ancaman tidak selalu canggih—seringkali sederhana**
- **Kesalahan kecil dapat berdampak besar**

Keamanan bukan alat, tetapi cara berpikir

BAB 2: Manajemen Risiko & Threat Modeling

Dari Kesadaran ke Analisis

Pada bab sebelumnya, pembaca diajak untuk memahami **apa yang harus dilindungi** melalui identifikasi aset dan ancaman. Namun, **tidak semua ancaman memiliki tingkat bahaya yang sama**, dan **tidak semua aset memerlukan perlindungan dengan prioritas yang setara**. Di sinilah **manajemen risiko** dan *threat modeling* berperan.

Bab ini menekankan bahwa **keamanan informasi bukan tentang menambahkan sebanyak mungkin alat**, melainkan tentang **kemampuan menganalisis risiko secara rasional dan terstruktur**. Fokus utama bab ini adalah **cara berpikir analitis**, bukan konfigurasi teknis atau penggunaan perangkat keamanan tertentu.

Konsep Risiko dalam Keamanan Informasi

Dalam konteks keamanan informasi, **risiko** tidak muncul secara kebetulan. Risiko selalu merupakan hasil dari **interaksi antara nilai aset, ancaman, dan kelemahan sistem**. Secara konseptual, risiko sering dirumuskan sebagai:

$$\text{Risk} = \text{Asset} \times \text{Threat} \times \text{Vulnerability}$$

Rumus ini menegaskan bahwa **ancaman yang besar tidak selalu menghasilkan risiko tinggi** jika asetnya tidak bernilai atau sistemnya tidak rentan. Sebaliknya, **aset yang sangat bernilai dengan kerentanan kecil sekalipun dapat menciptakan risiko serius**.

- **Asset** menunjukkan **nilai atau kepentingan informasi dan sistem**
- **Threat** menggambarkan **potensi kejadian yang dapat merugikan aset**
- **Vulnerability** adalah **kelemahan yang memungkinkan ancaman terjadi**

Dengan memahami hubungan ini, mahasiswa belajar bahwa **keamanan bukan soal “aman atau tidak aman”**, melainkan soal **seberapa besar risiko yang bersedia diterima**.

Manajemen Risiko sebagai Proses Berkelanjutan

Manajemen risiko bukanlah aktivitas sekali jalan, melainkan **proses yang terus berulang dan beradaptasi**. Sistem berubah, pengguna berubah, dan ancaman pun berkembang. Oleh karena itu, risiko harus **dinilai, diprioritaskan, dan dievaluasi secara berkala**.

Tujuan utama manajemen risiko **bukan menghilangkan seluruh risiko**, karena hal tersebut tidak realistis. Tujuannya adalah **mengelola risiko agar berada pada tingkat yang dapat diterima**, baik secara teknis, organisasi, maupun sosial.

Pendekatan ini membantu organisasi **mengalokasikan sumber daya secara rasional**, sehingga upaya keamanan difokuskan pada **risiko paling signifikan**, bukan pada ketakutan atau asumsi semata.

Threat Modeling: Memetakan Cara Sistem Bisa Gagal

Jika manajemen risiko menjawab pertanyaan “**risiko mana yang paling penting?**”, maka *threat modeling* menjawab pertanyaan “**bagaimana sistem bisa diserang atau gagal?**”.

Threat modeling adalah **pendekatan sistematis untuk mengidentifikasi, mengklasifikasikan, dan memahami ancaman** berdasarkan cara kerja sistem. Pendekatan ini memaksa analis untuk **melihat sistem dari sudut pandang penyerang maupun kegagalan internal**.

Salah satu kerangka yang sering digunakan adalah *STRIDE*, yang mengelompokkan ancaman ke dalam enam kategori utama: *Spoofing*, *Tampering*, *Repudiation*, *Information Disclosure*, *Denial of Service*, dan *Elevation of Privilege*. Kerangka ini membantu memastikan bahwa **tidak ada jenis ancaman penting yang terlewatkan**.

Selain itu, konsep *attack surface* digunakan untuk memahami **bagian sistem mana saja yang dapat diakses atau dieksploitasi oleh pihak luar**. Semakin luas *attack surface*, semakin besar peluang terjadinya insiden keamanan.

Praktik: Threat Modeling pada Aplikasi Fiktif

Pada bagian praktik, mahasiswa menerapkan konsep ini pada **aplikasi fiktif**, misalnya aplikasi akademik, sistem pemesanan online, atau aplikasi layanan publik. Penggunaan sistem fiktif bertujuan agar mahasiswa **bebas berpikir kritis tanpa terikat pada detail teknis tertentu**.

Mahasiswa diminta untuk:

- memetakan komponen utama sistem,
- mengidentifikasi alur data dan pengguna,
- menentukan kemungkinan ancaman berdasarkan *STRIDE*,
- serta mengaitkannya dengan aset dan kerentanan yang telah dikenali.

Hasil analisis ini kemudian digunakan untuk **menilai tingkat risiko secara manual**, tanpa alat otomatis.

Penilaian Risiko dan Risk Scoring

Untuk memperkuat pemahaman konseptual, penilaian risiko dilakukan secara **manual menggunakan spreadsheet**. Setiap risiko dinilai berdasarkan **tingkat kemungkinan** dan **besarnya dampak**, lalu diberi skor sederhana.

Pendekatan ini mengajarkan bahwa **angka risiko bukan kebenaran mutlak**, melainkan **alat bantu pengambilan keputusan**. Yang terpenting bukanlah presisi matematis, tetapi **konsistensi dan logika penilaian**.

Melalui proses ini, mahasiswa belajar:

- membandingkan risiko satu dengan lainnya,
- menentukan prioritas mitigasi,
- dan memahami konsekuensi dari pilihan keamanan.

Output dan Keterampilan yang Dihasilkan

Hasil akhir dari bab ini bukan konfigurasi sistem, melainkan **dokumen analisis risiko** yang terstruktur. Dokumen ini mencerminkan **kemampuan berpikir keamanan secara profesional**, yang dapat diterapkan pada berbagai konteks sistem.

Keterampilan utama yang dihasilkan meliputi:

- **kemampuan menganalisis risiko secara sistematis,**
- **kemampuan melakukan *threat modeling*,**
- **kemampuan mengkomunikasikan risiko secara tertulis,**
- **serta kemampuan membedakan risiko kritis dan nonkritis.**

Keamanan sebagai Keputusan yang Disadari

Bab ini menegaskan bahwa **keamanan informasi bukan soal alat paling mahal atau teknologi paling mutakhir**, melainkan **keputusan sadar berbasis analisis risiko**. Sistem yang aman adalah sistem yang **memahami risikonya, mengelolanya dengan bijak, dan siap menghadapi kegagalan**.

Bab selanjutnya akan melanjutkan proses ini dengan membahas **strategi mitigasi dan kontrol keamanan**, agar risiko yang telah diidentifikasi **tidak berhenti sebagai analisis**, tetapi diterjemahkan menjadi **tindakan nyata yang terukur**.

STRIDE: Kerangka Klasifikasi Ancaman Keamanan

STRIDE adalah **kerangka *threat modeling*** yang digunakan untuk **mengidentifikasi dan mengelompokkan jenis ancaman keamanan** secara terstruktur. STRIDE membantu analisis keamanan **tidak melewatkan tipe ancaman penting**, terutama pada sistem yang kompleks dan terdistribusi.

STRIDE terdiri dari **enam kategori ancaman utama**, masing-masing merepresentasikan **cara berbeda sebuah sistem dapat diserang atau gagal**.

Tujuan Penggunaan STRIDE

STRIDE digunakan untuk:

- **memetakan ancaman secara sistematis,**
- **melihat sistem dari sudut pandang penyerang,**
- **menghubungkan ancaman dengan CIA Triad,**
- **serta menjadi dasar penilaian risiko dan mitigasi.**

STRIDE **bukan alat teknis**, melainkan **alat berpikir analitis**.

Enam Kategori Ancaman STRIDE

1. *Spoofing* (Pemalsuan Identitas)

Spoofing adalah ancaman di mana **penyerang menyamar sebagai pengguna atau sistem lain** untuk mendapatkan akses ilegal.

Ancaman ini **langsung menyerang aspek *Confidentiality*** dan sering kali *Integrity*.

Contoh:

- Login menggunakan akun orang lain
- Pemalsuan identitas admin
- *Session hijacking*

Pertanyaan analisis:

Apakah identitas pengguna dan sistem dapat dipalsukan?

2. *Tampering* (Perubahan Data Tanpa Izin)

Tampering terjadi ketika **data diubah, dimodifikasi, atau dirusak tanpa otorisasi**.

Ancaman ini **berkaitan langsung dengan *Integrity***.

Contoh:

- Nilai mahasiswa diubah
- Transaksi keuangan dimanipulasi
- File konfigurasi diubah

Pertanyaan analisis:

Apakah data dapat diubah tanpa terdeteksi?

3. *Repudiation* (Penyangkalan Aktivitas)

Repudiation terjadi ketika **pengguna dapat menyangkal telah melakukan suatu tindakan karena tidak adanya bukti yang kuat**.

Ancaman ini menyerang **kepercayaan sistem dan auditabilitas**.

Contoh:

- Pengguna mengklaim tidak pernah melakukan transaksi
- Tidak ada *audit log* yang valid

- Log dapat dihapus atau dimodifikasi

Pertanyaan analisis:

Apakah sistem memiliki bukti yang dapat dipertanggungjawabkan?

4. *Information Disclosure* (Kebocoran Informasi)

Information Disclosure adalah ancaman di mana **informasi sensitif bocor ke pihak yang tidak berwenang**.

Ancaman ini **langsung melanggar Confidentiality**.

Contoh:

- Database terbuka tanpa autentikasi
- File *backup* dalam *plaintext*
- Kesalahan konfigurasi *cloud storage*

Pertanyaan analisis:

Apakah data sensitif dapat bocor?

5. *Denial of Service* (Gangguan Layanan)

Denial of Service (DoS) bertujuan **membuat sistem atau layanan tidak dapat digunakan** oleh pengguna yang sah.

Ancaman ini **menyerang Availability**.

Contoh:

- Serangan *DDoS*
- *Resource exhaustion*
- Aplikasi *crash* karena input tidak valid

Pertanyaan analisis:

Apakah sistem dapat dibuat tidak tersedia?

6. *Elevation of Privilege* (Peningkatan Hak Akses)

Elevation of Privilege terjadi ketika **pengguna biasa berhasil mendapatkan hak akses lebih tinggi**, seperti menjadi admin.

Ancaman ini **memperbesar dampak seluruh ancaman lainnya**.

Contoh:

- Bug yang memungkinkan akses admin

- Salah konfigurasi hak akses
- *Privilege escalation exploit*

Pertanyaan analisis:

Apakah pengguna bisa mendapatkan hak lebih dari seharusnya?

Hubungan STRIDE dengan CIA Triad

STRIDE	CIA Terdampak
<i>Spoofing</i>	Confidentiality
<i>Tampering</i>	Integrity
<i>Repudiation</i>	Integrity
<i>Information Disclosure</i>	Confidentiality
<i>Denial of Service</i>	Availability
<i>Elevation of Privilege</i>	Confidentiality, Integrity, Availability

Catatan penting:

STRIDE membantu menjawab “bagaimana sistem diserang”, CIA menjawab “apa yang rusak”.

Contoh Penggunaan STRIDE dalam *Threat Modeling*

Pada sistem akademik:

- *Spoofing*: akun dosen dipalsukan
- *Tampering*: nilai diubah
- *Repudiation*: tidak ada log perubahan
- *Information Disclosure*: data mahasiswa bocor
- *Denial of Service*: server KRS down
- *Elevation of Privilege*: mahasiswa menjadi admin

STRIDE memastikan **tidak ada kategori ancaman penting yang terlewat**.

STRIDE adalah alat berpikir yang kuat namun sederhana untuk memahami ancaman keamanan. Dengan STRIDE:

- ancaman dapat **diidentifikasi secara sistematis**,
- diskusi keamanan menjadi **terstruktur dan objektif**,
- dan analisis risiko menjadi **lebih lengkap dan rasional**.

STRIDE mengajarkan bahwa **keamanan bukan tentang paranoia**, tetapi tentang **memahami cara sistem bisa gagal sebelum kegagalan itu terjadi**.

Contoh Tabel Risk Scoring (Manual)

Skala Penilaian (Disepakati di Awal)

Likelihood (Kemungkinan Terjadi)

- 1 = Sangat jarang
- 2 = Jarang
- 3 = Mungkin terjadi
- 4 = Sering
- 5 = Sangat sering

Impact (Dampak Jika Terjadi)

- 1 = Dampak sangat kecil
- 2 = Dampak kecil
- 3 = Dampak sedang
- 4 = Dampak besar
- 5 = Dampak sangat besar / kritikal

Rumus sederhana

$$\text{Risk Score} = \text{Likelihood} \times \text{Impact}$$

Tabel Analisis Risiko

ID Risiko	Aset	Ancaman	Kerentanan	CIA Terdampak	Likelihood (1–5)	Impact (1–5)	Risk Score	Level Risiko	Catatan
R1	Data nilai mahasiswa	Akun admin bocor	Password lemah	C, I	3	4	12	Tinggi	Perlu MFA
R2	Sistem KRS	Server down	Tidak ada failover	A	4	3	12	Tinggi	Gangguan akademik
R3	Database akademik	Ransomware	Backup tidak rutin	C, I, A	2	5	10	Tinggi	Risiko kritikal
R4	Akun dosen	Phishing	Edukasi rendah	C, I	4	2	8	Sedang	Perlu pelatihan
R5	Website kampus	Deface	Patch terlambat	I	2	2	4	Rendah	Dampak reputasi

Klasifikasi Level Risiko (Contoh)

Risk Score	Level Risiko
1 – 4	Rendah
5 – 9	Sedang
10 – 14	Tinggi
15 – 25	Sangat Tinggi / Kritis

Catatan penting:

Batas level dapat disesuaikan, yang penting **konsisten dan logis**.

Contoh Interpretasi (Untuk Laporan Mahasiswa)

- **R1 (Risk Score 12 – Tinggi)**
Risiko kebocoran akun admin memiliki dampak besar terhadap *integrity* nilai mahasiswa dan perlu diprioritaskan.
- **R3 (Risk Score 10 – Tinggi)**
Walaupun kemungkinan tidak sering, dampak *ransomware* sangat besar karena melanggar seluruh aspek CIA.
- **R5 (Risk Score 4 – Rendah)**
Risiko deface website berdampak terbatas dan dapat ditangani setelah risiko kritis.

Versi Ringkas (Untuk Kelas Non-Teknis)

Jika ingin **lebih sederhana**, gunakan tabel berikut:

Aset	Ancaman	Likelihood	Impact	Risk	Prioritas
Data nilai	Akun bocor	Sedang	Tinggi	Tinggi	1
Sistem KRS	Server down	Tinggi	Sedang	Tinggi	2
Website	Deface	Rendah	Rendah	Rendah	4

Skill yang Dilatih dengan Tabel Ini

Mahasiswa belajar bahwa:

- **Risiko bukan tebakan**, tetapi hasil penilaian terstruktur
- **Angka membantu diskusi**, bukan kebenaran absolut
- **Prioritas keamanan harus rasional**, bukan emosional
- **Keamanan adalah keputusan**, bukan sekadar teknologi

PRAKTEK 2: Risk Assessment & Threat Modeling

Pada bab sebelumnya, mahasiswa telah belajar **mengidentifikasi aset dan ancaman**. Namun, identifikasi saja **belum cukup**. Tidak semua ancaman memiliki **tingkat bahaya yang sama**, dan tidak semua risiko harus ditangani dengan prioritas yang setara. Oleh karena itu, praktikum ini difokuskan pada **penilaian risiko (*risk assessment*) dan threat modeling**.

Praktikum ini **tidak menggunakan alat keamanan teknis**, melainkan melatih **cara berpikir analitis**. Mahasiswa akan belajar **menilai, membandingkan, dan memprioritaskan risiko** secara rasional, menggunakan pendekatan sederhana namun terstruktur.

Tujuan

Setelah menyelesaikan praktek, mahasiswa **mampu**:

- Melakukan penilaian risiko keamanan informasi
- Mengidentifikasi ancaman menggunakan pendekatan **STRIDE**
- Memberi skor risiko berdasarkan *likelihood* dan *impact*
- Menentukan prioritas mitigasi secara logis
- Menyusun dokumen analisis risiko (*risk register*)

Konsep yang Digunakan

Praktikum ini menggunakan dua konsep inti:

1. **Risk = Likelihood × Impact**
Risiko ditentukan oleh **kemungkinan ancaman terjadi** dan **besarnya dampak jika terjadi**.
2. **STRIDE**
Digunakan sebagai **alat bantu berpikir**, bukan teori berat, untuk memastikan **jenis ancaman penting tidak terlewat**.

Tools yang Digunakan

- **Spreadsheet** (Excel / Google Sheets / LibreOffice)
- **Template *risk register*** (tabel analisis risiko)

Tidak diperlukan software keamanan khusus.

Langkah Praktek (Step-by-Step)

Langkah 1 — Menentukan Aplikasi Fiktif

Mahasiswa memilih **satu aplikasi fiktif**, misalnya:

- Sistem Akademik Kampus
- Aplikasi Pemesanan Online
- Aplikasi Layanan Publik

Contoh yang digunakan di praktikum ini:
Aplikasi Akademik Kampus (SIKAD)

Langkah 2 — Memetakan Sistem Secara Sederhana

Mahasiswa menggambarkan sistem secara **konseptual**, bukan teknis detail.

Contoh komponen utama:

- Pengguna: Mahasiswa, Dosen, Admin
- Aplikasi Web
- Database Akademik
- Jaringan / Internet

Contoh alur data sederhana:

Mahasiswa → Login → SIKAD → Database Nilai

Tujuan tahap ini adalah **memahami bagaimana data mengalir**, bukan bagaimana sistem dikodekan.

Langkah 3 — Menentukan Aset Utama

Mahasiswa memilih **aset paling bernilai**, misalnya:

- Data nilai mahasiswa
- Akun admin akademik
- Database akademik
- Sistem KRS

Prinsip penting:

Jika aset rusak, bocor, atau tidak tersedia → ada dampak nyata.

Langkah 4 — Identifikasi Ancaman dengan STRIDE

Mahasiswa menentukan **5 ancaman utama** menggunakan pendekatan *STRIDE*.

Contoh ancaman pada SIAKAD:

- *Spoofing*: akun admin dipalsukan
- *Tampering*: nilai mahasiswa diubah
- *Information Disclosure*: data mahasiswa bocor
- *Denial of Service*: sistem KRS tidak bisa diakses
- *Elevation of Privilege*: mahasiswa menjadi admin

Tidak semua kategori STRIDE harus digunakan, yang penting relevan.

Langkah 5 — Menilai Likelihood dan Impact

Mahasiswa memberi skor **secara subjektif namun konsisten**.

Skala sederhana:

- *Likelihood* (1–5): seberapa mungkin terjadi
- *Impact* (1–5): seberapa besar dampaknya

Tidak ada jawaban “benar” absolut. Yang dinilai adalah logika.

Langkah 6 — Menghitung Risk Score

Rumus sederhana:

$$\text{Risk Score} = \text{Likelihood} \times \text{Impact}$$

Contoh Risk Register (Output Utama)

ID	Aset	Ancaman	STRIDE	Likelihood	Impact	Risk Score	Level Risiko
R1	Data nilai	Nilai diubah	<i>Tampering</i>	3	4	12	Tinggi
R2	Sistem KRS	Server down	<i>DoS</i>	4	3	12	Tinggi
R3	Akun admin	Phishing	<i>Spoofing</i>	3	5	15	Sangat Tinggi
R4	Database	Kebocoran data	<i>Information Disclosure</i>	2	5	10	Tinggi
R5	Sistem	Mahasiswa jadi admin	<i>Elevation of Privilege</i>	2	4	8	Sedang

Langkah 7 — Menentukan Prioritas Mitigasi

Mahasiswa **mengurutkan risiko** dari yang paling kritis.

Contoh prioritas:

1. R3 – Akun admin bocor
2. R1 – Nilai diubah
3. R2 – KRS tidak tersedia
4. R4 – Kebocoran data
5. R5 – Privilege escalation

Mahasiswa **tidak diminta menentukan solusi teknis detail**, cukup:

- **ditangani segera**
- **diterima sementara**
- **ditunda**

Output yang Harus Dikumpulkan

1. **Risk Register (Spreadsheet)**
 - Minimal 5 risiko
 - Ada *likelihood*, *impact*, dan *risk score*
2. **Ringkasan Prioritas Mitigasi (1 halaman)**
 - Risiko tertinggi
 - Alasan prioritas
 - Dampak jika tidak ditangani

Template Risk Register

A. Informasi Umum (Diisi di Bagian Atas Spreadsheet)

Item	Keterangan
Nama Sistem	
Jenis Sistem	(Kampus / UMKM / Publik / Fiktif)
Tim / Nama Mahasiswa	
Tanggal Analisis	
Versi Dokumen	

B. Skala Penilaian (Disepakati di Awal)

Likelihood (Kemungkinan Terjadi)

Nilai	Deskripsi
1	Sangat jarang
2	Jarang
3	Mungkin terjadi
4	Sering
5	Sangat sering

Impact (Dampak Jika Terjadi)

Nilai	Deskripsi
1	Dampak sangat kecil
2	Dampak kecil
3	Dampak sedang
4	Dampak besar
5	Dampak sangat besar / kritikal

Rumus

Risk Score = Likelihood × Impact

C. Tabel Risk Register Utama

ID Risiko	A set	Deskri psi Risiko	Anca man	Kerenta nan	STRI DE	CIA Terdam pak	Likelih ood (1–5)	Imp act (1–5)	Ris k Sc ore	Lev el Risi ko	Mitig asi Awal	Stat us
R1												
R2												
R3												
R4												
R5												

Kolom penting (wajib diisi):

- **Aset**
- **Ancaman**
- **Likelihood**
- **Impact**
- **Risk Score**
- **Level Risiko**

Kolom *Mitigasi* dan *Status* dapat diisi singkat.

D. Klasifikasi Level Risiko (Contoh)

Risk Score	Level Risiko
1 – 4	Rendah
5 – 9	Sedang
10 – 14	Tinggi
15 – 25	Sangat Tinggi

Batas dapat disesuaikan, yang penting konsisten.

E. Contoh Pengisian (Sebagai Referensi)

ID Risiko	Aset	Deskripsi Risiko	Ancaman	Kerentanan	STRIDE	CI A	Likelihood	Impact	Risk Score	Level	Mitigasi Awal	Status
R1	Data nilai	Nilai diubah tanpa izin	Akun bocor	Password lemah	Tampering	C, I	3	4	12	Tinggi	MFA	Open
R2	Sistem KRS	Sistem tidak bisa diakses	Server down	Tidak ada failover	DoS	A	4	3	12	Tinggi	Redundansi	Open

F. Ringkasan Prioritas Risiko (Opsional tapi Disarankan)

Prioritas	ID Risiko	Alasan
1	R1	Dampak akademik besar
2	R2	Ganggu layanan kritikal
3	R4	Risiko reputasi

Keterampilan yang Dilatih

Melalui praktikum ini, mahasiswa mengembangkan:

- **Kemampuan melakukan analisis risiko secara sistematis**, mulai dari identifikasi aset, ancaman, hingga dampak yang mungkin terjadi.
- **Kemampuan melakukan *threat modeling***, yaitu memetakan bagaimana suatu sistem dapat diserang atau gagal, bukan hanya bagaimana sistem tersebut berfungsi.
- **Kemampuan membandingkan berbagai risiko**, sehingga mahasiswa dapat melihat perbedaan tingkat bahaya antar risiko.
- **Kemampuan menentukan prioritas risiko**, dengan fokus pada risiko yang paling signifikan dan berdampak besar.
- **Kemampuan mengkomunikasikan risiko secara tertulis**, melalui penyusunan *risk register* dan ringkasan analisis yang jelas dan terstruktur.
- **Kesadaran bahwa keamanan informasi merupakan proses pengambilan keputusan**, bukan sekadar penggunaan alat atau teknologi tertentu.

Catatan Penting untuk Mahasiswa

- **Risk score bukan kebenaran absolut**, melainkan **alat bantu berpikir** untuk mendukung diskusi keamanan yang rasional dan dapat dipertanggungjawabkan.
- **Angka risiko digunakan untuk membandingkan dan memprioritaskan**, bukan untuk menunjukkan kepastian kejadian.

Catatan Tambahan

- **Risk register bukan dokumen teknis**, tetapi **dokumen pengambilan keputusan**.
- Penilaian dosen difokuskan pada:
 - **logika dalam penilaian risiko**,
 - **konsistensi skor antar risiko**,
 - **kemampuan menjelaskan alasan penetapan prioritas** secara jelas dan masuk akal.

Penutup

Praktikum ini menegaskan bahwa **keamanan informasi dimulai dari pemahaman risiko**, bukan dari membeli alat atau menulis kode. Mahasiswa yang mampu melakukan *risk assessment* dan *threat modeling* telah memiliki **fondasi berpikir keamanan yang profesional**, yang dapat diterapkan di **kampus, industri, pemerintahan, maupun UMKM**.

BAB 3: Kriptografi untuk Praktisi

Fokus utama: Menggunakan kriptografi dengan benar, bukan menghafal rumus

Pada tahap ini, kita **berhenti membicarakan kriptografi sebagai rumus matematika** dan mulai memperlakukannya sebagai **alat kerja harian**. Di dunia nyata, hampir semua engineer, sysadmin, dan developer **tidak pernah menghitung kriptografi secara manual**—yang mereka lakukan adalah **menggunakan library yang benar, dengan cara yang benar**.

Kriptografi di sini diposisikan sebagai **alat perlindungan data**, bukan sebagai *mystical black box*. Jika digunakan dengan benar, kriptografi membuat sistem **aman, tahan bocor, dan terpercaya**. Namun jika digunakan salah, ia justru bisa **memberi rasa aman palsu** (*false sense of security*).

Tujuan pertemuan ini sederhana namun krusial:

setelah selesai, pembaca mampu enkripsi, melakukan hashing, dan menyimpan password dengan aman menggunakan Python.

Kriptografi dalam Dunia Nyata (Tanpa Ribet)

Kriptografi modern bukan tentang “bagaimana algoritma bekerja”, melainkan:

- **Kapan harus hashing**
- **Kapan harus encryption**
- **Bagaimana menyimpan dan mengelola key**
- **Bagaimana tidak membuat kesalahan fatal**

Sebagian besar kebocoran data di dunia **bukan karena algoritmanya lemah**, tetapi karena:

- developer salah pakai,
- konfigurasi asal jalan,
- atau *copy-paste* kode tanpa paham konteks.

Kriptografi yang salah = tidak ada kriptografi sama sekali.

Hashing — Bukan untuk Disimpan Balik

Hashing adalah proses mengubah data menjadi nilai tetap (*fixed-length*) yang **tidak bisa dikembalikan ke bentuk awal**. Ini **satu arah** (*one-way*).

Hashing **bukan enkripsi**, dan ini penting untuk diingat.

Hashing **digunakan untuk**:

- **password**
- verifikasi integritas file

- fingerprint data

Hashing **tidak digunakan untuk:**

- menyimpan data pribadi yang perlu dibaca ulang
- menyimpan nomor KTP, email, atau alamat

Contoh sederhana:

- Password:  **hash**
- File upload check:  **hash**
- Nomor KTP:  **jangan hash, enkripsi**

Hal penting yang **WAJIB** diingat:

- Jangan pernah pakai *MD5* atau *SHA1*
- Gunakan algoritma modern seperti *bcrypt*, *scrypt*, atau *argon2*

**Password tidak pernah disimpan.
Yang disimpan hanyalah hasil hash-nya.**

Encryption — Saat Data Harus Bisa Dibaca Lagi

Jika hashing bersifat satu arah, **encryption (enkripsi)** bersifat **dua arah**.

Artinya:

- data enkripsi → aman saat disimpan
- data dekripsi → bisa dibaca kembali

Encryption **digunakan untuk:**

- data pribadi (*PII*)
- token
- API key
- data sensitif di database
- backup

Encryption **selalu melibatkan kunci (key)**.

Jika key bocor, maka data ikut bocor.

Jenis encryption yang umum dipakai:

- **Symmetric encryption** (*satu key*)
 - cepat
 - cocok untuk database
- **Asymmetric encryption** (*public & private key*)
 - cocok untuk komunikasi dan distribusi key

Dalam praktik sehari-hari, **AES** adalah pilihan standar dan aman.

Key Management — Bagian yang Paling Sering Diabaikan

Sebagus apapun algoritma kriptografi, **jika key disimpan sembarangan, semuanya percuma.**

Kesalahan umum:

- key ditulis langsung di source code
- key di-*commit* ke GitHub
- key disimpan dalam file tanpa proteksi
- satu key dipakai untuk semua data

Prinsip dasar **key management**:

- **Key ≠ password**
- **Key tidak pernah di-hardcode**
- **Key disimpan di environment variable atau secret manager**
- **Key bisa diganti (rotasi)**

Key adalah kunci rumah digital Anda.

Menaruhnya di depan pintu sama saja mengundang pencuri.

Kesalahan Umum Kriptografi (Ini yang Paling Sering Terjadi)

Bagian ini penting karena **99% masalah keamanan berasal dari sini**, bukan dari algoritma.

Kesalahan yang sering dilakukan:

- Menggunakan algoritma lama (*MD5*, *SHA1*)
- Enkripsi tanpa *authentication*
- Reuse key untuk banyak fungsi
- Menyimpan password dalam bentuk plaintext
- Menganggap encoding (*Base64*) sebagai enkripsi ❌

Perlu ditegaskan:

- *Base64* ❌ **bukan enkripsi**
- *Hash* ❌ **bukan untuk data pribadi**
- *Encryption* ❌ **bukan untuk password**

Jika satu prinsip ini dipahami, **80% kesalahan krypto sudah terhindari.**

PRAKTEK 3: Kriptografi Praktis

Tujuan

Setelah praktek, mahasiswa **benar-benar bisa mengamankan data**, bukan sekadar tahu istilah.

Mahasiswa mampu:

- Menyimpan **password dengan benar**
- Mengenkripsi **data pribadi sensitif**
- Membedakan **hashing vs encryption** secara nyata
- Memahami **kesalahan fatal yang sering terjadi di sistem nyata**
- Menghasilkan **script Python siap pakai (production-ready)**

Konsep Kunci (Wajib Dipahami Sebelum Koding)

Hash $\neq$ Enkripsi

Aspek	Hashing	Enkripsi
Bisa dibalik?	✗ Tidak	✓ Bisa (pakai key)
Tujuan	Verifikasi	Perlindungan data
Contoh	Password	Email, NIK, Token
Jika DB bocor	Relatif aman	Aman jika key aman

Kesalahan paling umum di dunia nyata:

- ✗ Password dienkripsi
- ✗ Password disimpan plaintext
- ✗ Password pakai MD5/SHA1
- ✗ Tidak pakai salt

Golden Rule Password Security

Password:

- ✗ tidak pernah disimpan plaintext
- ✗ tidak pernah dienkripsi
- ✓ selalu di-hash
- ✓ selalu pakai salt
- ✓ selalu pakai algoritma khusus password

Kenapa tidak dienkripsi?

Karena password **tidak perlu dibaca kembali**, cukup **dibandingkan hash-nya**.

Tools & Environment

Sistem

- OS: **Ubuntu 24.04 LTS**
- Python: **≥ 3.10**
- Semua library: **Open Source**

Install Dependency

```
sudo apt update
sudo apt install -y python3 python3-pip gnupg
pip install bcrypt cryptography
```

Praktek 1 — Hash Password dengan Benar

Contoh SALAH (Sering Terjadi di Sistem Nyata)

```
import hashlib

password = "rahasia123"
hash = hashlib.sha256(password.encode()).hexdigest()
print(hash)
```

Masalah besar:

-  Tidak ada salt
-  Rentan rainbow table
-  SHA-256 bukan algoritma password

Cara BENAR — Password Hashing dengan bcrypt

Prinsip bcrypt

- Built-in salt
- Slow by design (anti brute force)
- Standar industri (Linux, Django, Flask, dll)

Script Aman (Production-Ready)

```
import bcrypt

# Simulasi input user
password = b"PasswordKuat123!"

# Hash password
```

```

hashed = bcrypt.hashpw(password, bcrypt.gensalt())

print("Hash tersimpan di database:")
print(hashed)

# =====
# Verifikasi saat login
# =====
input_password = b"PasswordKuat123!"

if bcrypt.checkpw(input_password, hashed):
    print("Login berhasil")
else:
    print("Password salah")

```

Penjelasan

- `bcrypt.gensalt()` → otomatis generate salt
- `hashpw()` → hash + salt
- `checkpw()` → bandingkan tanpa tahu password asli

Real Case

Jika database bocor:

```

user | password_hash
-----
andi | $2b$12$K9x...

```

Attacker **tidak bisa login**, tidak bisa decrypt, tidak tahu password.

Ini adalah definisi sistem yang benar-benar aman.

Praktek 2 — Enkripsi Data Pribadi (AES)

Sekarang kita **butuh enkripsi**, bukan hashing.

Data yang WAJIB dienkripsi:

- Email
- NIK
- Nomor HP
- Token API
- Data medis
- Data akademik

Konsep AES (Symmetric Encryption)

- Data **bisa dikembalikan**
- Pakai **secret key**
- Key **HARUS** dijaga

Enkripsi AES dengan Python (**cryptography**)

Step 1 — Generate Key

```
from cryptography.fernet import Fernet

key = Fernet.generate_key()

with open("secret.key", "wb") as f:
    f.write(key)

print("Key disimpan ke secret.key")
```

JANGAN commit key ke Git

Step 2 — Enkripsi Data

```
from cryptography.fernet import Fernet

# Load key
key = open("secret.key", "rb").read()
cipher = Fernet(key)

data = "nik=3201010101010001;email=andi@email.com"
encrypted = cipher.encrypt(data.encode())

print("Encrypted data:")
print(encrypted)
```

Step 3 — Dekripsi Data

```
decrypted = cipher.decrypt(encrypted)
print("Data asli:")
print(decrypted.decode())
```

Perbedaan Hash vs Enkripsi (Nyata)

Operasi	Output
Hash password	Tidak bisa dibalik
Enkripsi NIK	Bisa dibaca kembali
Login	Compare hash
Kirim data	Decrypt

Praktek 3 — Simulasi Kesalahan Fatal (Plaintext Storage)

Sistem Salah (TAPI SERING ADA)

```
user = {  
  "username": "andi",  
  "password": "Password123",  
  "nik": "3201010101010001"  
}
```

Jika bocor:

- Password bocor
- Identitas bocor
- Melanggar UU PDP

Sistem Benar

```
user = {  
  "username": "andi",  
  "password_hash": "<bcrypt hash>",  
  "nik_encrypted": "<AES ciphertext>"  
}
```

Praktek 4 — GnuPG (GNU Privacy Guard)

Tujuan Pembelajaran

Mahasiswa mampu:

- Memahami **konsep kriptografi asimetris**
- Membuat dan mengelola **key GnuPG**
- Mengenkripsi & mendekripsi **file**
- Mengenkripsi & mendekripsi **pesan (text/message)**
- Menggunakan GnuPG untuk **key management system nyata**
- Menghindari **kesalahan fatal pengelolaan key**

Konsep Dasar GnuPG

Kriptografi Asimetris

Komponen	Fungsi
Public Key	Untuk enkripsi
Private Key	Untuk dekripsi
Passphrase	Melindungi private key

Analogi sederhana:

- Public key = alamat rumah
- Private key = kunci rumah

Semua orang boleh punya alamat rumah,

hanya pemilik kunci yang bisa membuka pintu.

Kapan GnuPG Dipakai?

GnuPG cocok untuk:

- Enkripsi **file backup**
- Enkripsi **key AES / API token**
- Distribusi data antar admin
- Proteksi dokumen sensitif
- Sistem **tanpa cloud / offline**

Bukan untuk hashing password.

Setup Awal (Ubuntu 24.04)

Install GnuPG

```
sudo apt update
sudo apt install -y gnupg
```

Cek versi:
`gpg --version`

BAGIAN 1 — Membuat Key Pair GnuPG

Step 1 — Generate Key

```
gpg --full-generate-key
```

Pilihan yang Direkomendasikan

Please select what kind of key you want:

```
(1) RSA and RSA  ← PILIH INI
```

```
RSA keys may be between 1024 and 4096 bits.
4096
```

```
Key is valid for? (0)  ← no expiry (atau 1y untuk
produksi)
```

```
Name: Admin Kampus
Email: admin@kampus.ac.id
```

Masukkan **passphrase kuat**.

Step 2 — Lihat Key yang Dibuat

```
gpg --list-keys  
gpg --list-secret-keys
```

Contoh output:

```
pub   rsa4096 2026-01-20 [SC]  
uid   Admin Kampus <admin@kampus.ac.id>  
sub   rsa4096 2026-01-20 [E]
```

BAGIAN 2 — Enkripsi File (Public Key)

Contoh File Sensitif

```
echo "INI DATA SANGAT RAHASIA" > data_rahasia.txt
```

Enkripsi File

```
gpg --encrypt \  
--recipient admin@kampus.ac.id \  
data_rahasia.txt
```

Hasil:

```
data_rahasia.txt.gpg
```

File asli masih ada → sebaiknya dihapus manual.

```
shred -u data_rahasia.txt
```

BAGIAN 3 — Dekripsi File (Private Key)

```
gpg --decrypt data_rahasia.txt.gpg
```

Atau simpan ke file:

```
gpg --decrypt data_rahasia.txt.gpg > data_buka.txt
```

Akan diminta **passphrase private key**.

BAGIAN 4 — Enkripsi & Dekripsi Message (Text)

Enkripsi Pesan

```
echo "Password database: RAHASIA" | \  
gpg --encrypt --armor -r admin@kampus.ac.id
```

Output:

```
-----BEGIN PGP MESSAGE-----
```

```
hQIMAz...
-----END PGP MESSAGE-----
```

--armor → format teks (ASCII), cocok email/chat.

Dekripsi Pesan

```
gpg --decrypt message.asc
```

Atau langsung:

```
echo "-----BEGIN PGP MESSAGE-----..." | gpg --decrypt
```

BAGIAN 5 — Enkripsi Simetris (Tanpa Public Key)

Digunakan jika:

- Tidak perlu distribusi key publik
- Sistem lokal / offline

Enkripsi File

```
gpg --symmetric rahasia.txt
```

Masukkan passphrase.

Dekripsi

```
gpg --decrypt rahasia.txt.gpg
```

Kurang ideal untuk tim besar.

BAGIAN 6 — Proteksi AES Key (Best Practice)

Contoh Kasus Nyata

- Python AES key → **jangan disimpan plaintext**

```
gpg --encrypt -r admin@kampus.ac.id secret.key
rm secret.key
```

Saat Dibutuhkan:

```
gpg --decrypt secret.key.gpg > secret.key
```

Dipakai di:

- Server kampus
- Sistem pemerintah
- Pipeline AI offline

BAGIAN 7 — Export & Import Public Key (Distribusi Aman)

Export Public Key

```
gpg --export --armor admin@kampus.ac.id > publickey.asc
```

Import Public Key

```
gpg --import publickey.asc
```

Public key boleh dibagikan bebas.

BAGIAN 8 — Kesalahan Fatal (WAJIB DIHINDARI)

- ✗ Private key di-share
- ✗ Passphrase kosong
- ✗ Key disimpan di GitHub
- ✗ Enkripsi tapi key bocor
- ✗ Tidak backup key

BAGIAN 9 — Backup & Recovery Key

```
gpg --export-secret-keys admin@kampus.ac.id > private_backup.gpg
```

Simpan di:

- Flashdisk offline
- Brankas fisik
- Air-gapped storage

Integrasi dengan Sistem Nyata

Digunakan untuk:

- Enkripsi backup database
- Proteksi token API
- Enkripsi dataset riset
- Secure DevOps pipeline
- Sistem offline-first

Output Skill Mahasiswa

Mahasiswa **bukan hanya paham**, tapi **bisa melakukan**:

- ✓ Generate & manage key
- ✓ Enkripsi file & message
- ✓ Proteksi key AES
- ✓ Key distribution
- ✓ Backup & recovery key

Analisis Kesalahan Umum (Wajib Masuk Laporan)

1. Menyimpan password plaintext
2. Enkripsi password (salah konsep)
3. Tidak pakai salt
4. Menyimpan key di source code
5. Commit key ke GitHub

Penutup

Sistem aman **bukan sistem yang tidak bisa dibobol**,
tetapi sistem yang **tetap aman meskipun database bocor**.

Praktikum ini bukan latihan akademik,
melainkan **fondasi keamanan nyata** untuk:

- Sistem kampus
- Startup
- AI pipeline
- Sistem pemerintah
- Infrastruktur data nasional

BAB 4: Keamanan Jaringan & *Traffic Analysis*

Fokus: *Visibility*

Keamanan jaringan pada praktiknya bukan ditentukan oleh seberapa banyak alat yang dipasang, melainkan oleh **kemampuan melihat dan memahami lalu lintas jaringan secara nyata**. Banyak sistem terlihat “aman” di permukaan, tetapi sebenarnya buta terhadap apa yang sedang terjadi di dalamnya. Tanpa *visibility*, serangan dapat berjalan lama tanpa terdeteksi, log tidak pernah dibaca dengan benar, dan ketika insiden terjadi, tidak ada bukti teknis yang kuat untuk menjelaskan apa yang salah.

Visibility berarti kita mampu mengetahui **siapa berkomunikasi dengan siapa, kapan pola trafik berubah, dan apa indikasi awal dari aktivitas yang mencurigakan**. Dengan *visibility*, keamanan tidak lagi berbasis asumsi, tetapi berbasis **data dan pengamatan**. Inilah fondasi utama dari *traffic analysis* dan sistem deteksi intrusi.

Network Attack Basics

Serangan jaringan pada dasarnya selalu meninggalkan jejak dalam bentuk **pola komunikasi yang tidak wajar**. Walaupun teknik serangan bisa berkembang, prinsip dasarnya relatif sama: attacker harus berkomunikasi dengan target. Tugas defender adalah **mengenali pola komunikasi tersebut sebelum berdampak besar**.

Beberapa pola serangan yang umum antara lain:

- **Port scanning**, di mana satu sumber melakukan banyak koneksi singkat ke berbagai port. Perilaku ini jarang dilakukan oleh aplikasi normal dan biasanya menjadi tahap awal eksplorasi sistem.
- **Brute force login**, yang ditandai dengan percobaan autentikasi berulang dalam waktu singkat. Kesalahan umum adalah menganggapnya sebagai kesalahan pengguna, padahal sering kali itu adalah serangan otomatis.
- **DoS/DDoS**, yang terlihat dari lonjakan trafik ekstrem hingga layanan menjadi lambat atau tidak bisa diakses. Serangan ini tidak selalu bertujuan menembus sistem, tetapi menghabiskan sumber daya.
- **Man-in-the-Middle**, yang lebih sulit dideteksi karena menyisip di tengah komunikasi. Indikasinya dapat berupa perubahan rute, perilaku ARP yang tidak normal, atau sertifikat yang mencurigakan.

Intinya, **trafik normal cenderung stabil dan dapat diprediksi**, sedangkan serangan hampir selalu menciptakan anomali.

Konsep IDS & IPS

Untuk meningkatkan *visibility*, digunakan **IDS (*Intrusion Detection System*)** dan **IPS (*Intrusion Prevention System*)**. Keduanya memiliki tujuan yang sama, yaitu mengenali serangan, tetapi dengan pendekatan yang berbeda.

IDS bersifat pasif. Sistem ini memantau trafik, menganalisis pola, lalu memberikan peringatan ketika menemukan indikasi serangan. IDS sangat cocok untuk lingkungan belajar, audit, dan tahap awal penerapan keamanan karena memungkinkan administrator memahami trafik tanpa resiko memblokir koneksi yang sah.

IPS, sebaliknya, bersifat aktif. Selain mendeteksi, IPS dapat langsung **menghentikan atau memblokir trafik** yang dianggap berbahaya. Pendekatan ini efektif untuk lingkungan produksi, tetapi membawa risiko *false positif* yang dapat menyebabkan gangguan layanan jika tidak dikonfigurasi dengan matang.

Secara praktis, pendekatan yang disarankan adalah:

- **Pahami trafik dengan IDS terlebih dahulu**
- Terapkan IPS setelah pola normal benar-benar dipahami
- Hindari memblokir sebelum yakin terhadap pola serangan

Cara IDS/IPS Mengenali Serangan

Secara umum, IDS dan IPS bekerja dengan dua pendekatan utama:

- **Signature-based**, yaitu mencocokkan trafik dengan pola serangan yang sudah dikenal. Metode ini cepat dan akurat untuk serangan lama, tetapi tidak efektif untuk teknik baru.
- **Anomaly-based**, yaitu mendeteksi penyimpangan dari perilaku normal. Pendekatan ini lebih adaptif, tetapi membutuhkan pemahaman yang baik tentang trafik normal agar tidak salah deteksi.

Prinsip pentingnya adalah **tidak mungkin mendeteksi anomali tanpa memahami kondisi normal terlebih dahulu**.

Penutup

Bab ini menekankan bahwa keamanan jaringan bukan sekadar urusan perangkat keras atau perangkat lunak, melainkan **kemampuan membaca dan menafsirkan trafik sebagai data**. Dengan *visibility*, jaringan berubah dari sesuatu yang “gelap” menjadi sistem yang dapat diawasi, dianalisis, dan dijelaskan secara teknis. Bekal inilah yang akan digunakan pada bab berikutnya, ketika pembaca mulai berhadapan langsung dengan **trafik nyata dan analisis praktis**.

PRAKTEK 4: Analisis Trafik Jaringan

Modul ini dirancang **praktis dan membumi**, seperti yang biasa dibutuhkan di **keamanan jaringan nyata**: membaca trafik, memahami apa yang *benar-benar lewat di kabel*, dan menyadari **perbedaan fundamental antara komunikasi aman dan tidak aman**.

Semua praktik menggunakan **Ubuntu Linux 24.04**, **open-source tools**, **tanpa software proprietary**, dan **reproducible di lab kampus maupun laptop pribadi**.

Tujuan

Setelah modul ini, mahasiswa **mampu**:

- Membaca dan memahami **trafik jaringan mentah**
- Mengidentifikasi **packet, protocol, dan payload**
- Menemukan **data sensitif yang bocor**
- Membandingkan **trafik tidak aman (HTTP)** vs **trafik aman (HTTPS/TLS)**
- Menarik **kesimpulan keamanan berbasis bukti**

Mindset penting:

Keamanan jaringan bukan asumsi — tapi hasil observasi.

Konsep Dasar (Wajib Dipahami)

Packet

Unit data terkecil di jaringan.

Satu request web → **puluhan packet**.

Setiap packet berisi:

- Header (alamat, protokol)
- Payload (isi data)

Protocol

Aturan komunikasi, contoh:

- **HTTP** → teks polos
- **HTTPS (TLS)** → terenkripsi
- **DNS** → pencarian nama
- **TCP/UDP** → transport

Payload

Bagian paling berbahaya jika tidak dienkripsi

Contoh payload:

- Username
- Password
- Token login
- Cookie sesi

Jika payload **terbaca**, maka **siapa pun di jaringan bisa membacanya**

Tools Utama

Wireshark

- Network protocol analyzer
- Open source
- Digunakan oleh:
 - Network engineer
 - Security analyst
 - Forensic investigator

Persiapan Lingkungan (Ubuntu 24.04)

Install Wireshark

```
sudo apt update
sudo apt install wireshark -y
```

Saat instalasi:

- Pilih **Yes** → allow non-root users capture packets
- Tambahkan user ke group wireshark:
 - `sudo usermod -aG wireshark $USER`
 - `newgrp wireshark`

Cek Interface Jaringan

```
ip a
```

Biasanya:

- **eth0** (LAN)
- **wlan0** (WiFi)

Praktik 1 — Capture Trafik HTTP (Tidak Aman)

Tujuan

Melihat **data sensitif terbaca jelas**

Langkah 1 — Jalankan Wireshark

```
wireshark
```

Pilih interface aktif → **Start Capture**

Langkah 2 — Akses Website HTTP

Gunakan terminal:

```
curl http://httpbin.org/get
```

Atau buka browser:

```
http://neverssl.com
```

Langkah 3 — Filter HTTP di Wireshark

Di filter bar:

```
http
```

Klik salah satu packet → lihat:

- **Request Method**
- **Host**
- **URI**
- **Payload**

Analisis

Perhatikan:

- URL terbaca
- Parameter GET terbaca
- Header lengkap

Kesimpulan sementara:

HTTP **tidak melindungi isi komunikasi**

Praktik 2 — Simulasi Data Sensitif Bocor

Kirim Data Login Palsu (HTTP)

```
curl -X POST http://httpbin.org/post \
-d "username=student&password=rahasia123"
```

Di Wireshark:

- Expand **HTML Form URL Encoded**
- Password terlihat jelas

Fakta Penting

Jika ini:

- WiFi publik
- Jaringan kampus
- Jaringan kantor

Password bisa dicuri tanpa hacking akun

Praktik 3 — Capture Trafik HTTPS (Aman)

Akses HTTPS

```
curl https://httpbin.org/get
```

Filter:

```
tls
```

Atau:

```
tcp.port == 443
```

Apa yang Terlihat

- TLS Handshake
- Certificate exchange
- Encrypted Application Data

Payload tidak terbaca

Perbandingan HTTP vs HTTPS

Aspek	HTTP	HTTPS
Payload	Terbaca	Terenkripsi
Password	Bocor	Aman
Integrity	Tidak ada	Ada
Authentication	Tidak ada	Sertifikat

Praktik 4 — Analisis Metadata HTTPS

Walaupun isi terenkripsi, **metadata tetap terlihat**

Yang masih bisa dilihat:

- IP tujuan
- Domain (SNI)
- Waktu koneksi
- Ukuran data

Ini penting untuk:

- Traffic analysis
- Censorship
- Monitoring

Diskusi Keamanan

- HTTPS **melindungi isi**, bukan pola
- VPN menyembunyikan metadata sebagian
- Keamanan = **layered decision**

Praktik 5 — Capture & Analisis dengan Python (Advanced)

Install Library

```
sudo apt install python3-pip -y
pip install scapy
```

Script Capture Packet Sederhana

```
from scapy.all import sniff
```

```
def analyze(packet):  
    if packet.haslayer("IP"):  
        print(packet["IP"].src, "->", packet["IP"].dst)  
  
sniff(prn=analyze, count=10)
```

Catatan:

Python cocok untuk:

- Automation
 - Detection
 - Logging
- Bukan GUI inspection (Wireshark tetap utama)

Output yang Wajib Dikumpulkan

Screenshot

- HTTP payload terbaca
- HTTPS payload terenkripsi
- Filter yang digunakan

Laporan Singkat (1–2 halaman)

Wajib menjawab:

1. Apa perbedaan HTTP & HTTPS dari hasil capture?
2. Data apa yang berisiko bocor?
3. Mengapa HTTPS lebih aman?
4. Apakah HTTPS sepenuhnya aman? Jelaskan.

Refleksi Penting untuk Mahasiswa

*Keamanan jaringan bukan teori kriptografi.
Ia terlihat jelas saat kamu membuka packet.*

BAB 5: Deteksi Serangan (Rule-Based & AI-Based)

Fokus: Dari statis ke cerdas

Pada pertemuan ini, kita masuk ke inti dari dunia *cyber defense*: **bagaimana sebuah sistem mengenali bahwa sesuatu yang “aneh” sedang terjadi di jaringan**. Jika pada pertemuan sebelumnya kita belajar *melihat* trafik (visibility), maka sekarang kita belajar **mengambil keputusan: ini normal atau serangan?**

Perjalanan ini akan membawa pembaca dari pendekatan lama yang **statis dan kaku**, menuju pendekatan modern yang **adaptif dan cerdas**. Di sinilah perbedaan antara sistem yang *sekadar hidup* dan sistem yang *benar-benar waspada*.

Teori: Signature Detection vs Anomaly Detection

Signature-Based Detection (Rule-Based)

Pendekatan *signature-based* adalah **cara klasik** dalam mendeteksi serangan. Prinsipnya sederhana:

jika pola trafik cocok dengan pola serangan yang sudah dikenal, maka dianggap sebagai serangan.

Dengan kata lain, sistem bekerja seperti **kamus ancaman**.

- Setiap serangan punya *signature* (ciri khas)
- *Signature* ini disimpan dalam bentuk **rule**
- Trafik jaringan dibandingkan dengan rule tersebut

Contoh sederhananya:

“Jika ada request HTTP mengandung */etc/passwd*, maka ini adalah serangan.”

Pendekatan ini **sangat efektif** untuk:

- Serangan yang **sudah dikenal**
- *Exploit* yang polanya **jelas dan konsisten**
- Lingkungan dengan trafik yang relatif stabil

Namun, dibalik kesederhanaannya, tersembunyi masalah besar.

Anomaly-Based Detection (AI-Based / Behavior-Based)

Berbeda dengan *signature-based*, pendekatan *anomaly-based* **tidak bertanya:**

“apakah ini serangan yang saya kenal?”

Melainkan bertanya:

“apakah perilaku ini normal?”

Sistem terlebih dahulu **belajar pola trafik normal**, lalu akan menandai apa pun yang *menyimpang* dari kebiasaan tersebut sebagai potensi serangan.

Pendekatan ini biasanya melibatkan:

- *Statistical analysis*
- *Machine Learning*
- *Behavior modeling*

Contoh pemikiran sistem:

“Biasanya server ini hanya menerima 10 request/detik.
Kenapa sekarang tiba-tiba 2.000 request/detik?”

Inilah yang membuat *anomaly detection*:

- **Lebih adaptif**
- **Lebih tahan terhadap serangan baru (*zero-day*)**
- **Lebih dekat dengan cara manusia berpikir**

Namun, kecerdasan ini datang dengan harga tertentu.

Perbandingan Singkat: Signature vs Anomaly

Secara ringkas, perbedaannya bisa dirasakan sebagai berikut:

- **Signature-based**
 - Cepat
 - Akurat untuk serangan lama
 - Mudah dipahami
 - Tapi **buta terhadap serangan baru**
- **Anomaly-based**
 - Lebih fleksibel
 - Bisa mendeteksi pola baru
 - Cocok untuk lingkungan dinamis
 - Tapi **rawan *false positive*** jika tidak dilatih dengan baik

Di dunia nyata, **sistem keamanan modern hampir selalu menggabungkan keduanya.**

Kelemahan Rule-Based Detection (Kenapa Tidak Cukup?)

Di sinilah kita mulai bersikap **kritis**.

Walaupun *rule-based detection* terlihat rapi dan meyakinkan, ia memiliki kelemahan mendasar yang **sering menjadi titik kegagalan sistem keamanan**.

1. Tidak Bisa Melihat Serangan Baru (Zero-Day)

Rule hanya bisa mendeteksi **apa yang sudah diketahui**.

Jika penyerang:

- Memodifikasi payload
- Mengubah urutan serangan
- Menggunakan teknik baru

Maka sistem **akan menganggapnya normal**.

Rule tidak bisa menebak masa depan.

2. Rule Explosion (Ledakan Aturan)

Semakin banyak serangan:

- Semakin banyak rule
- Semakin kompleks konfigurasi
- Semakin sulit dikelola

Pada skala besar, ini menyebabkan:

- Kesalahan konfigurasi
- Rule saling bertabrakan
- Sistem sulit dipelihara

Ironisnya, **sistem keamanan bisa menjadi rapuh karena terlalu banyak aturan keamanan**.

3. Mudah Diakali (Bypassable)

Karena rule bersifat **deterministik**, penyerang yang cerdas bisa:

- Menghindari pola tertentu
- Menyisipkan noise
- Mengubah encoding data

Selama tidak *exactly match* dengan rule, serangan bisa **lolos tanpa terdeteksi**.

4. Tidak Memahami Konteks

Rule melihat **potongan trafik**, bukan **cerita lengkap**.

Contoh:

- Login gagal 1 kali → normal
- Login gagal 100 kali → serangan

Tanpa pemahaman konteks:

- Rule bisa terlalu permisif
- Atau terlalu ketat dan menghasilkan *false alarm*

Mengapa Kita Harus Bergerak ke Sistem yang Lebih Cerdas?

Dunia jaringan hari ini:

- Dinamis
- Cepat berubah
- Dipenuhi otomatisasi dan AI

Mengandalkan sistem yang **statis** untuk melawan ancaman yang **adaptif** adalah pertarungan yang tidak seimbang.

Penyerang belajar. Sistem pertahanan juga harus belajar.

Inilah alasan mengapa:

- *AI-based detection*
- *Behavior analysis*
- *Adaptive security*

bukan lagi *nice to have*, tetapi **kebutuhan mutlak**.

Mindset Penting untuk Mahasiswa

Pada akhir pertemuan ini, pembaca diharapkan menyadari satu hal penting:

- Keamanan **bukan soal alat**
- Keamanan **bukan soal banyaknya rule**
- Keamanan adalah **kemampuan berpikir, mengamati, dan beradaptasi**

Rule adalah fondasi.

AI adalah akselerator.

Manusia tetap pengambil keputusan utama.

Dan di situlah peran Anda sebagai praktisi keamanan dimulai.

PRAKTEK 5: Intrusion Detection Dasar

Tujuan

Setelah praktikum ini, mahasiswa **mampu mendeteksi serangan sederhana di jaringan, memahami jenis alert IDS**, dan **menafsirkan makna teknis di balik alert**, bukan hanya membaca log secara pasif.

Yang dilatih bukan hanya “menjalankan tool”, tetapi **cara berpikir security analyst**:

- Apa yang *normal*?
- Apa yang *mencurigakan*?
- Kenapa ini dianggap *serangan*?

Konsep Inti (Wajib Dipahami)

1. Apa itu Intrusion Detection System (IDS)

IDS adalah sistem yang **mengamati trafik atau log**, lalu **memberi peringatan (alert)** ketika menemukan pola yang dianggap berbahaya.

IDS ≠ Firewall

Firewall *mencegah*, IDS *mendeteksi*.

2. Signature-based vs Anomaly-based Detection

Signature-based Detection

Mencocokkan trafik dengan **pola serangan yang sudah dikenal**.

Contoh:

- Request HTTP mengandung */etc/passwd*
- Scan port berurutan
- Payload mengandung *shellcode*

Kelebihan

- Akurat
- Sedikit *false positive*

Kekurangan

- Tidak bisa mendeteksi serangan baru (*zero-day*)

Anomaly-based Detection

Mencari **perilaku yang menyimpang dari kebiasaan normal**.

Contoh:

- 1000 request dalam 1 detik
- Login gagal berulang
- Pola waktu akses tidak wajar

Kelebihan

- Bisa mendeteksi serangan baru
- Cocok untuk *unknown attack*

Kekurangan

- Banyak *false positive*
- Perlu baseline yang baik

Di dunia nyata, IDS modern menggabungkan dua pendekatan ini.

Tools yang Digunakan (Open Source)

Kita akan menggunakan **dua pendekatan nyata**:

1. **Snort** → contoh *signature-based IDS*
2. **Python Log Parser** → simulasi *anomaly-based IDS*

Semua dijalankan di:

- **Ubuntu Linux 24.04**
- **Tanpa software proprietary**

BAGIAN A — IDS Signature-Based dengan Snort

Step 1 — Instalasi Snort (Ubuntu 24.04)

```
sudo apt update
sudo apt install -y snort
```

Saat instalasi:

- Pilih **interface jaringan** (misalnya **eth0** atau **ens33**)
- **HOME_NET** → isi dengan jaringan lokal, contoh:

```
192.168.1.0/24
```

Verifikasi:

```
snort -V
```

Step 2 — Menjalankan Snort dalam Mode IDS

Gunakan mode console agar alert langsung terlihat:

```
sudo snort -A console -q -c /etc/snort/snort.conf -i eth0
```

Keterangan:

- **-A console** → tampilkan alert di terminal
- **-q** → quiet mode
- **-c** → file konfigurasi
- **-i** → interface jaringan

Step 3 — Membuat Rule IDS Sederhana (Signature)

Buka file rule lokal:

```
sudo nano /etc/snort/rules/local.rules
```

Tambahkan rule berikut:

```
alert icmp any any -> any any (msg:"ICMP Ping Detected";  
sid:1000001; rev:1;)
```

Reload Snort:

```
sudo systemctl restart snort
```

Step 4 — Simulasi Serangan Sederhana (Ping Flood)

Dari mesin lain (atau VM kedua):

```
ping -c 5 IP_TARGET
```

Hasil yang Diharapkan

Snort akan menampilkan alert:

```
[**] ICMP Ping Detected [**]
```

Interpretasi

- IDS mendeteksi trafik ICMP
- Bukan serangan fatal, tapi **indikasi reconnaissance**

Step 5 — Simulasi Serangan Port Scanning

Install **nmap**:

```
sudo apt install -y nmap
```

Lakukan scan:

```
nmap -sS IP_TARGET
```

Snort akan memunculkan alert scanning jika rule aktif.

Output Snort

- Alert di terminal
- Log di **/var/log/snort/alert**

Contoh log:

```
[1:1000001:1] ICMP Ping Detected
```

BAGIAN B — IDS Sederhana dengan Python (Anomaly-Based)

Sekarang kita **tidak pakai signature**, tapi **analisis perilaku log**.

Tujuan

Mendeteksi:

- Terlalu banyak request
- Pola akses tidak normal
- Brute-force login

Step 1 — Contoh Log yang Dianalisis

Misalnya log web server (**access.log**):

```
192.168.1.10 - - [20/Jan/2026:10:00:01] "GET /login HTTP/1.1" 200
192.168.1.10 - - [20/Jan/2026:10:00:02] "GET /login HTTP/1.1" 401
192.168.1.10 - - [20/Jan/2026:10:00:03] "GET /login HTTP/1.1" 401
192.168.1.10 - - [20/Jan/2026:10:00:04] "GET /login HTTP/1.1" 401
```

Step 2 — Script Python IDS Sederhana

Buat file `ids_log_parser.py`:

```
from collections import defaultdict

THRESHOLD = 3
ip_counter = defaultdict(int)

with open("access.log", "r") as log:
    for line in log:
        ip = line.split()[0]
        ip_counter[ip] += 1

for ip, count in ip_counter.items():
    if count > THRESHOLD:
        print(f"[ALERT] Suspicious activity from {ip} ({count} requests)")
```

Jalankan:

```
python3 ids_log_parser.py
```

Interpretasi Output

```
[ALERT] Suspicious activity from 192.168.1.10 (4 requests)
```

Maknanya:

- IP melakukan request berulang
- Bisa indikasi **brute-force**, **bot**, atau **scanner**

Diskusi Keamanan (Penting)

- Apakah ini **serangan pasti**? → **Tidak**
- Apakah ini **indikasi awal**? → **Ya**
- Apakah perlu investigasi lanjut? → **Wajib**

**IDS tidak mengambil keputusan, IDS memberi sinyal.
Manusialah yang memutuskan.**

Output Praktikum (Yang Dikumpulkan)

Mahasiswa wajib menyerahkan:

1. **Log alert Snort**
2. **Output script Python**
3. **Interpretasi singkat**, menjawab:
 - Serangan apa?
 - Bukti teknisnya?

- Dampak jika tidak terdeteksi?

Kesimpulan

- **Security bukan tentang tool**
- **IDS bukan alat ajaib**
- Yang penting adalah:
 - *Visibility*
 - *Log*
 - *Reasoning*

Jika kamu bisa menjelaskan **KENAPA** sebuah alert muncul, berarti kamu **sudah berpikir seperti security engineer**.

BAB 6: Perlindungan Data Pribadi

Pendahuluan: Dari Hukum ke Keyboard

Banyak orang mengira **Perlindungan Data Pribadi (PDP)** adalah urusan **hukum semata**—dokumen, pasal, tanda tangan, dan audit. Padahal, di dunia nyata, **pelanggaran data hampir selalu terjadi karena kegagalan teknis dan operasional**, bukan karena kurangnya pasal.

Di pertemuan ini, kita akan mengubah cara pandang tersebut. **UU PDP bukan dibaca untuk dihafal, tetapi untuk diterjemahkan menjadi keputusan teknis sehari-hari**: bagaimana menyimpan data, siapa yang boleh mengakses, berapa lama data disimpan, dan bagaimana data dihapus dengan benar.

**Regulasi adalah “KENAPA”,
Implementasi teknis adalah “BAGAIMANA”.**

Teori 1 — UU Perlindungan Data Pribadi Indonesia (UU PDP)

UU PDP Indonesia hadir untuk menjawab satu realitas penting: **data pribadi adalah aset, dan penyalahgunaannya berdampak langsung pada manusia nyata**—bukan sekadar sistem.

Dalam konteks teknis, UU PDP memperkenalkan perubahan besar:

- Data **bukan lagi sekadar field di database**
- Data melekat pada **hak subjek data**
- Setiap pemrosesan harus **punya alasan yang sah**

Secara sederhana, UU PDP menegaskan bahwa:

- **Tidak semua data boleh dikumpulkan**
- **Tidak semua data boleh disimpan selamanya**
- **Tidak semua orang boleh mengakses data**

Bagi engineer, admin sistem, dan developer, ini berarti:

“Bisa secara teknis” ≠ “Boleh secara hukum”

Prinsip Kunci UU PDP (Versi Teknis)

Tanpa masuk ke bahasa hukum yang berat, prinsip UU PDP bisa diterjemahkan menjadi aturan teknis berikut:

- **Purpose limitation**
Data hanya boleh diproses untuk tujuan yang jelas
→ jangan simpan data “siapa tahu nanti kepakai”

- **Data minimization**
Simpan **data yang benar-benar perlu saja**
→ kalau email cukup, jangan minta NIK
- **Security by design**
Keamanan **bukan tambahan**, tapi bagian dari desain awal
→ enkripsi, kontrol akses, logging sejak hari pertama
- **Accountability**
Harus bisa menjawab:
“Siapa mengakses data ini, kapan, dan untuk apa?”

Log adalah bukti kepatuhan. Tanpa log, kepatuhan tidak bisa dibuktikan.

Teori 2 — Data Lifecycle: Hidup, Digunakan, Lalu Dihapus

Salah satu kesalahan paling umum dalam sistem IT adalah **menganggap data itu statis**. Padahal, dalam perspektif PDP, **data punya siklus hidup (*data lifecycle*)**.

Tahap 1 — Data Collection (Pengumpulan)

Data pribadi lahir ketika:

- User mengisi formulir
- Mahasiswa mendaftar
- Pelanggan membuat akun

Di tahap ini, pertanyaan paling penting bukan:

“Data apa yang bisa kita ambil?”

melainkan:

“Data apa yang *boleh* kita ambil?”

Implikasi teknis:

- Form harus **minimal**
- Field sensitif harus **punya justifikasi**
- Tidak boleh ada *silent collection* (pengumpulan diam-diam)

Tahap 2 — Consent (Persetujuan)

Consent bukan sekadar checkbox.

Dalam UU PDP, consent harus:

- *Explicit* (jelas)
- *Informed* (pengguna tahu datanya dipakai untuk apa)
- *Revocable* (bisa ditarik kembali)

Secara teknis, ini berarti:

- Consent harus **tersimpan**
- Versi consent harus **tercatat**
- Perubahan consent harus **berdampak ke sistem**

Contoh nyata:

- Jika user mencabut izin email marketing
→ sistem **tidak boleh lagi mengirim email**, bukan cuma “niatnya berhenti”

Consent tanpa enforcement hanyalah formalitas.

Tahap 3 — Data Processing & Storage

Di tahap ini, data:

- Diproses
- Disimpan
- Diakses oleh sistem atau manusia

Di sinilah **keamanan teknis menjadi sangat krusial**:

- Enkripsi data sensitif
- Role-based access
- Segmentasi database

Pertanyaan reflektif penting:

- Apakah admin *benar-benar perlu* melihat data ini?
- Apakah data ini tersimpan dalam bentuk *plain text*?
- Jika database bocor, **apa dampaknya ke manusia?**

Tahap 4 — Data Sharing (Jika Ada)

Berbagi data adalah titik risiko tertinggi.

Setiap kali data keluar dari sistem:

- Risiko bertambah

- Kontrol berkurang
- Tanggung jawab tetap melekat

Implementasi teknis yang baik:

- API dengan scope terbatas
- Masking data
- Audit log untuk setiap akses eksternal

Jika data bocor dari vendor, publik tetap menyalahkan pemilik data.

Tahap 5 — Data Retention & Deletion

Ini tahap yang paling sering diabaikan.

Banyak sistem:

- Menyimpan data **selamanya**
- Tidak punya mekanisme penghapusan
- Mengandalkan “*nanti juga lupa*”

Dalam PDP:

- Data harus **dihapus ketika tujuan tercapai**
- Penghapusan harus **nyata**, bukan hanya “tidak ditampilkan”

Secara teknis:

- Delete ≠ hide
- Backup juga harus diperhitungkan
- Log penghapusan penting sebagai bukti kepatuhan

Data yang tidak perlu disimpan adalah risiko yang tidak perlu diambil.

Penutup: PDP adalah Soal Etika Teknis

Pada akhirnya, **Perlindungan Data Pribadi bukan tentang takut sanksi**, tetapi tentang **tanggung jawab profesional**.

Sebagai technical person:

- Kita membangun sistem yang mempengaruhi hidup orang lain
- Kesalahan desain bisa berdampak jangka panjang
- Keamanan adalah keputusan, bukan fitur

Engineer yang baik membuat sistem berjalan.

Engineer yang bertanggung jawab membuat sistem aman dan manusiawi.

PRAKTEK 6: Identifikasi & Klasifikasi Data Pribadi (PDP)

Tujuan

Pada praktek ini, mahasiswa **tidak hanya memahami PDP secara hukum**, tetapi **mampu membaca data secara teknis** dan menjawab pertanyaan penting:

“Data apa yang sedang kita pegang, seberapa sensitif, dan apa resikonya jika bocor?”

Setelah praktikum, mahasiswa mampu:

- **Mengidentifikasi data pribadi di dataset nyata**
- **Mengklasifikasikan tingkat sensitivitas data**
- **Menentukan kontrol pengamanan yang rasional**
- Menyusun **Data Classification Matrix** seperti yang digunakan di organisasi profesional

Konsep Inti (Wajib Dipahami)

Data Pribadi Umum vs Sensitif

Data Pribadi Umum

→ Data yang **mengidentifikasi seseorang**, tetapi **dampak kebocorannya relatif rendah**

Contoh:

- Nama
- Email
- Nomor telepon
- Alamat (tanpa detail spesifik)

Data Pribadi Sensitif

→ Data yang **berpotensi menyebabkan kerugian serius** bila bocor

Contoh:

- NIK / No KTP
- Data kesehatan
- Rekam biometrik
- Password, token, API key
- Data keuangan

Prinsip penting:

Semakin besar dampak kebocoran, semakin ketat kontrolnya.

Data Lifecycle (Siklus Hidup Data)

Setiap data **pasti melewati tahapan ini**:

1. **Collect** – dikumpulkan (form, API, upload)
2. **Store** – disimpan (database, file, spreadsheet)
3. **Process** – diproses (analitik, AI, laporan)
4. **Share** – dibagikan (email, API, integrasi)
5. **Archive** – diarsipkan
6. **Delete** – dihapus

Kesalahan umum:

Fokus hanya saat *collect*, lupa mengamankan *store & share*.

Tools yang Digunakan (100% Open Source)

Kebutuhan	Tools
Dataset	Dummy CSV
Analisis	Python 3 (default Ubuntu 24.04)
Spreadsheet	LibreOffice Calc
Enkripsi	GnuPG
OS	Ubuntu 24.04 LTS

Praktek Step-by-Step

STEP 1 — Siapkan Dataset Dummy

Buat file **data_mahasiswa.csv**

```
nama,email,no_hp,nik,tanggal_lahir,alamat,password
Andi Pratama,andi@email.com,08123456789,3174010101010001,
2000-01-01,Jakarta,password123
Siti Rahmawati,siti@email.com,08234567890,3274020202020002,
1999-05-12,Bandung,rahasia456
```

Catatan penting:

Dataset **dummy** ini menyerupai data nyata, tetapi **aman untuk praktikum**.

STEP 2 — Identifikasi Kolom Data Pribadi (Python)

Buat file **identify_pdp.py**

```
import pandas as pd

# Load dataset
df = pd.read_csv("data_mahasiswa.csv")

# Daftar pola data pribadi
pdp_patterns = {
    "nama": "Data Pribadi Umum",
    "email": "Data Pribadi Umum",
    "no_hp": "Data Pribadi Umum",
    "alamat": "Data Pribadi Umum",
    "nik": "Data Pribadi Sensitif",
    "tanggal_lahir": "Data Pribadi Sensitif",
    "password": "Data Sangat Sensitif"
}

print("Identifikasi Data Pribadi:\n")

for col in df.columns:
    kategori = pdp_patterns.get(col, "Bukan Data Pribadi")
    print(f"- {col}: {kategori}")
```

Jalankan:

```
python3 identify_pdp.py
```

Output yang diharapkan:

```
- nama: Data Pribadi Umum
- email: Data Pribadi Umum
- no_hp: Data Pribadi Umum
- nik: Data Pribadi Sensitif
- tanggal_lahir: Data Pribadi Sensitif
- alamat: Data Pribadi Umum
- password: Data Sangat Sensitif
```

STEP 3 — Klasifikasi Tingkat Sensitivitas

Gunakan **3 level sederhana (praktis & realistis)**:

Level	Makna
Low	Dampak kebocoran kecil
Medium	Risiko reputasi / privasi
High	Risiko hukum, finansial, kriminal

Tambahkan ke script:

```
sensitivity_level = {
    "Data Pribadi Umum": "Medium",
    "Data Pribadi Sensitif": "High",
    "Data Sangat Sensitif": "Critical"
}

print("\nKlasifikasi Sensitivitas:\n")
for col in df.columns:
    kategori = pdp_patterns.get(col, "Bukan Data Pribadi")
    level = sensitivity_level.get(kategori, "None")
    print(f"- {col}: {level}")
```

STEP 4 — Tentukan Kontrol Pengamanan

Prinsip Dasar Kontrol

Level	Kontrol Minimal
Medium	Akses terbatas
High	Enkripsi
Critical	Enkripsi + hashing + audit

Contoh Implementasi: Enkripsi Kolom Sensitif

Enkripsi file dengan GnuPG

```
gpg --symmetric --cipher-algo AES256 data_mahasiswa.csv
```

Output:

```
data_mahasiswa.csv.gpg
```

Makna:

Data tidak bisa dibaca tanpa passphrase → **compliance by design**

STEP 5 — Buat Data Classification Matrix

Template (Spreadsheet / Markdown)

Kolom	Jenis Data	Sensitivitas	Risiko Kebocoran	Kontrol
nama	Pribadi Umum	Medium	Pelanggaran privasi	Access control
email	Pribadi Umum	Medium	Spam / phishing	Masking
no_hp	Pribadi Umum	Medium	Social engineering	Rate limit
nik	Sensitif	High	Identitas palsu	Encryption
tanggal_lahir	Sensitif	High	Profiling	Encryption
password	Sangat Sensitif	Critical	Account takeover	Hashing + Salt

Iniilah dokumen yang dinilai dosen.

Output Praktikum (Wajib Dikumpulkan)

Mahasiswa mengumpulkan:

1. ☒ Dataset dummy
2. ☒ Script Python identifikasi PDP
3. ☒ File terenkripsi (**.gpg**)
4. ☒ **Data Classification Matrix**
5. ☒ Analisis singkat (½–1 halaman):
 - o Kolom paling berisiko
 - o Alasan klasifikasi
 - o Dampak kebocoran

Kesalahan Umum (Dan Harus Dihindari)

- ☒ Menganggap semua data sama
- ☒ Menyimpan password plaintext
- ☒ Tidak tahu data disimpan di mana
- ☒ Tidak bisa menjelaskan *kenapa* data harus dienkripsi

Skill Nyata yang Didapat

Setelah praktikum ini, mahasiswa:

- **Bisa membaca database dengan “kacamata keamanan”**
- **Siap terlibat audit PDP**
- **Siap kerja di sistem real (kampus, startup, gov, AI system)**

BAB 7: Audit Kebocoran Data Pribadi

Tujuan

Setelah mempelajari bab ini, pembaca diharapkan mampu:

- **Memahami siklus kebocoran data (*data breach lifecycle*)** dari awal sampai pemulihan
- **Mengenali tanda-tanda awal insiden** sebelum dampaknya membesar
- **Menjalankan respons insiden dasar (*incident response*)** secara terstruktur
- **Berpikir defensif:** bagaimana *mencegah pengulangan* setelah insiden selesai

Teori Inti

Data Breach Lifecycle

Kebocoran data **tidak terjadi tiba-tiba**. Hampir selalu ada *jejak* dan *tahapan* yang bisa dilacak. Memahami siklus ini membuat kita **tidak panik**, dan tahu **apa yang harus dilakukan di setiap fase**.

Tahapan umum kebocoran data:

1. **Reconnaissance**
Penyerang melakukan *pengintaian*: memetakan sistem, IP, API, endpoint, atau bahkan kebiasaan manusia.
👉 *Sering tidak disadari karena terlihat “normal”*
2. **Initial Access**
Akses awal diperoleh melalui:
 - *Phishing*
 - Password lemah
 - Credential reuse
 - Vulnerability yang belum ditambal**Ini adalah titik kritis pertama.**
3. **Privilege Escalation & Lateral Movement**
Setelah masuk, penyerang **tidak langsung mencuri data**. Mereka:
 - Menaikkan hak akses
 - Bergerak ke sistem lain*Tujuannya: mencari data paling bernilai.*
4. **Data Exfiltration**
Data mulai **keluar dari sistem**:
 - Diunduh
 - Dikirim ke server luar
 - Disamarkan sebagai trafik normal **Inilah momen kebocoran sebenarnya.**

5. Covering Tracks

Log dihapus, jejak disamarkan, akun backdoor ditinggalkan.

Jika fase ini berhasil, kebocoran bisa baru ketahuan berbulan-bulan kemudian.

6. Discovery & Disclosure

Insiden akhirnya diketahui:

- Dari log
- Dari user
- Dari pihak ketiga

Semakin lambat diketahui, semakin besar dampaknya.

Pesan penting:

Audit bukan hanya mencari “siapa yang salah”, tetapi memahami “apa yang gagal”.

Incident Response Basics

Incident response adalah **proses terstruktur**, bukan reaksi panik. Tim yang baik **tidak improvisasi**, tetapi mengikuti langkah yang jelas.

Tahapan Incident Response

1. Preparation

Sebelum insiden terjadi:

- Log aktif
 - Backup rutin
 - Tim & peran jelas
- 👉 *Audit yang baik dimulai sebelum insiden.*

2. Identification

Menjawab pertanyaan kritis:

- Apa yang terjadi?
- Sistem mana terdampak?
- Data apa yang mungkin bocor?

3. Containment

Tujuan utama: **hentikan kebocoran secepat mungkin**

- Isolasi server
 - Disable akun
 - Cabut akses API
- ⚠️ *Containment bukan memperbaiki, tapi menghentikan pendarahan.*

4. Eradication

Menghilangkan akar masalah:

- Patch vulnerability
- Reset credential
- Hapus backdoor

5. Recovery

Sistem dikembalikan:

- Restore dari backup bersih
- Monitoring ekstra
- Uji ulang keamanan

6. Lessons Learned

Tahap paling sering dilewatkan, padahal **paling penting**:

- Apa penyebab utamanya?
- Kontrol apa yang gagal?
- Apa yang harus diubah?

Prinsip emas:

Tidak ada sistem yang aman selamanya, tapi sistem yang belajar akan semakin kuat.

Pendekatan Audit: Investigatif & Defensif

Audit kebocoran data **bukan sekadar checklist**, melainkan *cerita kronologis*.

Audit investigatif fokus pada:

- *What happened*
- *How it happened*
- *Which data affected*

Audit defensif fokus pada:

- *How to prevent recurrence*
- *Which control must be strengthened*
- *Which assumption was wrong*

Gabungan keduanya membuat audit **bernilai strategis**, bukan hanya administratif.

Kesalahan Umum yang Harus Dihindari

- ✗ Fokus hanya pada *tool*, lupa proses
- ✗ Menyalahkan individu, bukan sistem
- ✗ Tidak mencatat kronologi
- ✗ Menganggap insiden selesai setelah sistem hidup kembali

Insiden belum selesai sampai pelajarannya diambil.

Mengapa Bab Ini Penting?

Bab ini melatih:

- **Mental incident handler**
- **Cara berpikir sistemik**
- **Kepekaan terhadap data pribadi**
- **Keputusan berbasis risiko, bukan emosi**

Ini adalah **jembatan** antara teori keamanan dan **realitas insiden di dunia nyata**—baik di kampus, startup, pemerintahan, maupun sistem berbasis AI.

PRAKTEK 7: Audit Kebocoran Data

Tujuan

Pada praktek ini, mahasiswa **berperan sebagai tim audit insiden**. Fokusnya bukan sekadar “mencari kesalahan”, tetapi:

- **Menelusuri jejak kebocoran data (*digital footprint*)**
- **Memahami bagaimana insiden terjadi**
- **Menyusun laporan insiden yang rasional & defensible**
- Melatih *incident handling mindset*, bukan sekadar teknis alat

Audit kebocoran = membaca cerita yang ditinggalkan oleh data

Konsep Inti

Incident Handling

Incident handling adalah **proses terstruktur** untuk:

- mendeteksi,
- menganalisis,
- mengendalikan,
- dan mendokumentasikan insiden keamanan.

Audit dilakukan **setelah atau saat insiden berlangsung**, dengan tujuan:

- memahami dampak,
- menentukan penyebab,
- mencegah pengulangan.

Digital Footprint

Digital footprint adalah **jejak data yang tertinggal di dunia digital**, baik sengaja maupun tidak.

Contoh nyata:

- File CSV terbuka di web server
- Backup database tanpa enkripsi
- Email & nomor HP di repository publik
- Log aplikasi yang berisi token / password

Prinsip penting:

Jika data bisa diakses tanpa otorisasi, maka data dianggap bocor.

Tools (100% Open Source)

Kebutuhan	Tools
OS	Ubuntu 24.04 LTS
OSINT simulasi	File HTML & CSV dummy
Analisis	Python 3
Parsing	requests, beautifulsoup4
Enkripsi bukti	GnuPG
Dokumentasi	Markdown / PDF

Install library Python:

```
sudo apt update
sudo apt install -y python3 python3-pip gnupg
pip3 install requests beautifulsoup4
```

Skenario Praktikum (Realistis)

Sebuah **file mahasiswa** ditemukan dapat diakses publik melalui URL internal. Diduga berisi **data pribadi**.

Tugas Anda:

1. Memverifikasi kebocoran
2. Mengidentifikasi data yang terekspos
3. Menilai dampak
4. Menyusun **Incident Report**

STEP 1 — Siapkan Data Kebocoran (Simulasi)

Buat file **leaked_data.html**

```
<html>
<body>
<h1>Backup Mahasiswa</h1>
<table>
<tr><th>Nama</th><th>Email</th><th>No HP</th><th>NIK</th></tr>
<tr><td>Andi</td><td>andi@email.com</td><td>08123456789</td><td>
3174010101010001</td></tr>
<tr><td>Siti</td><td>siti@email.com</td><td>08234567890</td><td>
3274020202020002</td></tr>
</table>
</body>
</html>
```

Ini merepresentasikan:

- backup web
- hasil scraping
- halaman admin bocor

STEP 2 — OSINT Simulasi: Deteksi Data yang Terekspos

Buat file **audit_scraper.py**

```
from bs4 import BeautifulSoup

# Load file simulasi
with open("leaked_data.html", "r") as f:
    soup = BeautifulSoup(f, "html.parser")

rows = soup.find_all("tr")

print("Data yang terdeteksi bocor:\n")

for row in rows[1:]:
    cols = row.find_all("td")
    nama = cols[0].text
    email = cols[1].text
    hp = cols[2].text
    nik = cols[3].text

    print(f>Nama: {nama}")
    print(f>Email: {email}")
    print(f>No HP: {hp}")
    print(f>NIK: {nik}")
    print("-" * 30)
```

Jalankan:

```
python3 audit_scraper.py
```

Makna teknis:

- Audit tidak perlu *hacking*
- Cukup membaca data yang **sudah terbuka**

STEP 3 — Klasifikasi Dampak Kebocoran

Tambahkan analisis otomatis:

```
print("\nAnalisis Dampak:\n")

impact = {
    "Nama": "Medium",
    "Email": "Medium",
    "No HP": "Medium",
    "NIK": "High"
}

for k, v in impact.items():
    print(f"{k}: Risiko {v}")
```

Interpretasi audit:

- Email + HP → *social engineering*
- NIK → *identity fraud*
- Kombinasi data → **risiko eskalasi**

STEP 4 — Amankan Bukti Audit (Chain of Custody)

Bukti audit **HARUS dilindungi**, bukan dibiarkan terbuka.

Enkripsi bukti dengan GnuPG

```
gpg --symmetric --cipher-algo AES256 leaked_data.html
```

Hasil:

```
leaked_data.html.gpg
```

Prinsip audit profesional:

Bukti insiden harus **lebih aman** daripada data aslinya.

STEP 5 — Susun Incident Report (Template Nyata)

Incident Report (Ringkas & Profesional)

1. Identitas Insiden

- Tanggal temuan:
- Sistem terdampak:
- Pelapor:

2. Kronologi Singkat

- File backup ditemukan dapat diakses publik
- Berisi data pribadi mahasiswa
- Tidak ada autentikasi

3. Jenis Data Terekspos

Data	Kategori	Risiko
Nama	Pribadi Umum	Medium
Email	Pribadi Umum	Medium
No HP	Pribadi Umum	Medium
NIK	Sensitif	High

4. Dampak Potensial

- Phishing
- Identity fraud
- Pelanggaran UU PDP

5. Tindakan Darurat

- Akses file ditutup
- Bukti dienkripsi
- Audit log dilakukan

6. Rekomendasi

- Enkripsi backup
- Review akses web server
- Data minimization
- Audit berkala

Output yang Dikumpulkan

Mahasiswa menyerahkan:

1. ☒ Script Python audit
2. ☒ Bukti terenkripsi (.gpg)
3. ☒ Incident Report (PDF/Markdown)
4. ☒ Analisis reflektif ½ halaman:
 - Apa akar masalah?
 - Kontrol apa yang gagal?
 - Apa yang harus diubah?

Kesalahan Umum (Sering Terjadi)

- ☒ Menghapus bukti sebelum audit
- ☒ Fokus ke “siapa yang salah”
- ☒ Tidak mendokumentasikan kronologi
- ☒ Menganggap insiden selesai setelah file dihapus

Insiden selesai jika sistemnya membaik, bukan hanya sunyi.

Skill Nyata yang Didapat

Setelah praktikum ini, mahasiswa:

- **Berpikir seperti auditor & incident responder**
- **Mampu membaca kebocoran tanpa panik**
- **Siap menghadapi audit nyata (kampus, startup, gov, AI system)**

BAB 8: Keamanan AI sebagai Sistem (AI sebagai Attack Surface)

Pendahuluan: AI Tidak Berdiri Sendiri

Dalam banyak diskusi populer, kecerdasan buatan sering dipersepsikan sebagai “mesin pintar” yang berdiri sendiri. Padahal, dalam praktik nyata, **AI selalu merupakan bagian dari sistem yang lebih besar**—terdiri dari data, pipeline pemrosesan, model, infrastruktur, API, pengguna, serta proses operasional manusia.

Dari sudut pandang keamanan, hal ini berarti satu kesimpulan penting:

AI bukan hanya objek yang dilindungi, tetapi juga bagian dari permukaan serangan sistem.

Bab ini mengajak pembaca untuk melihat AI **sebagai sistem sosio-teknis**, bukan sekadar algoritma. Fokus utama bab ini bukan pada serangan spesifik terhadap model (yang dibahas di BAB 9), melainkan pada **bagaimana AI memperluas attack surface dan mengubah peta risiko keamanan informasi**.

AI sebagai Attack Surface Baru

Apa itu Attack Surface?

Attack surface adalah seluruh titik pada sistem yang dapat:

- diakses,
- dipengaruhi,
- atau dieksploitasi oleh pihak luar.

Pada sistem tradisional, attack surface biasanya terbatas pada:

- jaringan,
- aplikasi,
- database,
- dan akun pengguna.

Namun ketika AI ditambahkan ke dalam sistem, **attack surface meningkat secara signifikan**, karena AI memperkenalkan komponen baru seperti:

- dataset pelatihan,
- model terlatih,
- pipeline preprocessing,
- endpoint inferensi,
- serta keputusan otomatis berbasis probabilitas.

Perubahan Fundamental yang Dibawa AI

AI mengubah karakter sistem keamanan dalam beberapa aspek penting:

1. **Dari deterministik ke probabilistik**
Sistem tidak lagi selalu menghasilkan output yang pasti untuk input tertentu.
2. **Dari rule-based ke data-driven**
Perilaku sistem ditentukan oleh data, bukan hanya oleh aturan eksplisit.
3. **Dari statis ke adaptif**
Model dapat diperbarui, di-*retrain*, atau di-*fine-tune* secara berkala.

Perubahan ini membawa konsekuensi serius terhadap:

- auditabilitas,
- prediktabilitas,
- dan kepercayaan terhadap sistem.

Komponen Sistem AI yang Perlu Diamankan

Untuk memahami risiko secara utuh, sistem AI perlu diurai menjadi beberapa komponen keamanan.

1. Data (Training, Validation, dan Input)

Data adalah fondasi utama AI. Tanpa data yang aman dan terpercaya, **tidak ada AI yang aman**.

Risiko utama:

- kebocoran data pelatihan (PII),
- manipulasi data input,
- reuse data tanpa consent,
- bias tersembunyi dalam dataset.

Prinsip penting:

Keamanan AI dimulai dari keamanan data, bukan dari model.

2. Model AI

Model bukan sekadar file statis, melainkan **representasi pengetahuan sistem**.

Risiko utama:

- pencurian model,
- reverse engineering,
- model leakage,
- penggunaan model di luar konteks awal.

Dalam konteks CIA Triad:

- *Confidentiality*: model dapat diekstrak,
- *Integrity*: model dapat dimodifikasi,
- *Availability*: model dapat disabotase atau dilumpuhkan.

3. Pipeline & Infrastruktur

AI jarang berjalan di satu mesin tertutup. Ia biasanya terintegrasi dengan:

- server,
- container,
- GPU,
- API,
- dan sistem monitoring.

Risiko utama:

- salah konfigurasi environment,
- akses tidak sah ke pipeline training,
- dependency berbahaya,
- kebocoran artefak model.

4. Output & Keputusan AI

Berbeda dengan sistem tradisional, **output AI sering dianggap “otoritatif”** oleh pengguna.

Risiko utama:

- overtrust terhadap AI,
- keputusan otomatis tanpa validasi manusia,
- eksploitasi output untuk manipulasi sosial atau ekonomi.

AI yang “salah tapi meyakinkan” sering kali **lebih berbahaya** daripada sistem yang jelas-jelas error.

Ancaman Keamanan Khas Sistem AI

Bab ini menekankan bahwa ancaman terhadap AI **tidak selalu berbentuk serangan teknis eksplisit**.

Beberapa ancaman khas antara lain:

- penyalahgunaan AI oleh pengguna internal,
- penggunaan AI di luar desain awal (*misuse*),
- akumulasi kesalahan kecil yang berdampak sistemik,
- ketergantungan berlebihan pada rekomendasi AI.

Ancaman-ancaman ini sering:

- tidak tercatat sebagai “incident”,
- sulit dideteksi secara otomatis,
- tetapi berdampak jangka panjang.

AI dan Perubahan Paradigma Keamanan

Dari “Secure the System” ke “Secure the Decision”

Pada sistem AI, keamanan tidak lagi hanya soal:

- melindungi server,
- menutup port,
- atau mengenkripsi database.

Keamanan juga mencakup:

- **keamanan keputusan,**
- **keamanan proses belajar,**
- **keamanan asumsi data.**

Kesalahan pada AI sering kali:

- sah secara teknis,
- tetapi keliru secara konteks,
- dan berbahaya secara sosial.

Hubungan dengan CIA Triad

AI memperluas makna CIA Triad:

- **Confidentiality**
Tidak hanya data mentah, tetapi juga *knowledge* dalam model.
- **Integrity**
Tidak hanya file, tetapi juga proses belajar dan distribusi data.
- **Availability**
Tidak hanya uptime sistem, tetapi juga keandalan keputusan.

Dengan kata lain, **CIA Triad tetap relevan**, tetapi perlu diterapkan secara lebih luas dan kontekstual pada sistem AI.

Mindset yang Harus Dibangun

Bab ini bertujuan membentuk cara berpikir berikut:

- AI **tidak otomatis aman** hanya karena “pintar”.
- AI **bukan black box yang kebal kritik**.
- Keamanan AI adalah **tanggung jawab desain**, bukan sekadar patch.
- Setiap sistem AI **harus diasumsikan dapat gagal**.

AI yang aman bukan AI yang tidak pernah salah, tetapi AI yang kesalahannya dapat dipahami, dibatasi, dan diperbaiki.

Transisi ke Bab Selanjutnya

Jika bab ini membahas **AI sebagai sistem dan permukaan serangan**, maka **BAB 9** akan membawa pembaca masuk ke level yang lebih teknis:

- bagaimana AI diserang secara langsung,
- bagaimana data dan model dimanipulasi,
- serta bagaimana eksperimen serangan AI dapat dilakukan secara etis dan terkontrol.

Bab selanjutnya tidak hanya menjelaskan *apa yang bisa salah*, tetapi juga **mengapa AI sering gagal dengan cara yang tidak disadari**.

PRAKTEK 8: Analisis Keamanan Sistem AI (Pre-Attack Lab)

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa mampu:

- Memahami **AI sebagai bagian dari sistem**, bukan sekadar model
- Mengidentifikasi **attack surface pada sistem AI**
- Menganalisis risiko keamanan AI menggunakan **CIA Triad**
- Menyusun **peta risiko awal** sebelum melakukan eksperimen serangan AI
- Mengembangkan **mindset security-first** sebelum menyentuh teknik ofensif

Fokus utama praktikum ini adalah cara berpikir, bukan koding.

Konsep Inti

1. **AI sebagai Sistem Sosio-Teknis**
2. **Attack Surface**
3. **CIA Triad (Confidentiality, Integrity, Availability)**
4. **Asset – Threat – Impact**
5. **Risk Awareness sebelum Attack**

Tools yang Digunakan (Ringan & Open Source)

Tidak ada tools kompleks. Gunakan:

- Kertas / Whiteboard (diskusi awal)
- draw.io / Excalidraw → diagram sistem AI
- Spreadsheet (LibreOffice / Google Sheets)
- (Opsional) Python + Jupyter Notebook → hanya untuk eksplorasi data sederhana

 **Tidak ada eksploitasi, fuzzing, atau serangan di modul ini.**

Skenario Praktikum (Wajib Dipilih 1)

Mahasiswa memilih **satu sistem AI fiktif tapi realistis**:

Pilihan Skenario:

1. **AI Deteksi Phishing Email**
2. **AI Penilaian Risiko Kredit UMKM**
3. **AI Deteksi Anomali Log Server**
4. **AI Chatbot Layanan Publik**
5. **AI Audit Data Pribadi (PII Scanner)**

Langkah Praktikum (Step-by-Step)

STEP 1 — Memahami Sistem AI Secara Utuh

Mahasiswa **tidak langsung bicara model**, tetapi menjawab:

- Siapa pengguna sistem?
- Data apa yang masuk?
- Keputusan apa yang dihasilkan AI?
- Dampak apa jika AI salah?

Output singkat (1 paragraf):

“Sistem AI ini digunakan untuk ... dan berdampak pada ...”

STEP 2 — Identifikasi Komponen Sistem AI

Mahasiswa memetakan sistem AI menjadi komponen berikut:

1. **Data**
 - data training
 - data input (inferensi)
2. **Model**
 - file model
 - versi model
3. **Pipeline**
 - preprocessing
 - training / inference
4. **Infrastruktur**
 - server / VM / container
5. **Output**
 - prediksi
 - rekomendasi
 - keputusan otomatis
6. **Manusia**
 - user
 - admin
 - operator

Output: daftar komponen sistem AI.

STEP 3 — Membuat Diagram AI sebagai Sistem

Gunakan draw.io / Excalidraw.

Diagram minimal harus memuat:

- alur data → model → output
- interaksi manusia
- titik input & output

Contoh sederhana (teks):

```
User → Input Data → Preprocessing → AI Model → Output
                        ↑
                Training Dataset
```

Output: 1 diagram sistem AI.

STEP 4 — Identifikasi Attack Surface

Untuk setiap komponen, mahasiswa menjawab:

- Bagian mana yang bisa diakses?
- Bagian mana yang bisa disalahgunakan?
- Bagian mana yang paling sering dipercaya tanpa dicek?

Contoh:

- Input bebas dari user → **attack surface**
- Dataset training → **attack surface**
- API inference → **attack surface**

Output: tabel sederhana Attack Surface.

STEP 5 — Pemetaan CIA Triad pada Sistem AI

Mahasiswa menganalisis **dampak kegagalan AI** terhadap CIA.

Contoh tabel:

Komponen	Confidentiality	Integrity	Availability	Dampak
Dataset training	✗	✗	✓	Model bias
Model AI	✗	✗	✗	Keputusan salah
API inference	✓	✗	✗	Layanan lumpuh

Output: Tabel CIA Impact.

STEP 6 — Analisis Risiko Awal (Tanpa Angka Rumit)

Mahasiswa memilih **3 risiko terbesar**, lalu menjawab:

- Mengapa risiko ini berbahaya?
- Siapa yang terdampak?
- Apakah risikonya teknis, etis, atau sosial?

Output: 3 paragraf analisis risiko.

STEP 7 — Refleksi Pra-Serangan (WAJIB)

Mahasiswa menulis refleksi singkat:

- Apa asumsi paling berbahaya dalam sistem AI ini?
- Bagian mana yang paling sering dianggap “aman” padahal tidak?
- Apa yang bisa salah tanpa ada “hacker”?

Output: 1 halaman refleksi.

Output yang Wajib Dikumpulkan

1. Diagram sistem AI
2. Tabel attack surface
3. Tabel CIA impact
4. Analisis 3 risiko utama
5. Refleksi pra-serangan

Evaluasi Penilaian

Aspek	Bobot
Pemahaman sistem AI	25%
Ketepatan attack surface	25%
Analisis CIA	20%
Kualitas refleksi	20%
Kerapihan & kejelasan	10%

Kesalahan Umum (Wajib Dihindari)

- Langsung bicara algoritma tanpa memahami sistem
- Menganggap AI selalu benar
- Fokus ke hacker, lupa human error
- Mengira “belum diserang” berarti “aman”

Insight Penting

**Sebelum AI diserang,
AI biasanya sudah berbahaya karena asumsi yang salah.**

Praktikum ini **menjadi fondasi mental** agar pada **BAB selanjutnya**, mahasiswa:

- tidak sekadar “menyerang model”,
- tetapi memahami **mengapa serangan itu relevan dan berbahaya**.

BAB 9: Keamanan AI (AI sebagai Target Serangan)

Fokus: AI tidak otomatis aman

Pada tahap awal belajar AI, banyak orang beranggapan bahwa **model AI adalah “otak pintar” yang kebal terhadap serangan**. Padahal, dalam dunia nyata, **AI justru menjadi target yang sangat menarik bagi penyerang**. Alasannya sederhana: **AI membuat keputusan**, dan siapa pun yang bisa memanipulasi keputusan tersebut akan mendapatkan keuntungan besar — finansial, politik, maupun strategis.

AI berbeda dengan software tradisional. Jika aplikasi biasa rusak, dampaknya mungkin terbatas. Namun **jika AI rusak atau dimanipulasi**, maka **kesalahan akan direplikasi secara otomatis dan masif**. Inilah mengapa **keamanan AI (AI Security)** menjadi isu kritis dalam sistem modern.

Mengubah Cara Pandang: AI sebagai Attack Surface

Dalam keamanan klasik, kita mengenal *attack surface* seperti port jaringan, API, atau database. Pada sistem berbasis AI, **model dan data itu sendiri adalah attack surface baru**.

Beberapa fakta penting yang perlu dipahami sejak awal:

- **Model AI belajar dari data → data bisa diserang**
- **Model AI membuat prediksi → prediksi bisa dimanipulasi**
- **Model AI bisa di-query → informasi internal bisa bocor**

AI bukan hanya alat pertahanan, tetapi juga objek yang harus dilindungi.

Adversarial Attack — Menipu AI Tanpa Merusak Sistem

Apa itu Adversarial Attack?

Adversarial attack adalah teknik menyerang AI dengan **memberikan input yang tampak normal bagi manusia, tetapi menyebabkan AI salah mengambil keputusan**.

Yang menarik (dan menakutkan):

- Tidak perlu *hack server*
- Tidak perlu *exploit OS*
- Cukup **manipulasi input**

Contoh nyata:

- Gambar *stop sign* yang ditemplei stiker kecil → AI mobil otonom mengira itu *speed limit*
- Email yang tampak biasa → lolos dari *spam filter*
- Prompt teks yang dirancang khusus → membuat *chatbot* melanggar aturan

Mengapa Ini Berbahaya?

Karena:

- **Manusia tidak menyadari ada serangan**
- **Sistem tetap berjalan normal**
- **Kesalahan terjadi di level keputusan**

AI bisa “salah dengan percaya diri tinggi”, dan itulah masalah terbesarnya.

Cara Kerja Sederhana (Intuisi)

AI belajar pola statistik. *Adversarial input*:

- Tidak melanggar format
- Tidak terlihat aneh
- Tetapi **menggeser distribusi data sedikit demi sedikit**

Hasilnya:

- Model “tersesat” secara matematis
- Tapi terlihat “baik-baik saja” secara kasat mata

Data Poisoning — Meracuni AI dari Dalam

Apa itu Data Poisoning?

Data poisoning adalah serangan dengan **menyusupkan data yang salah, bias, atau berbahaya ke dalam dataset pelatihan AI**.

Jika *adversarial attack* menyerang saat AI digunakan, maka **data poisoning menyerang saat AI dilatih**.

Analogi sederhana:

“Kalau sejak kecil seseorang diajari informasi salah, maka ia akan membuat keputusan salah seumur hidup.”

Contoh Kasus Nyata

- Dataset wajah berisi label keliru → sistem *face recognition* salah identifikasi
- Data transaksi dimanipulasi → model deteksi fraud jadi tidak sensitif

- Data opini publik dimasukkan bot → AI analisis sentimen jadi bias

Yang membuat ini berbahaya:

- **Kesalahan tertanam permanen**
- **Sulit dideteksi setelah model jadi**
- **Validasi sering kali lolos karena data terlihat “normal”**

Titik Lemah yang Sering Terjadi

- Dataset dikumpulkan otomatis (*scraping*)
- Tidak ada verifikasi sumber
- Tidak ada *data lineage*
- Tidak ada audit dataset

AI sekuat data yang melahirkannya.

Model Leakage — Ketika Model “Bocor” Tanpa Disadari

Apa itu Model Leakage?

Model leakage terjadi ketika **informasi internal model atau data sensitif bisa ditebak melalui interaksi dengan model tersebut.**

Penyerang tidak perlu akses ke file model. Cukup:

- Mengirim banyak query
- Mengamati output
- Menyimpulkan pola internal

Bentuk-Bentuk Kebocoran Model

- **Model inversion**
Menebak data training (misalnya wajah, teks, atau atribut pribadi)
- **Membership inference**
Menentukan apakah suatu data *pernah digunakan* untuk melatih model
- **Model extraction**
Meniru model dengan mengamati input–output (mencuri *intellectual property*)

Mengapa Ini Masalah Besar?

Karena:

- **Bisa melanggar privasi**
- **Bisa melanggar UU Perlindungan Data**
- **Bisa mencuri nilai bisnis model AI**

AI bukan hanya teknologi — **ia adalah aset.**

Refleksi Penting untuk Mahasiswa

Beberapa *takeaway* kunci yang harus benar-benar dipahami:

- **AI tidak netral, tidak kebal, dan tidak sakral**
- **Keamanan AI dimulai dari desain, bukan setelah insiden**
- **Data adalah titik lemah terbesar AI**
- **Input yang “halus” bisa lebih berbahaya daripada malware**

Mindset yang Ingin Dibangun

Di akhir pertemuan ini, mahasiswa diharapkan memiliki pola pikir:

“Jika saya membangun AI, saya juga harus memikirkan bagaimana AI itu bisa diserang.”

Karena di dunia nyata:

- **Penyerang selalu lebih kreatif**
- **AI tidak pernah sepenuhnya benar**
- **Keamanan adalah proses, bukan fitur**

PRAKTEK 9: Serangan terhadap AI

Fokus modul ini adalah **membongkar asumsi “AI selalu benar”**. Mahasiswa diajak *hands-on* melihat sendiri bagaimana **model AI bisa ditipu, dimanipulasi, dan disesatkan**, bahkan dengan perubahan kecil yang **tidak terlihat oleh manusia**.

Seluruh praktik menggunakan **Linux Ubuntu 24.04, Python**, dan **open-source software**. Tidak ada *black box*, tidak ada proprietary tools.

Tujuan

Setelah menyelesaikan modul ini, mahasiswa mampu:

- **Memahami kelemahan fundamental sistem AI**
- Menjelaskan konsep *adversarial example* dan *data poisoning*
- **Mensimulasikan serangan nyata terhadap model AI**
- Menganalisis perubahan output akibat manipulasi input/data
- Menyadari bahwa **AI adalah sistem statistik, bukan sistem kebenaran**

Pesan kunci:

AI bisa salah — dan kesalahan itu bisa direayasa.

Konsep Inti (Fundamental yang Wajib Dipahami)

Adversarial Example

Adversarial example adalah **input yang dimodifikasi sangat kecil**, sering kali **tidak terlihat oleh manusia**, tetapi **mengubah prediksi AI secara drastis**.

Contoh dunia nyata:

- Gambar rambu **STOP** → AI membaca **Speed Limit**
- Email normal → diklasifikasikan sebagai **spam**
- Teks biasa → lolos sensor moderasi

👉 **Masalahnya bukan bug**, tetapi **cara AI belajar dari pola statistik**.

Data Poisoning

Data poisoning adalah serangan **sebelum AI digunakan**, yaitu dengan **merusak data latih** (*training data*).

Contoh nyata:

- Dataset wajah diberi label salah
- Review palsu dimasukkan ke data rekomendasi
- Data sensor dimanipulasi sebelum pelatihan

👉 Jika data rusak, **model secanggih apa pun akan rusak.**

Tools yang Digunakan

Semua tools bersifat **open source**:

- **Ubuntu Linux 24.04**
- **Python 3.12**
- **scikit-learn**
- **NumPy**
- **Jupyter Notebook**
- (opsional) **Matplotlib** untuk visualisasi

Praktikum Lengkap (Step-by-Step)

Bagian A — Setup Lingkungan

```
sudo apt update
sudo apt install python3 python3-pip python3-venv -y
```

Buat *virtual environment*:

```
python3 -m venv ai-attack-env
source ai-attack-env/bin/activate
```

Install library:

```
pip install numpy scikit-learn matplotlib jupyter
```

Bagian B — Train Model AI Sederhana

Kita mulai dari **model yang “benar” dan “bersih”**.

Dataset & Model

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load dataset
data = load_iris()
X, y = data.data, data.target

# Split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

```
# Train model
model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)

# Evaluate
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

Catatan Penting:

Model ini *tidak salah*. Ia bekerja sesuai statistik.

Bagian C — Adversarial Example (Serangan Input)

Tujuan

Membuktikan bahwa **perubahan kecil bisa menipu AI**.

Ambil Satu Data Uji

```
import numpy as np

sample = X_test[0]
label = y_test[0]

print("Original prediction:", model.predict([sample]))
print("True label:", label)
```

Manipulasi Input Secara Halus

```
# Tambahkan noise kecil
epsilon = 0.1
adversarial_sample = sample + epsilon * np.sign(sample)

print("Adversarial prediction:",
      model.predict([adversarial_sample]))
```

Yang terjadi:

- Secara visual: **data hampir sama**
- Secara matematis: **AI membaca berbeda**

Diskusi Kritis

- Apakah manusia bisa melihat perbedaannya? ❌
- Apakah AI berubah keputusan? ✅
- Apakah ini bug? ❌
- Apakah ini kelemahan desain AI? ✅

Bagian D — Data Poisoning (Serangan Data Training)

Tujuan

Menunjukkan bahwa **AI bisa “dibohongi” sejak awal pelatihan.**

Racuni Dataset

```
import random

# Salin data latih
X_poisoned = X_train.copy()
y_poisoned = y_train.copy()

# Ubah label secara acak
for i in random.sample(range(len(y_poisoned)), 10):
    y_poisoned[i] = (y_poisoned[i] + 1) % 3
```

Train Model dengan Data Rusak

```
poisoned_model = LogisticRegression(max_iter=200)
poisoned_model.fit(X_poisoned, y_poisoned)

y_pred_poisoned = poisoned_model.predict(X_test)
print("Accuracy poisoned model:", accuracy_score(y_test,
y_pred_poisoned))
```

Hasil:

Akurasi turun, tetapi **AI tidak tahu bahwa ia sedang salah.**

Bagian E — Analisis & Refleksi

Apa yang Dipelajari Mahasiswa?

- **AI tidak memahami makna**, hanya pola
- Perubahan kecil → dampak besar
- Kualitas data lebih penting dari algoritma
- Serangan AI **murah dan mudah dilakukan**

Output Praktikum (Wajib Dikumpulkan)

Mahasiswa menyerahkan:

1. **Notebook Jupyter**
 - Training normal
 - Adversarial example
 - Data poisoning
2. **Analisis singkat (1–2 halaman):**
 - Bagian mana yang paling berbahaya?
 - Mana yang lebih sulit dideteksi?
 - Implikasi untuk AI di pemerintahan / industri?

Insight Penutup (Inspiratif)

AI bukan masalahnya.

Cara kita mempercayainya tanpa verifikasi — itulah masalahnya.

Modul ini membekali mahasiswa **mindset security-first**:

- Jangan percaya output AI
- Selalu audit data
- Selalu uji asumsi

BAB 10: AI untuk Keamanan Informasi

Fokus: AI sebagai Alat Pertahanan

Jika pada bab-bab sebelumnya kita membahas **AI sebagai target serangan**, maka pada bab ini sudut pandang dibalik sepenuhnya. **AI bukan lagi korban, tetapi penjaga.**

AI digunakan untuk **mendeteksi, menganalisis, dan merespons ancaman keamanan informasi** yang terlalu kompleks, terlalu cepat, dan terlalu masif jika ditangani secara manual.

Di dunia nyata, **serangan tidak selalu eksplisit**. Banyak serangan bersifat *low and slow*, menyamar sebagai aktivitas normal. Di sinilah **AI menjadi krusial**: bukan untuk menggantikan manusia, tetapi **memperluas kemampuan analisis keamanan**.

Mengapa Keamanan Modern Butuh AI?

Lingkungan IT modern menghasilkan **data keamanan dalam jumlah sangat besar**:

- Log jaringan
- Log server
- Aktivitas user
- Trafik aplikasi
- Event cloud & container

Masalahnya bukan kekurangan data, tapi **kelebihan data**.

Tanpa AI:

- Analisis hanya melihat **alert statis**
- Banyak *false positive*
- Serangan canggih lolos tanpa terdeteksi

Dengan AI:

- Sistem belajar **pola normal**
- Fokus pada **penyimpangan**
- Deteksi lebih cepat, bahkan sebelum serangan benar-benar terjadi

👉 **AI membantu kita menjawab satu pertanyaan penting:**

“Apakah perilaku ini normal, atau mencurigakan?”

Teori 1 — Machine Learning untuk Anomaly Detection

Apa itu Anomaly Detection?

Anomaly detection adalah teknik untuk **mendeteksi perilaku yang menyimpang dari kondisi normal**.

Bukan mencari tanda serangan tertentu, tapi mencari **sesuatu yang terasa “tidak biasa”**.

Contoh sederhana:

- User biasanya login pukul 08.00–17.00
- Tiba-tiba login pukul 03.12 dari lokasi berbeda
→ **Ini anomali**

Yang penting:

- **Anomali ≠ serangan**
- Tapi **serangan hampir selalu diawali anomali**

Pendekatan Rule-Based vs ML-Based

Rule-based (tradisional):

- Jika port 22 diakses → alert
- Jika login gagal > 5 kali → alert

Kelemahan:

- Mudah ditebak attacker
- Tidak adaptif
- Banyak *false positive*

ML-based:

- Sistem belajar pola trafik normal
- Tidak butuh aturan eksplisit
- Adaptif terhadap perubahan lingkungan

Inilah kekuatan AI:

👉 *belajar dari data, bukan hanya aturan.*

Jenis Pendekatan ML untuk Anomaly Detection

Secara sederhana, ada tiga pendekatan populer:

1. **Statistical-based**
 - Berdasarkan rata-rata, deviasi
 - Cocok untuk data sederhana
2. **Unsupervised Learning**
 - Tidak butuh label serangan
 - Contoh: *Isolation Forest*, *Autoencoder*
3. **Semi-supervised**
 - Dilatih hanya dengan data “normal”
 - Anomali = yang tidak dikenali

Untuk keamanan informasi, **unsupervised learning paling realistis**, karena:

- Data serangan jarang
- Label sering tidak akurat
- Lingkungan selalu berubah

Teori 2 — Security Analytics

Jika *anomaly detection* fokus pada “**apa yang aneh**”, maka *security analytics* fokus pada “**apa artinya**”.

Apa itu Security Analytics?

Security analytics adalah proses:

- Mengumpulkan data keamanan
- Mengkorelasi berbagai sumber data
- Mengambil insight berbasis analitik & AI

Bukan sekadar *alert*, tetapi **pemahaman konteks**.

Contoh:

- Login gagal + akses file sensitif + upload data
➡ **indikasi insider threat**

Perbedaan Log Monitoring vs Security Analytics

Log monitoring biasa:

- Event berdiri sendiri
- Alert reaktif
- Banyak noise

Security analytics:

- Event dikorelasi
- Analisis berbasis konteks
- Fokus pada *risk*

👉 Tujuannya bukan banyak alert, tapi alert yang tepat.

Peran AI dalam Security Analytics

AI digunakan untuk:

- Mengelompokkan perilaku user (*behavior profiling*)
- Mendeteksi pola serangan kompleks
- Menemukan hubungan tersembunyi antar event

Dengan AI:

- Analisis tidak tenggelam di log
- Fokus pada **investigasi bernilai tinggi**
- Respon insiden lebih cepat dan akurat

Mindset Penting: AI adalah Copilot, Bukan Pengganti

Hal yang sangat penting dipahami:

- **AI tidak menggantikan analisis keamanan**
- AI membantu mengurangi beban kognitif
- Keputusan akhir tetap di manusia

AI bisa:

- Salah interpretasi
- Overfitting
- Bias terhadap data tertentu

Karena itu:

AI + manusia = sistem pertahanan terbaik

Kenapa Bab Ini Penting untuk Mahasiswa & Praktisi?

Bab ini membuka perspektif bahwa:

- Keamanan bukan hanya firewall & antivirus
- Data keamanan bisa “dibaca” oleh mesin
- AI bukan hanya untuk chatbot atau image generation

Mahasiswa yang memahami ini:

- Siap masuk dunia *cybersecurity modern*
- Tidak terjebak pendekatan statis
- Mampu berpikir *data-driven security*

Ringkasan Kunci (Takeaways)

- **AI mengubah keamanan dari reaktif menjadi proaktif**
- *Anomaly detection* fokus pada penyimpangan perilaku
- *Security analytics* fokus pada konteks & makna
- **Unsupervised learning sangat relevan untuk keamanan**
- AI adalah **alat bantu cerdas**, bukan solusi ajaib

PRAKTEK 10: AI untuk Deteksi Anomali

Fokus sesi ini: kamu bikin “AI security” sederhana yang bisa **belajar pola normal** dari log, lalu menandai yang **aneh** (*anomaly*). Ini bukan “AI yang tahu segalanya”, tapi **alat bantu triage** biar analis tidak tenggelam dalam jutaan baris log.

Tujuan

Mahasiswa mampu:

- membangun pipeline deteksi anomali dari log nyata (Linux / web / auth),
- melatih model *unsupervised* (tanpa label),
- menghasilkan **daftar event mencurigakan** + alasan/fitur ringkas,
- menyimpan model dan menjalankan deteksi ulang secara berkala.

Output akhir yang ditargetkan:

- **Model tersimpan** (`.joblib`)
- **Laporan evaluasi sederhana** (rasio anomali, contoh top N anomali)
- **File hasil deteksi** (CSV/JSON)
- (Opsional) hasil/model dienkripsi dengan **GnuPG**

Konsep Inti

Deteksi anomali = mencari data yang “jarang”, “jauh dari pola normal”, atau “punya kombinasi fitur yang aneh”.

Kamu akan pakai dua pendekatan:

1. **Isolation Forest** (*tree-based*): bagus untuk anomaly detection umum, sering jadi baseline kuat.
2. **KMeans + jarak ke centroid**: sederhana, cepat, mudah dijelaskan (jarak besar = makin aneh).

Catatan penting: Model *unsupervised* akan menandai “aneh”, bukan otomatis “jahat”. **Anomali ≠ serangan**, tapi **anomali yang harus kamu cek dulu**.

Tools (Open Source)

- OS: **Ubuntu 24.04**
- Python 3 + venv
- Library: `pandas`, `numpy`, `scikit-learn`, `joblib`
- (Opsional) `matplotlib` untuk grafik ringan
- (Opsional) **GnuPG** untuk enkripsi file output/model

Skenario Data Log yang Real (Pilih salah satu atau gabungan)

Kamu bisa latihan pakai:

1. **Linux auth log:** `/var/log/auth.log`
Cocok untuk mendeteksi percobaan login gagal masif, lonjakan aktivitas sudo, jam akses aneh.
2. **Nginx access log** (lab): misalnya file `access.log` dari web server
Cocok untuk mendeteksi lonjakan request, path aneh, *user-agent* janggal, pola scanning.
3. **Suricata eve.json** (kalau sudah main IDS): event security lebih kaya.

Di modul ini kita buat pipeline yang **paling mudah jalan di semua laptop/server**: mulai dari **auth.log** + opsi format log sederhana.

Tahap Praktikum (Step-by-step)

0. Setup Environment di Ubuntu 24.04

Jalankan:

```
sudo apt update
sudo apt install -y python3-venv python3-pip gnupg
mkdir -p ~/modul10-ai-anomali/{data,models,output,scripts}
cd ~/modul10-ai-anomali
python3 -m venv .venv
source .venv/bin/activate
pip install -U pip
pip install pandas numpy scikit-learn joblib
```

Checklist: pastikan `python --version` mengarah ke `venv` dan `pip show scikit-learn` ada.

1. Ambil Dataset Log

Opsi A — pakai log asli mesin (paling real)

Copy auth log:

```
sudo cp /var/log/auth.log ~/modul10-ai-anomali/data/auth.log
sudo chown $USER:$USER ~/modul10-ai-anomali/data/auth.log
```

Opsi B — bikin dataset latihan (biar kontrol)

Kita akan generate log sintetik “mirip event” (normal + aneh) dari Python (nanti ada script).

2. Prinsip Feature Engineering (Supaya “AI” mengerti)

Log itu teks; model butuh angka. Maka kita ubah event jadi fitur numerik, contoh:

Untuk auth event:

- **hour** (jam kejadian)
- **fail_count_5m** (jumlah gagal login dalam 5 menit per IP/user)
- **distinct_users_10m**
- **is_sudo** (0/1)
- **is_failed_password** (0/1)
- **src_ip_hash** (hash → angka stabil; bukan identitas asli)
- **msg_len** (panjang pesan)

Yang penting: fitur harus menggambarkan **perilaku** (burst, jam tidak wajar, variasi user, dsb), bukan sekadar teks mentah.

Implementasi: Pipeline Lengkap (Python)

Di bawah ini 3 file utama:

1. parser & feature builder
2. training model
3. deteksi + export hasil

A. `scripts/parse_authlog.py` — parse `auth.log` → dataset fitur

Buat file:

```
nano ~/modul10-ai-anomali/scripts/parse_authlog.py
```

Isi:

```
#!/usr/bin/env python3
import re
import json
from datetime import datetime, timedelta
from collections import deque, defaultdict

# Auth log default Ubuntu: "Jan 20 13:01:02 hostname sshd[123]:
Failed password for ..."
# Catatan: tahun tidak ada, kita isi dengan tahun sekarang
(cukup untuk lab).
AUTH_RE = re.compile(

r'^(?P<mon>\w{3})\s+(?P<day>\d{1,2})\s+(?P<time>\d{2}:\d{2}:\d{2}
)\s+'

r'(?P<host>\S+)\s+(?P<proc>[\w\-\./]+)(?:\s+)?\s+(?P<msg>.*)
$'
)

IP_RE = re.compile(r'(\d{1,3}\.){3}\d{1,3}')
USER_RE = re.compile(r'for (invalid user
)?(?P<user>[a-zA-Z0-9_\-\.]+)')

MONTHS = {
    "Jan": 1, "Feb": 2, "Mar": 3, "Apr": 4, "May": 5, "Jun": 6,
    "Jul": 7, "Aug": 8, "Sep": 9, "Oct": 10, "Nov": 11, "Dec":
12
}

def stable_hash_to_int(s: str, mod: int = 1000003) -> int:
    # hash stabil sederhana (bukan cryptographic, cukup untuk
fitur)
    h = 2166136261
    for ch in s.encode("utf-8", errors="ignore"):
        h ^= ch
        h = (h * 16777619) & 0xFFFFFFFF
    return int(h % mod)

def parse_ts(line: str, year: int) -> datetime | None:
    m = AUTH_RE.match(line)
    if not m:
        return None
    mon = MONTHS.get(m.group("mon"))
    day = int(m.group("day"))
```

```

        t = m.group("time")
        hh, mm, ss = map(int, t.split(":"))
        return datetime(year, mon, day, hh, mm, ss)

def extract_ip(msg: str) -> str:
    m = IP_RE.search(msg)
    return m.group(0) if m else "0.0.0.0"

def extract_user(msg: str) -> str:
    m = USER_RE.search(msg)
    return m.group("user") if m else "unknown"

def build_features(lines: list[str], year: int):
    """
    Membuat event-level features + rolling window counts (burst
    behavior)
    """
    events = []
    # rolling windows untuk hitung burst
    window_5m_by_ip = defaultdict(deque)      # ip -> timestamps
    window_10m_by_user = defaultdict(deque)   # user ->
    timestamps

    for line in lines:
        m = AUTH_RE.match(line)
        if not m:
            continue

        ts = parse_ts(line, year)
        if not ts:
            continue

        msg = m.group("msg")
        proc = m.group("proc")
        ip = extract_ip(msg)
        user = extract_user(msg)

        is_failed = 1 if "Failed password" in msg else 0
        is_invalid_user = 1 if "invalid user" in msg else 0
        is_accepted = 1 if "Accepted password" in msg or
        "Accepted publickey" in msg else 0
        is_sudo = 1 if proc.startswith("sudo") or "sudo:" in msg
        else 0
        msg_len = len(msg)

        # update rolling window 5m per IP (untuk failed count)
        dq = window_5m_by_ip[ip]
        dq.append(ts)
        while dq and (ts - dq[0]) > timedelta(minutes=5):
            dq.popleft()
        count_5m_ip = len(dq)

        # update rolling window 10m per user

```

```

        du = window_10m_by_user[user]
        du.append(ts)
        while du and (ts - du[0]) > timedelta(minutes=10):
            du.popleft()
        count_10m_user = len(du)

        event = {
            "ts": ts.isoformat(),
            "hour": ts.hour,
            "minute": ts.minute,
            "proc": proc,
            "ip": ip,
            "user": user,
            "ip_hash": stable_hash_to_int(ip),
            "user_hash": stable_hash_to_int(user),
            "is_failed": is_failed,
            "is_invalid_user": is_invalid_user,
            "is_accepted": is_accepted,
            "is_sudo": is_sudo,
            "msg_len": msg_len,
            "count_5m_ip": count_5m_ip,
            "count_10m_user": count_10m_user,
        }
        events.append(event)

    return events

def main():
    import argparse
    parser = argparse.ArgumentParser()
    parser.add_argument("--infile", required=True, help="path ke
auth.log")
    parser.add_argument("--outfile", required=True, help="output
JSONL features")
    args = parser.parse_args()

    year = datetime.now().year

    with open(args.infile, "r", encoding="utf-8",
errors="ignore") as f:
        lines = f.readlines()

    events = build_features(lines, year)

    with open(args.outfile, "w", encoding="utf-8") as out:
        for ev in events:
            out.write(json.dumps(ev) + "\n")

    print(f"[OK] Parsed {len(events)} events -> {args.outfile}")

if __name__ == "__main__":
    main()

```

Jalankan:

```
chmod +x scripts/parse_authlog.py
./scripts/parse_authlog.py --infile data/auth.log --outfile
data/auth_features.jsonl
head -n 3 data/auth_features.jsonl
```

B. `scripts/train_models.py` — latih Isolation Forest + KMeans

Buat file:

```
nano ~/modul10-ai-anomali/scripts/train_models.py
```

Isi:

```
#!/usr/bin/env python3
import json
import joblib
import numpy as np
from pathlib import Path
from sklearn.ensemble import IsolationForest
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

FEATURE_COLS = [
    "hour", "minute",
    "ip_hash", "user_hash",
    "is_failed", "is_invalid_user", "is_accepted", "is_sudo",
    "msg_len",
    "count_5m_ip", "count_10m_user"
]

def load_jsonl(path: str):
    rows = []
    with open(path, "r", encoding="utf-8") as f:
        for line in f:
            rows.append(json.loads(line))
    return rows

def to_matrix(rows):
    X = []
    for r in rows:
        X.append([float(r.get(c, 0.0)) for c in FEATURE_COLS])
    return np.array(X, dtype=float)

def main():
    import argparse
    p = argparse.ArgumentParser()
    p.add_argument("--infile", required=True, help="JSONL
features")
    p.add_argument("--outdir", required=True, help="folder
simpan model")
```

```

        p.add_argument("--contamination", type=float, default=0.02,
                        help="perkiraan rasio anomali (mis.
0.01-0.05)")
        p.add_argument("--k", type=int, default=8, help="jumlah
cluster KMeans")
        args = p.parse_args()

        outdir = Path(args.outdir)
        outdir.mkdir(parents=True, exist_ok=True)

        rows = load_jsonl(args.infile)
        X = to_matrix(rows)

        scaler = StandardScaler()
        Xs = scaler.fit_transform(X)

        iso = IsolationForest(
            n_estimators=300,
            contamination=args.contamination,
            random_state=42,
            n_jobs=-1
        )
        iso.fit(Xs)

        km = KMeans(n_clusters=args.k, random_state=42,
n_init="auto")
        km.fit(Xs)

        bundle = {
            "feature_cols": FEATURE_COLS,
            "scaler": scaler,
            "isolation_forest": iso,
            "kmeans": km
        }
        model_path = outdir / "anomali_models.joblib"
        joblib.dump(bundle, model_path)

        # ringkasan cepat
        iso_pred = iso.predict(Xs) # -1 anomali, 1 normal
        anom_ratio = float(np.mean(iso_pred == -1))

        print("[OK] Model saved:", model_path)
        print(f"[INFO] Events: {len(rows)} | Estimated anomaly ratio
(IF): {anom_ratio:.4f}")

if __name__ == "__main__":
    main()

```

Jalankan:

```
chmod +x scripts/train_models.py
./scripts/train_models.py --infile data/auth_features.jsonl
--outdir models --contamination 0.02 --k 8
ls -lah models
```

Tips tuning seru: mainkan `--contamination` (mis. 0.01, 0.03, 0.05). Lihat bagaimana jumlah anomali berubah.

C. `scripts/detect_anomalies.py` — scoring, ranking, export

Buat file:

```
nano ~/modul10-ai-anomali/scripts/detect_anomalies.py
```

Isi:

```
#!/usr/bin/env python3
import json
import csv
import joblib
import numpy as np
from pathlib import Path

def load_jsonl(path: str):
    rows = []
    with open(path, "r", encoding="utf-8") as f:
        for line in f:
            rows.append(json.loads(line))
    return rows

def main():
    import argparse
    p = argparse.ArgumentParser()
    p.add_argument("--features", required=True, help="JSONL features")
    p.add_argument("--model", required=True, help="joblib model bundle")
    p.add_argument("--outcsv", required=True, help="output CSV anomali")
    p.add_argument("--top", type=int, default=50, help="ambil top N paling anomali")
    args = p.parse_args()

    bundle = joblib.load(args.model)
    cols = bundle["feature_cols"]
    scaler = bundle["scaler"]
    iso = bundle["isolation_forest"]
    km = bundle["kmeans"]

    rows = load_jsonl(args.features)
```

```

        X = np.array([[float(r.get(c, 0.0)) for c in cols] for r in
rows], dtype=float)
        Xs = scaler.transform(X)

        # IsolationForest: decision_function makin kecil -> makin
anomali
        iso_score = iso.decision_function(Xs) # higher = more
normal
        iso_label = iso.predict(Xs)          # -1 anomali

        # KMeans: jarak ke centroid terdekat (makin jauh -> makin
anomali)
        centers = km.cluster_centers_
        # hitung jarak L2 ke centroid terdekat
        dists = np.sqrt(((Xs[:, None, :] - centers[None, :, :]) **
2).sum(axis=2))
        km_dist = dists.min(axis=1)

        # gabung score sederhana: rank berdasarkan 2 sinyal
        # normalisasi kasar
        iso_norm = (iso_score - iso_score.min()) / (iso_score.max()
- iso_score.min() + 1e-9)
        km_norm = (km_dist - km_dist.min()) / (km_dist.max() -
km_dist.min() + 1e-9)

        # semakin kecil iso_norm = semakin anomali; semakin besar
km_norm = semakin anomali
        combined = (1.0 - iso_norm) * 0.6 + (km_norm * 0.4)

        # pilih top N
        idx = np.argsort(-combined)[:args.top]

        outpath = Path(args.outcsv)
        outpath.parent.mkdir(parents=True, exist_ok=True)

        fieldnames = [
            "rank", "combined_score",
            "iso_label", "iso_score",
            "km_dist",
            "ts", "hour", "minute", "proc", "user", "ip",
            "is_failed", "is_invalid_user", "is_accepted",
            "is_sudo",
            "count_5m_ip", "count_10m_user", "msg_len"
        ]

        with open(outpath, "w", newline="", encoding="utf-8") as f:
            w = csv.DictWriter(f, fieldnames=fieldnames)
            w.writeheader()
            for rnk, i in enumerate(idx, start=1):
                r = rows[int(i)]
                w.writerow({
                    "rank": rnk,
                    "combined_score": float(combined[i]),

```

```

        "iso_label": int(iso_label[i]),
        "iso_score": float(iso_score[i]),
        "km_dist": float(km_dist[i]),
        "ts": r.get("ts", ""),
        "hour": r.get("hour", 0),
        "minute": r.get("minute", 0),
        "proc": r.get("proc", ""),
        "user": r.get("user", ""),
        "ip": r.get("ip", ""),
        "is_failed": r.get("is_failed", 0),
        "is_invalid_user": r.get("is_invalid_user", 0),
        "is_accepted": r.get("is_accepted", 0),
        "is_sudo": r.get("is_sudo", 0),
        "count_5m_ip": r.get("count_5m_ip", 0),
        "count_10m_user": r.get("count_10m_user", 0),
        "msg_len": r.get("msg_len", 0),
    })

    anom_count = int(np.sum(iso_label == -1))
    print(f"[OK] Wrote top-{args.top} anomalies -> {outpath}")
    print(f"[INFO] Total events: {len(rows)} | IF anomalies
flagged: {anom_count}")

if __name__ == "__main__":
    main()

```

Jalankan:

```

chmod +x scripts/detect_anomalies.py
./scripts/detect_anomalies.py --features
data/auth_features.jsonl --model models/anomali_models.joblib
--outcsv output/anomali_top.csv --top 50
column -s, -t output/anomali_top.csv | head -n 20

```

Cara Membaca Hasil

Di [output/anomali_top.csv](#), fokus ke:

- **count_5m_ip tinggi + is_failed=1** → indikasi brute force
- **hour sangat dini** (mis. 02:00) + **is_sudo=1** → aktivitas admin jam aneh
- **user=unknown / invalid_user=1** berulang → scanning user
- **proc/sshd** dominan → serangan ke SSH (umum banget di server publik)

Tugas mini yang menantang:

- Ambil 10 anomali teratas, lalu tulis analisis 1–2 kalimat per event:
 - “Kenapa ini anomali?”
 - “Apa tindakan lanjut?” (block IP? cek user? cek sistem?)

Simulasi Serangan Ringan (Aman untuk Lab)

Kalau kamu punya VM/host lab sendiri, bisa memicu event gagal login (tanpa merusak):

- Coba login SSH dengan user salah beberapa kali dari client lab.
- Atau buat event sudo beberapa kali.

Penting: lakukan hanya di lingkungan yang kamu miliki/diizinkan.

(Opsional) Amankan Model & Output dengan GnuPG

Tujuannya: hasil deteksi bisa berisi data sensitif (user, IP, pola aktivitas). Minimal, kamu bisa enkripsi file output dan model sebelum dipindah/diupload.

1. Generate key (sekali saja)

```
gpg --full-generate-key
```

Cek key:

```
gpg --list-keys
```

2. Enkripsi output CSV

Misal email key kamu **you@example.com**:

```
gpg --output output/anomali_top.csv.gpg --encrypt --recipient  
you@example.com output/anomali_top.csv
```

Decrypt:

```
gpg --output output/anomali_top.csv --decrypt  
output/anomali_top.csv.gpg
```

3. Enkripsi model

```
gpg --output models/anomali_models.joblib.gpg --encrypt --recipient  
you@example.com models/anomali_models.joblib
```

Skill security yang dinilai: kamu tidak hanya bikin AI, tapi juga **mengelola artefak** (model/output) dengan aman.

Output yang Wajib Dikumpulkan

1. `data/auth_features.jsonl` (atau ringkasannya)
2. `models/anomali_models.joblib` (atau versi `.gpg`)
3. `output/anomali_top.csv` (atau versi `.gpg`)
4. ...
5. `Laporan.md` singkat berisi:
 - o deskripsi dataset (berapa event),
 - o parameter model (*contamination*, *k*),
 - o 10 anomali teratas + analisis,
 - o 3 rekomendasi aksi.

Template laporan cepat:

```
# Laporan Modul 10 – AI Deteksi Anomali

## Dataset
- Sumber: auth.log
- Jumlah event: ...
- Rentang waktu: ...

## Model
- IsolationForest contamination: ...
- KMeans k: ...
- Fitur: hour, minute, is_failed, count_5m_ip, ...

## Temuan Top 10
1) ...
  - alasan: ...
  - tindak lanjut: ...

## Rekomendasi
- ...
- ...
- ...
```

Challenge Upgrade (Kalau Mau Naik Level)

Kalau mahasiswa cepat selesai, kasih 1–2 tantangan ini:

- Tambahkan fitur: **hari (weekday)** dan deteksi “akses weekend”.
- Buat mode “stream”: baca log baru (tail) dan skor on-the-fly.
- Ganti `auth.log` ke **Nginx access log** dan buat fitur:
 - o request per IP per menit,
 - o status code 404/500 burst,
 - o path yang jarang muncul.

BAB 11: AI untuk Audit Data & Privasi

Pada tahap ini, kita masuk ke dunia **audit data modern**—bukan audit manual yang mengandalkan spreadsheet dan insting, tetapi audit yang **dibantu AI**, berjalan **otomatis, konsisten, dan bisa diskalakan**. Tujuan utama sesi ini adalah mengubah cara berpikir mahasiswa: **audit bukan aktivitas reaktif**, melainkan **proses berkelanjutan (*continuous audit*)** yang berjalan seiring sistem.

Dalam praktik nyata, organisasi **tidak lagi tahu persis dimana data sensitif mereka berada**. File log, database dump, backup lama, dokumen PDF, CSV, bahkan chat internal—semuanya bisa mengandung data pribadi. Di sinilah AI berperan sebagai “**mata tambahan**” yang tidak lelah membaca ribuan file.

Tujuan

Setelah sesi ini, mahasiswa diharapkan:

- Memahami **apa itu data sensitif dan mengapa harus diaudit**
- Mampu menjelaskan konsep **data sensitivity detection** dan **PII detection**
- Memiliki *mindset* bahwa **privasi harus diaudit, bukan diasumsikan aman**
- Siap mengimplementasikan audit otomatis pada sistem nyata

Gambaran Besar: Mengapa Audit Data Butuh AI?

Audit manual biasanya:

- Lambat
- Tidak konsisten
- Bergantung pada sampel (bukan keseluruhan data)

Sedangkan audit berbasis AI:

- Bisa memindai **seluruh data**
- Konsisten mengikuti aturan
- Mendeteksi pola yang **tidak terlihat oleh manusia**

Prinsip penting:

Kalau data tidak pernah diaudit, maka kebocoran hanya soal waktu.

Data Sensitivity Detection

(Mendeteksi Tingkat Sensitivitas Data)

Data sensitivity detection adalah proses untuk **mengklasifikasikan data berdasarkan tingkat risikonya** jika bocor, disalahgunakan, atau diakses tanpa izin.

Bayangkan sebuah folder server berisi:

- `mahasiswa.csv`
- `backup_2022.sql`
- `log_aplikasi.txt`
- `email_list.xlsx`

Sekilas terlihat biasa. Namun AI bisa menemukan bahwa:

- `mahasiswa.csv` berisi **NIK dan tanggal lahir**
- `backup_2022.sql` menyimpan **password hash**
- `log_aplikasi.txt` menyimpan **token API**
- `email_list.xlsx` berisi **email pribadi**

Semua ini **tidak aman jika dibiarkan tanpa audit**.

Tingkat Sensitivitas (Contoh Praktis)

- **Rendah:** Data publik, dokumentasi umum
- **Sedang:** Email, username, metadata
- **Tinggi:** NIK, nomor rekening, biometrik, token autentikasi

Catatan penting:

Sensitivitas data tidak ditentukan oleh format file, tetapi oleh isinya.

PII Detection (Personally Identifiable Information)

PII detection adalah proses otomatis untuk **menemukan data yang dapat mengidentifikasi individu secara langsung atau tidak langsung**.

Contoh *PII* yang sering ditemukan tanpa sadar:

- Nama lengkap + email
- NIK / No. KTP
- Nomor HP
- Alamat rumah
- Kombinasi data kecil yang *jika digabung* bisa mengidentifikasi seseorang

AI sangat efektif di sini karena mampu:

- Mengenali **pola** (regex, struktur data)
- Memahami **konteks teks**
- Mendeteksi *PII tersembunyi* di log atau catatan lama

Contoh Nyata

Sebuah file log aplikasi:

```
User login success: nama=Andi, nik=3174xxxx, ip=103.xxx.xxx
```

Bagi manusia, ini hanya log biasa.

Bagi AI, ini adalah **data pribadi sensitif tingkat tinggi**.

Pendekatan AI dalam Audit Privasi

AI tidak “menebak”, tetapi bekerja berdasarkan:

- **Pola (*pattern recognition*)**
- **Konteks bahasa**
- **Aturan yang bisa dikombinasikan dengan pembelajaran mesin**

Pendekatan umum:

- Rule-based (regex, keyword)
- AI/NLP-based (memahami makna teks)
- Hybrid (aturan + AI)

Insight penting:

Audit terbaik bukan yang paling canggih, tapi yang **konsisten dijalankan**.

Studi Kasus Mini (Untuk Dipikirkan)

Bayangkan kamu adalah auditor data di kampus:

- Ada folder **/backup/**
- Ada folder **/log/**
- Ada folder **/dokumen/**

Pertanyaan reflektif:

- Data apa yang **paling berbahaya jika bocor**?
- Apakah data lama **lebih aman dari data baru**?
- Apakah log seharusnya menyimpan data pribadi?

Diskusi ini penting karena **AI hanya alat**—keputusan tetap di manusia.

Kesalahan Umum dalam Audit Data

Beberapa kesalahan fatal yang sering terjadi:

- Menganggap **backup aman karena jarang diakses**
- Tidak pernah mengaudit **log aplikasi**
- Fokus pada database, lupa file teks & dokumen
- Mengira enkripsi = tidak perlu audit

Bold reminder:

Enkripsi melindungi data, audit melindungi organisasi.

Penutup: Dari Audit ke Budaya Privasi

Tujuan akhir dari AI audit bukan sekadar laporan, tetapi:

- Membangun **kesadaran privasi**
- Mengurangi risiko kebocoran
- Mendukung kepatuhan regulasi (UU PDP, GDPR, dsb.)
- Membuat sistem **lebih bertanggung jawab secara etis**

Audit otomatis dengan AI adalah langkah awal menuju:

Privasi-by-design, bukan *privacy-by-accident*.

PRAKTEK 11: AI untuk Audit Data Pribadi

Di dunia nyata, kebocoran data sering bukan karena hacker super canggih—tapi karena **data pribadi “nyangkut” di tempat yang tidak seharusnya**: file CSV di folder proyek, log aplikasi, backup SQL lama, export Excel, atau chat yang dipaste ke dokumen.

Di sesi ini, mahasiswa akan bikin *tool* audit yang bertindak seperti “**robot auditor**”: ia masuk ke folder data, membaca file satu per satu, lalu **mendeteksi PII** (*Personally Identifiable Information*), memberi **skor risiko**, dan menghasilkan laporan.

Mindset penting:

- **Audit itu kebiasaan (*continuous*)**, bukan tugas akhir semester.
- Target kita bukan “sempurna”, tapi **lebih aman tiap iterasi**.

Tujuan

Mahasiswa mampu:

- Mendesain pipeline audit data: **scan** → **detect** → **classify** → **report** → **protect**
- Mendeteksi PII dengan pendekatan **Regex + NLP/ML ringan**
- Memberi **risk score** untuk membantu prioritas perbaikan
- Menghasilkan **AI Data Scanner** (CLI tool) + laporan JSON/CSV
- Melakukan langkah “tindak lanjut” menggunakan *GnuPG*: **enkripsi / redaksi**

Konsep: PII Detection (Realistic)

PII adalah data yang bisa mengidentifikasi seseorang, baik langsung maupun kombinasi.

Contoh PII yang sering ditemukan di data kampus/perusahaan:

- **Email** (*nama@domain*)
- **Nomor HP** (*08xx...* / *+62...*)
- **NIK** (16 digit, kadang dipisah spasi/strip)
- **No. Rekening** (variasi panjang)
- **Alamat** (lebih sulit—banyak variasi)
- **IP address** (bukan selalu PII, tapi bisa sensitif)
- **API token / session token** (bukan PII, tapi sensitif)

Trik audit yang benar:

Deteksi bukan hanya “ada email”, tapi juga **konteks**:

- Email muncul di data “publik” → risiko sedang
- Email + NIK + tanggal lahir → risiko tinggi
- Token API di log → risiko kritikal

Tools (Open Source)

1. Python (Utama)

- `re` untuk *regex*
- `json`, `csv`, `pathlib`
- (opsional) `scikit-learn` untuk ML ringan
- (opsional) `spacy` untuk NLP (bisa berat, tapi open-source)

2. GnuPG (GNU Privacy Guard)

Dipakai untuk “aksi setelah audit”:

- **enkripsi** file yang mengandung PII
- **sign** laporan audit (integritas)
- **dekripsi** saat perlu akses aman

Tahap Praktikum (Step-by-step)

Tahap 0 — Setup Environment (Ubuntu 24.04)

Jalankan:

```
sudo apt update
sudo apt install -y python3 python3-venv python3-pip gnupg jq
python3 -m venv venv
source venv/bin/activate
pip install --upgrade pip
```

Buat folder proyek:

```
mkdir -p ai_auditor/{data,output}
```

Tahap 1 — Siapkan Dataset Contoh (Realistic)

Buat file `data/sample.txt`:

```
cat > data/sample.txt << 'EOF'
Nama: Budi Santoso
Email: budi.santoso@gmail.com
NIK: 3174012301990002
HP: +62 812-3456-7890
Alamat: Jl. Melati No. 10, Tangerang Selatan
Catatan: reset token = sk_live_51Hxxxxxxx
IP Login: 103.10.20.30
EOF
```

Buat juga contoh CSV `data/mahasiswa.csv`:

```
cat > data/mahasiswa.csv << 'EOF'
nama,email,nik,no_hp
Ayu,ayu@kampus.ac.id,3174020101010003,081234567890
Doni,doni@gmail.com,3174030202020004,628123111222
EOF
```

Tahap 2 — Bangun “AI Data Scanner” Versi 1 (Regex Engine)

Buat file `ai_auditor.py`:

```
#!/usr/bin/env python3
import re
import json
from pathlib import Path
from datetime import datetime

# --- Pola PII (sengaja sederhana dulu, nanti bisa ditingkatkan)
---
PATTERNS = {
    "EMAIL":
re.compile(r"\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b"
),
    "PHONE_ID":
re.compile(r"\b(\+62\s?8\d{2}[-\s]?d{3,4}[-\s]?d{3,4}|\b08\d{8
,12}\b|\b62\d{9,13}\b)"),
    "NIK_ID": re.compile(r"\b\d{16}\b"),
    "IP_ADDR":
re.compile(r"\b(?:?:25[0-5]|2[0-4]\d|[01]?\d?\d)\.){3}(?:25[0-5
]|2[0-4]\d|[01]?\d?\d)\b"),
    # Token contoh (jangan terlalu agresif biar tidak banyak
false positive)
    "API_TOKEN":
re.compile(r"\b(sk_live|sk_test|token|apikey|api_key|bearer)\S{6
,}\b", re.IGNORECASE),
}

# Bobot risiko (bisa disesuaikan)
RISK_WEIGHT = {
    "EMAIL": 2,
    "PHONE_ID": 3,
    "NIK_ID": 5,
    "IP_ADDR": 1,
    "API_TOKEN": 6,
}

def classify_risk(found_types):
    score = sum(RISK_WEIGHT.get(t, 0) for t in found_types)
    if score >= 10:
        return score, "HIGH"
    if score >= 5:
```

```

        return score, "MEDIUM"
    return score, "LOW"

def scan_text(text):
    findings = []
    found_types = set()
    for t, rx in PATTERNS.items():
        matches = rx.findall(text)
        if matches:
            found_types.add(t)
            # Normalisasi hasil findall (kadang tuple)
            norm = []
            for m in matches:
                norm.append(m[0] if isinstance(m, tuple) else m)
            findings.append({
                "type": t,
                "count": len(norm),
                "samples": list(dict.fromkeys(norm)[:3], # max
3 sample unik
            })
    score, level = classify_risk(found_types)
    return findings, score, level

def read_file(path: Path):
    # Baca sebagai text "best effort"
    try:
        return path.read_text(encoding="utf-8",
errors="replace")
    except Exception:
        return ""

def main(input_dir="data", output_dir="output"):
    input_path = Path(input_dir)
    out_path = Path(output_dir)
    out_path.mkdir(parents=True, exist_ok=True)

    report = {
        "generated_at": datetime.utcnow().isoformat() + "Z",
        "scanned_dir": str(input_path.resolve()),
        "files": [],
        "summary": {"total_files": 0, "flagged_files": 0,
"by_level": {"LOW": 0, "MEDIUM": 0, "HIGH": 0}},
    }

    for p in input_path.rglob("*"):
        if not p.is_file():
            continue

        # Filter file sederhana (bisa diperluas)
        if p.suffix.lower() not in [".txt", ".csv", ".log",
".json"]:
            continue

```

```

text = read_file(p)
findings, score, level = scan_text(text)

report["summary"]["total_files"] += 1

file_entry = {
    "path": str(p),
    "size_bytes": p.stat().st_size,
    "risk_score": score,
    "risk_level": level,
    "findings": findings,
}

if findings:
    report["summary"]["flagged_files"] += 1
    report["summary"]["by_level"][level] += 1

report["files"].append(file_entry)

# Simpan laporan
out_file = out_path / "audit_report.json"
out_file.write_text(json.dumps(report, indent=2,
ensure_ascii=False), encoding="utf-8")

print(f"[OK] Report saved: {out_file}")
print(json.dumps(report["summary"], indent=2))

if __name__ == "__main__":
    main()

```

Jalankan:

```

chmod +x ai_auditor.py
./ai_auditor.py

```

Cek laporan:

```

cat output/audit_report.json | jq .

```

Tahap 3 — “Tandai Risiko” + Output yang Bisa Dipakai (CSV Ringkas)

Tambahkan output ringkas `output/audit_summary.csv` (ini real dipakai tim audit).

Buat file `export_summary.py`:

```
#!/usr/bin/env python3
import json
import csv
from pathlib import Path

def main(report_path="output/audit_report.json",
out_csv="output/audit_summary.csv"):
    report =
    json.loads(Path(report_path).read_text(encoding="utf-8"))
    rows = []
    for f in report["files"]:
        if not f["findings"]:
            continue
        types = ",".join(x["type"] for x in f["findings"])
        rows.append([f["path"], f["risk_level"],
f["risk_score"], types, f["size_bytes"]])

    with open(out_csv, "w", newline="", encoding="utf-8") as fp:
        w = csv.writer(fp)
        w.writerow(["path", "risk_level", "risk_score",
"finding_types", "size_bytes"])
        w.writerows(rows)

    print(f"[OK] CSV saved: {out_csv}")

if __name__ == "__main__":
    main()
```

Run:

```
chmod +x export_summary.py
./export_summary.py
column -t -s, output/audit_summary.csv
```

Tahap 4 — Upgrade ke “Regex + ML ringan” (Opsional tapi Keren)

Regex bagus untuk pola tetap (email, NIK). Tapi untuk **alamat / nama / kalimat sensitif**, kita butuh *ML* sederhana.

Konsep ML ringan yang realistis untuk kelas:

- Kita bikin **text classifier** yang menentukan kalimat termasuk “sensitif” atau “tidak”
- Model: *Logistic Regression / Linear SVM*
- Fitur: *TF-IDF*

Install:

```
pip install scikit-learn
```

Buat file `ml_sensitive_classifier.py`:

```
#!/usr/bin/env python3
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression

# Dataset mini (contoh). Di kelas, mahasiswa harus menambah data
sendiri.
TRAIN_TEXT = [
    "NIK saya 31740123019900002",
    "Nomor HP saya 081234567890",
    "Email saya budi@gmail.com",
    "Token reset: sk_live_123xxx",
    "Alamat saya Jl. Melati No. 10",
    "Hari ini cuaca cerah",
    "Saya suka belajar keamanan informasi",
    "Meeting jam 2 siang",
]
TRAIN_LABEL = [1,1,1,1,1,0,0,0] # 1 = sensitif, 0 =
non-sensitif

def build_model():
    vec = TfidfVectorizer(ngram_range=(1,2))
    X = vec.fit_transform(TRAIN_TEXT)
    clf = LogisticRegression(max_iter=200)
    clf.fit(X, TRAIN_LABEL)
    return vec, clf

def predict_lines(lines, threshold=0.6):
    vec, clf = build_model()
    X = vec.transform(lines)
    proba = clf.predict_proba(X)[:,1]
    results = []
    for line, p in zip(lines, proba):
        results.append((line, float(p), "SENSITIVE" if p >=
threshold else "OK"))
```

```

return results

if __name__ == "__main__":
    test = [
        "Silakan kirim NIK dan nomor HP untuk verifikasi",
        "Jangan lupa rapat jam 3",
        "Password admin adalah admin123",
        "Email saya ayu@kampus.ac.id",
    ]
    for line, p, label in predict_lines(test):
        print(f"{label}\t{p:.2f}\t{line}")

```

Ini bukan “AI super”, tapi **realistic ML pipeline** yang bisa dibesarkan dataset-nya.

Tahap 5 — Aksi Setelah Audit dengan *GnuPG*

Audit tanpa aksi = cuma “tahu masalah”. Sekarang kita lakukan **aksi aman**.

1. Generate Key (untuk kelas, pakai mode cepat)

```
gpg --full-generate-key
```

Cek key:

```
gpg --list-keys
```

2. Enkripsi Laporan Audit

Enkripsi file laporan supaya aman saat dikirim ke dosen/tim:

```
gpg -c output/audit_report.json
```

Hasilnya: **output/audit_report.json.gpg**

Dekripsi:

```
gpg -d output/audit_report.json.gpg >
output/audit_report.decrypted.json
```

3. Sign Laporan (Integritas)

Ini penting untuk audit: **laporan harus bisa dibuktikan tidak diubah**.

```
gpg --clearsign output/audit_summary.csv
```

Hasil: **output/audit_summary.csv.asc**

Verifikasi:

```
gpg --verify output/audit_summary.csv.asc
```

Output yang Wajib Dikumpulkan (Sesuai Modul)

Mahasiswa menyerahkan:

- `ai_auditor.py` (AI data scanner)
- `output/audit_report.json` (detail)
- `output/audit_summary.csv` (ringkas)
- `output/audit_report.json.gpg` (enkripsi)
- `output/audit_summary.csv.asc` (signature)
- Laporan singkat (1–2 halaman) berisi:
 - Temuan PII paling sering
 - File paling berisiko
 - Rekomendasi perbaikan

Challenge (Fun & Menantang)

Pilih minimal 2:

1. Tambah deteksi *PII* lain: **NPWP / Passport / KK** (buat regex-nya)
2. Buat fitur **redaction**: output file versi “tersensor” (contoh: email jadi `***@***`)
3. Buat rule “**HIGH jika NIK + email muncul di file yang sama**”
4. Buat *whitelist* domain email kampus (mis. `@kampus.ac.id`) dan beri skor lebih rendah
5. Tambah deteksi “**leak token**” dengan pola lebih baik

Bonus: Redaction (Contoh Kode yang Real Dipakai)

Buat `redact.py`:

```
#!/usr/bin/env python3
import re
from pathlib import Path

EMAIL =
re.compile(r"\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b")
NIK = re.compile(r"\b\d{16}\b")
PHONE =
re.compile(r"\b(\+62\s?8\d{2}[-\s]?d{3,4}[-\s]?d{3,4}|\b08\d{8},12\b|\b62\d{9,13}\b)")

def redact_text(t: str) -> str:
    t = EMAIL.sub("***@1", t)
    t = NIK.sub("*****", t)
```

```

        t = PHONE.sub("***REDACTED_PHONE***", t)
        return t

def main(infile="data/sample.txt",
outfile="output/sample.redacted.txt"):
    p = Path(infile)
    text = p.read_text(encoding="utf-8", errors="replace")
    out = redact_text(text)
    Path(outfile).write_text(out, encoding="utf-8")
    print(f"[OK] Redacted saved: {outfile}")

if __name__ == "__main__":
    main()

```

Run:

```

chmod +x redact.py
./redact.py
cat output/sample.redacted.txt

```

BAB 12 — Privacy-Preserving AI

Fokus: Aman tanpa bocor

Di era AI modern, **data adalah bahan bakar utama**. Namun, semakin canggih model yang kita bangun, semakin besar pula **risiko kebocoran data pribadi**. *Privacy-Preserving AI* hadir sebagai pendekatan yang **menjaga manfaat AI tanpa mengorbankan privasi manusia**. Pada pertemuan ini, kita tidak berbicara soal “menyembunyikan data”, melainkan **merancang sistem AI yang sejak awal memang tidak membutuhkan data mentah secara langsung**.

Materi ini dibuat **mudah dipahami oleh technical reader**, fun, dan tetap menantang. Anggap ini sebagai *mindset upgrade*: dari “*bagaimana bikin model pintar*” menjadi “*bagaimana bikin model pintar **dan** bertanggung jawab*”.

Gambaran Besar Privacy-Preserving AI

Secara sederhana, *Privacy-Preserving AI* adalah kumpulan teknik yang memungkinkan:

- **AI belajar dari data sensitif tanpa melihat data aslinya**
- **Risiko kebocoran data ditekan sejak tahap desain**
- **Kepatuhan terhadap regulasi privasi (UU PDP, GDPR, dsb.) menjadi realistis**

Pendekatan ini penting karena:

- **Data bocor ≠ hanya masalah teknis**, tapi juga hukum, reputasi, dan etika
- AI yang “terlalu tahu” sering kali **menyimpan jejak data personal**
- Keamanan bukan hanya firewall, tapi **cara model dilatih**

Differential Privacy (DP)

Privasi lewat ketidakpastian terkontrol

Differential Privacy adalah konsep yang terdengar rumit, tapi ide intinya sangat elegan:

👉 **Tambahkan *noise* (gangguan acak) yang terukur agar data individu tidak bisa dilacak, tetapi pola besar tetap terlihat.**

Bayangkan kita punya data nilai ujian mahasiswa. Kita ingin tahu **rata-rata nilai**, tapi **tidak ingin siapa pun bisa menebak nilai individu tertentu**. DP memastikan bahwa:

- Kehadiran atau ketiadaan **satu orang** dalam dataset
- **Tidak mengubah hasil secara signifikan**

Intuisi Sederhana

- Sistem dengan DP:
“Hasilnya hampir sama, **baik kamu ikut maupun tidak.**”
- Artinya: **privasi individu terlindungi secara matematis**

Konsep Kunci

- **Noise** → gangguan acak yang ditambahkan ke hasil
- **Epsilon (ϵ)** → parameter yang mengatur *trade-off*
 - ϵ kecil → **lebih privat**, tapi akurasi turun
 - ϵ besar → **lebih akurat**, tapi privasi berkurang
- **Query-based thinking** → DP fokus pada *hasil*, bukan data mentah

Kenapa Ini Penting?

- Cocok untuk **statistik publik**, *dashboard*, dan *analytics*
- Banyak dipakai oleh perusahaan besar untuk data sensitif
- **Privasi bukan asumsi**, tapi **dijamin secara matematis**

Insight penting:

Differential Privacy tidak menyembunyikan data, tapi membuat data individu menjadi “tidak penting” bagi penyerang.

Federated Learning (FL) — Konsep Dasar

Belajar bersama tanpa berbagi data

Jika Differential Privacy bermain di level *statistik*, **Federated Learning** bermain di level *arsitektur sistem*. Ide besarnya sangat powerful:

**Data tidak pernah dikirim ke server pusat.
Model-lah yang datang ke data.**

Cara Kerja (Konseptual)

1. **Model awal** dikirim ke banyak perangkat (*client*)
2. Setiap client:
 - Melatih model **di data lokal**
 - Data **tidak pernah keluar**
3. Client hanya mengirim:
 - **Update model** (misalnya *gradient*)
4. Server:
 - Menggabungkan (*aggregate*) update
 - Menghasilkan model global yang lebih pintar

Kenapa Ini Aman?

- **Tidak ada data mentah di server pusat**

- Kebocoran satu node $\neq$ kebocoran seluruh dataset
- Cocok untuk data:
 - Medis
 - Finansial
 - Perilaku pengguna

Federated Learning $\neq$ Tanpa Risiko

Perlu dipahami juga bahwa:

- Update model **masih bisa bocor informasi** jika tidak diamankan
- Bias data lokal bisa mempengaruhi model global
- Biasanya dikombinasikan dengan:
 - *Differential Privacy*
 - *Secure aggregation*

Use Case Nyata

- Keyboard prediksi kata
- Model rekomendasi personal
- Analitik terdistribusi di organisasi besar

Insight penting:

Federated Learning memindahkan pusat risiko dari “database besar” ke “koordinasi yang cerdas”.

Perbandingan Singkat

Aspek	Differential Privacy	Federated Learning
Fokus	Hasil / statistik	Arsitektur pelatihan
Data mentah	Bisa di pusat	Tetap di lokal
Teknik utama	<i>Noise injection</i>	<i>Distributed training</i>
Cocok untuk	Analytics, publikasi data	AI personal & sensitif

Mindset yang Harus Dibawa Pulang

- **Privasi bukan fitur tambahan**, tapi bagian dari desain
- *AI* yang hebat adalah *AI* yang **tidak rakus data**
- *Privacy-Preserving AI* bukan mengurangi kemampuan, tapi **meningkatkan kepercayaan**

Challenge untuk pembaca:

Coba pikirkan satu sistem *AI* yang pernah kamu bangun.

👉 **Bagian mana yang paling berisiko bocor data?**

👉 **Apakah bisa diganti dengan DP, FL, atau kombinasi keduanya?**

PRAKTEK 12: Privacy-Preserving AI (Differential Privacy)

Tujuan

Di akhir sesi ini, mahasiswa bisa:

- Memahami **trade-off: privasi vs akurasi (utility)**.
- Menerapkan *differential privacy* (DP) pada analitik sederhana (mean, count, histogram).
- Mengukur **utility loss** (mis. MAE/RMSE) setelah diberi noise.
- Menghasilkan **dataset/hasil analitik yang lebih aman** untuk dibagikan.
- (Opsional) Mengamankan file hasil rilis memakai **GnuPG** (enkripsi & tanda tangan).

Inti mindset: **“Kalau datanya akan dibagi, asumsi terburuk: musuh punya informasi tambahan.”** DP membantu mengurangi risiko “kebocoran lewat statistik”.

Konsep Inti yang Perlu Dipahami

1. Apa itu Differential Privacy?

Secara sederhana: sebuah mekanisme memenuhi *differential privacy* jika **hasil rilis tidak berubah drastis** walaupun **1 orang** di dataset **ditambahkan atau dihapus**.

- Parameter utama: **ϵ (epsilon)**
 - **ϵ kecil** $\Rightarrow$ privasi lebih kuat $\Rightarrow$ noise lebih besar $\Rightarrow$ akurasi turun
 - **ϵ besar** $\Rightarrow$ privasi lebih lemah $\Rightarrow$ noise kecil $\Rightarrow$ akurasi naik

2. Kenapa “Noise Injection” masuk akal?

Karena banyak kebocoran privasi terjadi bukan dari “membuka row data”, tapi dari:

- rilis *mean / count / histogram*
- dashboard statistik UMKM/kampus/kelas
- laporan “rata-rata nilai” + filter kecil (mis. per prodi, per angkatan)

DP menambahkan noise terukur agar **statistik tetap berguna**, tapi **sulit dipakai untuk menebak data individu**.

3. Istilah penting (biar tidak bingung)

- *Sensitivity*: seberapa besar output statistik bisa berubah jika 1 data berubah.
- *Laplace mechanism*: noise dari distribusi Laplace (umum untuk count/mean).

- *Clipping*: membatasi nilai agar sensitivity terkendali (praktik wajib).

Rule utama DP di praktikum ini: sebelum DP, lakukan **clipping** agar sensitivity jelas.

Tools (Open Source)

- Ubuntu 24.04
- Python 3 (venv, pip)
- Library: **numpy**, **pandas**
- (Opsional) **matplotlib** untuk visualisasi
- (Opsional) **GnuPG** untuk mengamankan file rilis

Skenario Real yang Bisa Diimplementasi

Bayangkan Anda punya dataset internal:

- **age** (umur)
- **income** (pendapatan)
- **score** (nilai)
- **department** (kategori)

Anda ingin merilis:

1. **Rata-rata pendapatan** per kelas/kelompok
2. **Jumlah orang** per kategori
3. **Histogram** nilai untuk laporan

Tanpa DP, mean/count bisa jadi celah (mis. *difference attack* saat seseorang tahu semua orang kecuali 1).

LANGKAH PRAKTIKUM (Ubuntu 24.04)

Step 0 — Setup Environment

```
sudo apt update
sudo apt install -y python3 python3-venv python3-pip gnupg
mkdir -p praktek12_dp && cd praktek12_dp
python3 -m venv .venv
source .venv/bin/activate
pip install -U pip
pip install numpy pandas matplotlib
```

Step 1 — Siapkan Dataset Contoh (atau pakai data Anda)

Kita buat dataset sintetis supaya aman untuk latihan, tapi strukturnya realistis.

Buat file: `generate_dataset.py`

```
import numpy as np
import pandas as pd

def main():
    rng = np.random.default_rng(42)
    n = 2000

    dept = rng.choice(["TI", "SI", "DKV", "Bisnis"], size=n,
p=[0.35, 0.30, 0.20, 0.15])
    age = rng.integers(18, 45, size=n)

    # income realistis (lognormal) dalam juta rupiah
    income = rng.lognormal(mean=2.2, sigma=0.45, size=n) # ~
5-20 jutaan
    income = income * 1_000_000

    # score 0-100
    score = np.clip(rng.normal(loc=78, scale=10, size=n), 0,
100)

    df = pd.DataFrame({
        "department": dept,
        "age": age,
        "income": income.round(0).astype(int),
        "score": score.round(1),
    })

    df.to_csv("raw_dataset.csv", index=False)
    print("OK: raw_dataset.csv dibuat. Jumlah row:", len(df))
    print(df.head(5))

if __name__ == "__main__":
    main()
```

Jalankan:

```
python generate_dataset.py
```

Step 2 — Implementasi Differential Privacy (Noise Injection + Clipping)

Kita akan:

- **clip** income ke rentang wajar (mis. 0..50 juta)
- hitung mean/count/histogram
- tambahkan noise Laplace dengan parameter ϵ
- bandingkan utility (error)

Buat file: `dp_release.py`

```
import argparse
import numpy as np
import pandas as pd

def laplace_noise(scale: float, rng: np.random.Generator) -> float:
    # Laplace(0, b) => scale = b
    return rng.laplace(0.0, scale)

def dp_count(true_count: int, epsilon: float, rng: np.random.Generator) -> int:
    # sensitivity untuk count = 1
    scale = 1.0 / epsilon
    noisy = true_count + laplace_noise(scale, rng)
    return int(max(0, round(noisy)))

def dp_mean(values: np.ndarray, epsilon: float, clip_min: float, clip_max: float, rng: np.random.Generator) -> float:
    """
    DP mean via Laplace:
    - clip values => sensitivity mean = (clip_max - clip_min) / n
    - add Laplace noise to mean
    """
    clipped = np.clip(values, clip_min, clip_max)
    n = len(clipped)
    if n == 0:
        return float("nan")

    true_mean = float(np.mean(clipped))
    sensitivity = (clip_max - clip_min) / n
    scale = sensitivity / epsilon
    noisy_mean = true_mean + laplace_noise(scale, rng)
    return noisy_mean

def dp_histogram(values: np.ndarray, bins: np.ndarray, epsilon: float, rng: np.random.Generator) -> np.ndarray:
    """
    DP histogram:
    - hitung count per bin
    - tambahkan Laplace noise ke setiap bin
    Catatan: ini menghabiskan budget DP per bin (komposisi).
    Untuk praktikum, kita pakai epsilon_per_bin = epsilon / k
    """
    counts, _ = np.histogram(values, bins=bins)
    k = len(counts)
    epsilon_per_bin = epsilon / max(1, k)

    noisy = []
    for c in counts:
        noisy.append(dp_count(int(c), epsilon_per_bin, rng))
```

```

        return np.array(noisy, dtype=int)

def utility_metrics(true_arr: np.ndarray, noisy_arr: np.ndarray)
-> dict:
    err = noisy_arr.astype(float) - true_arr.astype(float)
    mae = float(np.mean(np.abs(err)))
    rmse = float(np.sqrt(np.mean(err**2)))
    return {"MAE": mae, "RMSE": rmse}

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("--input", default="raw_dataset.csv")
    parser.add_argument("--epsilon", type=float, default=1.0)
    parser.add_argument("--seed", type=int, default=123)
    parser.add_argument("--clip_income_max", type=float,
default=50_000_000.0) # 50 juta
    args = parser.parse_args()

    rng = np.random.default_rng(args.seed)
    df = pd.read_csv(args.input)

    eps = args.epsilon
    if eps <= 0:
        raise ValueError("epsilon harus > 0")

    # ===== RELEASE 1: DP Count per department =====
    dept_counts = df["department"].value_counts().sort_index()
    true_counts = dept_counts.values

    noisy_counts = np.array([dp_count(int(c), eps, rng) for c in
true_counts], dtype=int)

    # ===== RELEASE 2: DP Mean income (global) =====
    income = df["income"].to_numpy(dtype=float)
    dp_income_mean = dp_mean(income, eps, clip_min=0.0,
clip_max=args.clip_income_max, rng=rng)
    true_income_mean = float(np.mean(np.clip(income, 0.0,
args.clip_income_max)))

    # ===== RELEASE 3: DP Histogram score =====
    score = df["score"].to_numpy(dtype=float)
    bins = np.array([0,10,20,30,40,50,60,70,80,90,100],
dtype=float)
    true_hist, _ = np.histogram(score, bins=bins)
    dp_hist = dp_histogram(score, bins=bins, epsilon=eps,
rng=rng)

    # ===== Utility evaluation =====
    count_metrics = utility_metrics(true_counts, noisy_counts)
    hist_metrics = utility_metrics(true_hist, dp_hist)

    result = {
        "epsilon": eps,

```

```

        "release": {
            "dp_count_per_department": {
                "departments": dept_counts.index.tolist(),
                "true_counts": true_counts.tolist(),
                "noisy_counts": noisy_counts.tolist(),
                "utility": count_metrics,
            },
            "dp_mean_income": {
                "clip_max": args.clip_income_max,
                "true_mean": true_income_mean,
                "noisy_mean": dp_income_mean,
                "abs_error": abs(dp_income_mean -
true_income_mean),
            },
            "dp_hist_score": {
                "bins": bins.tolist(),
                "true_hist": true_hist.tolist(),
                "noisy_hist": dp_hist.tolist(),
                "utility": hist_metrics,
            }
        }
    }

    # Simpan "dataset aman" versi rilis (bukan raw rows!)
    # Konsep penting: seringkali yang aman itu *released
statistics*, bukan "anonymized rows"
    import json
    with open("dp_release.json", "w") as f:
        json.dump(result, f, indent=2)

    print("OK: dp_release.json dibuat.")
    print("Ringkas:")
    print("- epsilon:", eps)
    print("- mean income true vs noisy:", true_income_mean,
"=>", dp_income_mean)
    print("- count utility (MAE/RMSE):", count_metrics)
    print("- hist utility (MAE/RMSE):", hist_metrics)

if __name__ == "__main__":
    main()

```

Jalankan beberapa eksperimen epsilon:

```

python dp_release.py --epsilon 0.2
python dp_release.py --epsilon 1.0
python dp_release.py --epsilon 5.0

```

Yang harus kamu amati:

- $\epsilon=0.2 \rightarrow$ **privasi kuat**, noise besar, error naik
- $\epsilon=5.0 \rightarrow$ privasi lemah, noise kecil, error turun

Step 3 — Bandingkan Utility Secara “Data Scientist”

Biar fun, bikin tabel perbandingan. Jalankan:

```
for e in 0.2 0.5 1 2 5; do
  echo "==== EPSILON $e ===="
  python dp_release.py --epsilon $e | tail -n 6
done
```

Tantangan (menantang tapi doable):

- Tentukan epsilon “layak rilis” untuk laporan kampus/UMKM.
- Target utility misalnya:
 - MAE count per kategori < 10
 - Error mean < 3% dari true mean

Output Praktikum (Yang Dikumpulkan)

Mahasiswa mengumpulkan:

1. `dp_release.json` (hasil rilis DP)
2. Laporan singkat (1–2 halaman) berisi:
 - nilai ϵ yang dicoba
 - perbandingan error (MAE/RMSE) + interpretasi
 - rekomendasi ϵ final dan alasan
 - catatan **trade-off** yang terjadi

Catatan penting: yang Anda rilis adalah *privacy-preserving analytics*, bukan “dataset baris-per-baris”.

BONUS: Amankan File Rilis dengan GnuPG (Enkripsi + Tanda Tangan)

Ini realistis banget: Anda mau kirim `dp_release.json` ke dosen/tim.

1. Generate key (sekali saja)

```
gpg --full-generate-key
```

Pilih:

- jenis: RSA and RSA
- size: 3072 atau 4096
- isi nama & email

Cek:

```
gpg --list-keys
```

2. Enkripsi file untuk penerima (recommended)

Misal penerima punya email `dosen@example.ac.id` (harus sudah ada public key-nya di keyring Anda).

Import public key dulu jika perlu:

```
gpg --import dosen_publickey.asc
```

Enkripsi:

```
gpg --encrypt --recipient dosen@example.ac.id  
dp_release.json  
# hasil: dp_release.json.gpg
```

3. Decrypt (di sisi penerima)

```
gpg --decrypt dp_release.json.gpg > dp_release_decrypted.json
```

4. Tanda tangan file (integrity)

```
gpg --detach-sign dp_release.json
```

hasil: dp_release.json.sig

Verifikasi:

```
gpg --verify dp_release.json.sig dp_release.json
```

Diskusi Kelas

Pertanyaan pemantik:

- Kalau Anda rilis mean income per departemen, **berapa minimal ukuran kelompok** agar aman?
- Apa yang terjadi kalau filter dibuat terlalu spesifik (mis. “TI semester 1 kelas A”)?
- Apakah DP cukup? Kapan perlu tambahan: *access control*, audit log, dan encryption?

Takeaway yang wajib dibawa pulang:

- **Privasi itu desain sistem**, bukan cuma “hapus kolom NIK”.
- DP adalah alat untuk membuat rilis statistik **lebih aman dan terukur**.

BAB 13: Proyek Akhir (AI Security Project)

Bab ini adalah **puncak dari seluruh perjalanan pembelajaran keamanan informasi**. Jika pada bab-bab sebelumnya mahasiswa belajar konsep, tools, dan praktik secara terpisah, maka pada **Proyek Akhir** ini semuanya **dirangkai menjadi satu sistem utuh**. Di sinilah mahasiswa **berpikir sebagai security engineer**, bukan sekadar menjalankan perintah.

Proyek ini dirancang agar mahasiswa:

- mengidentifikasi masalah keamanan nyata,
- merancang solusi berbasis AI,
- mengimplementasikan solusi dengan open-source tools, dan
- menjelaskan logika keamanannya secara rasional dan dapat dipertanggungjawabkan.

Tidak ada jawaban tunggal yang benar. Yang dinilai bukan kecanggihan AI, tetapi ketepatan desain, logika keamanan, dan pemahaman risiko.

Tujuan

Setelah menyelesaikan proyek ini, mahasiswa diharapkan mampu:

- membangun solusi keamanan end-to-end, dari data → model → analisis → rekomendasi
- memahami peran AI sebagai alat bantu keputusan, bukan pengganti manusia
- menjelaskan batasan, risiko, dan potensi kesalahan AI
- menghubungkan keamanan teknis dengan konteks sosial dan regulasi (PDP)

Security is not about tools. It is about decisions.

Kerangka Umum Proyek

Setiap pilihan proyek **HARUS** mengikuti alur berikut:

1. **Problem Definition**
Apa masalah keamanannya? Siapa yang terdampak? Apa risikonya?
2. **Data Understanding**
Data apa yang digunakan? Apakah mengandung *personal data*? Bagaimana risikonya?
3. **AI / Logic Design**
Apakah menggunakan *rule-based*, *machine learning*, atau *hybrid*?
4. **Implementation**
Script Python, pipeline sederhana, dan dokumentasi jelas.

5. Evaluation & Limitation

Di mana AI bisa salah? Apa dampaknya?

6. Security & Ethical Reflection

Apakah solusi ini aman, etis, dan realistis?

Pilihan Proyek

Mahasiswa **memilih satu** dari empat proyek berikut. Semua proyek **setara nilainya**, yang berbeda hanyalah **sudut pandang keamanan**.

AI Deteksi Phishing

Fokus: Perlindungan pengguna dari manipulasi sosial

Phishing adalah serangan paling umum dan paling efektif karena **menyerang manusia, bukan sistem**. Proyek ini menantang mahasiswa untuk membangun **AI sederhana yang membantu mendeteksi pesan phishing** berdasarkan konten teks.

Mahasiswa akan belajar bahwa:

- AI tidak memahami niat, hanya pola
- **false positive** dan **false negative** adalah risiko nyata
- deteksi phishing **bukan sekadar klasifikasi teks**, tetapi soal konteks

Implementasi dapat mencakup:

- analisis kata kunci mencurigakan (*urgent, verify now, account suspended*)
- deteksi pola URL tidak wajar
- scoring risiko pesan

Yang penting:

Mahasiswa **WAJIB menjelaskan** mengapa suatu pesan dianggap berbahaya, bukan hanya menampilkan hasil *prediction*.

AI Audit Data Pribadi

Fokus: Kepatuhan & kesadaran Perlindungan Data Pribadi

Banyak sistem menyimpan data tanpa sadar bahwa data tersebut **sensitif dan berisiko hukum**. Proyek ini mengajak mahasiswa membuat **AI atau script cerdas** untuk memindai data dan **mengidentifikasi potensi pelanggaran PDP**.

Mahasiswa akan menghadapi pertanyaan kritis:

- Apa yang disebut *personal data*?
- Apakah email, IP address, atau log termasuk data pribadi?
- Bagaimana jika data terenkripsi tapi disimpan tanpa izin?

Contoh kemampuan sistem:

- mendeteksi kolom sensitif (NIK, email, nomor HP)
- memberi peringatan tingkat risiko
- menyarankan mitigasi (*hashing, encryption, access control*)

Proyek ini menekankan bahwa:

Security tanpa privacy adalah ilusi keamanan.

AI IDS Sederhana

Fokus: Deteksi anomali dan serangan jaringan

Di proyek ini mahasiswa membangun **Intrusion Detection System berbasis AI sederhana**. Tidak perlu real-time atau kompleks — yang penting adalah **logika deteksi dan interpretasi hasil**.

Mahasiswa akan belajar bahwa:

- *Anomaly detection ≠ attack detection*
- AI bisa salah membaca trafik normal sebagai serangan
- konteks jaringan sangat menentukan hasil

Contoh pendekatan:

- membaca log koneksi (IP, port, frekuensi)
- mendeteksi pola tidak normal
- memberi skor risiko koneksi

Penekanan utama:

- **AI tidak menggantikan IDS konvensional**, tetapi **melengkapi**
- hasil AI **HARUS bisa dijelaskan secara teknis**

AI Privacy Checker

Fokus: Transparansi & kesadaran pengguna

Proyek ini berorientasi pada **user awareness**. Mahasiswa membuat sistem yang membantu pengguna **memahami risiko privasi dari data atau sistem yang mereka gunakan**.

Contoh implementasi:

- analisis file konfigurasi
- pemeriksaan permission aplikasi
- deteksi data sensitif dalam folder/log

Sistem tidak harus otomatis memblokir, tetapi:

- **memberi peringatan**
- **menjelaskan risiko**
- **memberi rekomendasi praktis**

Nilai utama proyek ini adalah:

Membantu manusia membuat keputusan yang lebih sadar.

Output yang Wajib Dikumpulkan

Setiap proyek **HARUS** menghasilkan:

- **Source code (Python, open-source only)**
- **README / laporan singkat** berisi:
 - tujuan sistem
 - desain logika
 - hasil pengujian
 - keterbatasan sistem
- **Refleksi keamanan & etika (1–2 halaman)**

Catatan Penting untuk Mahasiswa

- **AI bukan solusi ajaib**
- Sistem sederhana dengan penjelasan bagus **lebih bernilai** daripada model kompleks tanpa pemahaman
- *Overclaiming* adalah kesalahan fatal
- **Transparansi lebih penting daripada akurasi tinggi yang tidak bisa dijelaskan**

Penutup

Bab ini bukan tentang membuat AI paling pintar.

Bab ini tentang **menjadi engineer yang bertanggung jawab**.

Jika setelah proyek ini mahasiswa:

- lebih kritis terhadap klaim AI,
- lebih sadar risiko data,
- dan lebih berhati-hati dalam desain sistem,

maka proyek ini **telah berhasil**.

Good security is invisible.

Great security is understandable.

PRAKTEK 13 — Proyek Akhir AI Security

Tujuan

Di praktikum ini kamu tidak lagi “coba-coba tool”. Kamu akan membangun **produk keamanan mini** yang:

- punya **input data jelas**
- punya **proses deteksi**
- punya **output keputusan (risk score + alasan)**
- punya **laporan & demo yang bisa dipertanggungjawabkan**

Kuncinya: **jelaskan logika keamanan**. AI hanya membantu.

Pilihan Proyek (Pilih 1)

1. **AI Phishing Detector** (paling “nyata”, mudah diuji)
2. **AI Audit PDP** (privacy compliance, cocok untuk log/CSV/dataset)
3. **AI IDS Sederhana** (network log/anomaly, menantang tapi seru)

Semua proyek punya kerangka sama (end-to-end).

Struktur Wajib Proyek (Sama untuk semua)

Tahap 0 — Setup Environment (Ubuntu 24.04)

Instal dependensi dasar

```
sudo apt update
sudo apt install -y python3 python3-venv python3-pip git gpg
python3 --version
gpg --version
```

Buat folder proyek

```
mkdir -p ~/ai-security-final/{data,src,reports,models}
cd ~/ai-security-final
python3 -m venv .venv
source .venv/bin/activate
pip install --upgrade pip
```

Paket Python (open-source)

Untuk semua opsi proyek:

```
pip install pandas scikit-learn numpy joblib rich
```

Opsional (kalau butuh parsing log lebih rapih / regex kuat):

```
pip install python-dateutil
```

Struktur folder minimal:

```
ai-security-final/  
  data/  
  src/  
  models/  
  reports/  
  README.md
```

Tahap 1 — Keamanan Data Proyek (Wajib): GnuPG untuk Dataset & Output

Kenapa? Karena dataset dan laporan sering mengandung **data sensitif**. Kamu harus membuktikan bahwa kamu bisa **mengamankan data**.

1. Buat key GPG (untuk proyek)

```
gpg --full-generate-key
```

Pilih:

- (1) RSA and RSA
- 3072 atau 4096
- nama: **AI Security Student**
- email: **student@lab.local**

Cek key:

```
gpg --list-keys
```

2. Enkripsi dataset (contoh)

Misal dataset kamu **data/phishing_samples.csv**:

```
gpg --output data/phishing_samples.csv.gpg --symmetric  
--cipher-algo AES256 data/phishing_samples.csv  
shred -u data/phishing_samples.csv
```

Decrypt saat butuh:

```
gpg --output data/phishing_samples.csv --decrypt  
data/phishing_samples.csv.gpg
```

Aturan proyek: dataset yang berisi data pribadi/berisiko **harus disimpan terenkripsi** atau minimal data dummy.

Tahap 2 — Pilih Proyek + Jalankan Step-by-step

Di bawah ini saya kasih **3 jalur proyek lengkap**, masing-masing punya:

- data contoh realistis
- langkah implementasi
- kode training + inference
- output demo
- format laporan

Kamu tinggal pilih salah satu.

OPSI A — AI Phishing Detector (Recommended)

Goal: deteksi pesan *phishing* dari teks email/chat → keluarkan **label + risk score + alasan**.

1. Siapkan dataset (realistis tapi aman)

Buat file: `data/phishing_samples.csv` (contoh mini, bisa kamu tambah)

```
text,label
"URGENT: Your account will be suspended. Verify now at
http://secure-login.example.com",1
"Hi team, meeting moved to 3pm. Link:
https://meet.example.org/abc",0
"Reset password now. Your mailbox is full. Click
http://mailbox-reset.example.net",1
"Invoice attached, please review. Thanks",0
"Bank: unusual activity detected. Confirm your OTP at
http://bank-verify.example.xyz",1
"Reminder: submit assignment before Friday",0
```

Label: 1=phishing, 0=benign.

Challenge: nanti kamu tambahkan 50–200 contoh (bisa dari teks buatan sendiri yang realistis).

2. Buat training script (ML sederhana + explainable)

Buat `src/train_phishing.py`:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import
TfidfVectorizer
from sklearn.linear_model import LogisticRegression
```

```

from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report,
confusion_matrix
import joblib

DATA_PATH = "data/phishing_samples.csv"
MODEL_PATH = "models/phishing_model.joblib"

def main():
    df = pd.read_csv(DATA_PATH)
    X = df["text"].astype(str)
    y = df["label"].astype(int)

    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.3, random_state=42, stratify=y
    )

    model = Pipeline([
        ("tfidf", TfidfVectorizer(ngram_range=(1,2),
min_df=1)),
        ("clf", LogisticRegression(max_iter=200))
    ])

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    print("=== Confusion Matrix ===")
    print(confusion_matrix(y_test, y_pred))
    print("\n=== Classification Report ===")
    print(classification_report(y_test, y_pred))

    joblib.dump(model, MODEL_PATH)
    print(f"\nSaved model to: {MODEL_PATH}")

if __name__ == "__main__":
    main()

```

Jalankan:

```
python src/train_phishing.py
```

3. Buat detector + alasan (top keywords)

Buat `src/detect_phishing.py`:

```

import joblib
from rich import print
from rich.console import Console

MODEL_PATH = "models/phishing_model.joblib"

SUSPICIOUS_HINTS = [

```

```

        "urgent", "verify", "reset", "suspended", "otp",
        "password",
        "click", "confirm", "limited", "account", "bank"
    ]

def explain_text(text: str):
    low = text.lower()
    hits = [h for h in SUSPICIOUS_HINTS if h in low]
    return hits[:10]

def main():
    model = joblib.load(MODEL_PATH)

    console = Console()
    console.print("[bold]AI Phishing Detector
Demo[/bold]")
    console.print("Ketik pesan/email. Enter kosong untuk
keluar.\n")

    while True:
        text = input("Message> ").strip()
        if not text:
            break

        proba = model.predict_proba([text])[0][1] # prob
phishing
        label = "PHISHING" if proba >= 0.5 else "BENIGN"
        hints = explain_text(text)

        print("\n[bold]Result[/bold]")
        print(f"Label      : [bold]{label}[/bold]")
        print(f"Risk score: [bold]{proba:.2f}[/bold]
(0..1)")
        print(f"Reasons    : {hints if hints else 'No
obvious keyword hints'}")
        print("-" * 60)

    if __name__ == "__main__":
        main()

```

Run demo:

```
python src/detect_phishing.py
```

Contoh input nyata untuk demo:

- “Admin: akun kamu akan nonaktif. klik link ini untuk verifikasi ...”
- “Tolong cek invoice, ada file .zip passwordnya 12345”
- “Meeting jam 2, link google meet ...”

Penilaian tinggi kalau kamu menambahkan:
deteksi URL pendek, domain aneh, kata “urgent”, dan *pattern*
yang sering dipakai scam.

OPSI B — AI Audit PDP (Privacy Audit Tool)

Goal: scan file CSV/log → deteksi *personal data* → laporan risiko + rekomendasi.

1. Dataset contoh

Buat `data/sample_users.csv`:

```
name,email,phone,nik,address,notes
"Budi","budi@mail.com","08123456789","327xxxxxxxxxxxx","Be
kasi","token=abc123"
"Siti","siti@gmail.com","082233445566","320xxxxxxxxxxxx","
Jakarta","pwd=123456"
"Andi","andi@corp.co.id","081299988877","","Bandung","no
issues"
```

2. Tool audit (regex + scoring)

Buat `src/pdp_audit.py`:

```
import re
import pandas as pd
from rich import print
from rich.table import Table

PATTERNS = {
    "EMAIL": re.compile(r"\b[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\b"),
    "PHONE_ID": re.compile(r"\b(08\d{8,12})\b"),
    "NIK_LIKE": re.compile(r"\b\d{16}\b"),
    "TOKEN_LIKE": re.compile(r"\b(token|apikey|secret)\s*=\s*[A-Za-z0-9_-]{6,}\b", re.I),
    "PASSWORD_LIKE": re.compile(r"\b(pass|pwd|password)\s*=\s*\S+\b", re.I),
}

SEVERITY = {
    "EMAIL": 2,
    "PHONE_ID": 2,
    "NIK_LIKE": 4,
    "TOKEN_LIKE": 4,
    "PASSWORD_LIKE": 5,
}

def scan_value(val: str):
    findings = []
```

```

for k, rx in PATTERNS.items():
    if rx.search(val):
        findings.append(k)
return findings

def main():
    path = "data/sample_users.csv"
    df = pd.read_csv(path)

    findings_rows = []
    total_score = 0

    for idx, row in df.iterrows():
        row_findings = []
        row_score = 0
        for col, v in row.items():
            s = "" if pd.isna(v) else str(v)
            hits = scan_value(s)
            for h in hits:
                row_findings.append((col, h))
                row_score += SEVERITY[h]
        total_score += row_score
        findings_rows.append((idx, row_score,
row_findings))

    table = Table(title="PDP Audit Report (Quick Scan)")
    table.add_column("Row", justify="right")
    table.add_column("Risk Score", justify="right")
    table.add_column("Findings")

    for idx, score, f in findings_rows:
        pretty = ", ".join([f"{col}:{tag}" for col, tag in
f]) if f else "-"
        table.add_row(str(idx), str(score), pretty)

    print(table)
    print(f"\n[bold]Total Risk Score:[/bold]
{total_score}")

    if total_score >= 20:
        print("[bold red]High risk:[/bold red] segera
lakukan masking/encryption & akses kontrol.")
    elif total_score >= 10:
        print("[bold yellow]Medium risk:[/bold yellow]
audit consent + minimisasi data.")
    else:
        print("[bold green]Low risk:[/bold green] tetap
pastikan retention & akses log.")

if __name__ == "__main__":
    main()

```

Run:

```
python src/pdp_audit.py
```

Bonus nilai: hasil audit disimpan jadi file `reports/pdp_report.txt` lalu **dienkripsi pakai GPG**.

OPSI C — AI IDS Sederhana (Anomaly Detection dari Log)

Goal: baca log koneksi → deteksi “aneh” (contoh: port scanning / brute force) → alert.

1. Buat dataset log sederhana (contoh realistis)

Buat `data/connections.csv`:

```
src_ip,dst_port,count_per_minute
192.168.1.10,80,5
192.168.1.11,22,3
192.168.1.50,22,60
192.168.1.50,23,55
192.168.1.50,445,40
192.168.1.12,443,4
192.168.1.13,80,6
```

Interpretasi:

- IP `192.168.1.50` “rame banget” ke banyak port → *scan/bruteforce suspicion*

2. Anomaly detection dengan IsolationForest

Buat `src/ids_anomaly.py`:

```
import pandas as pd
from sklearn.ensemble import IsolationForest
from rich import print
from rich.table import Table

def main():
    df = pd.read_csv("data/connections.csv")

    # features sederhana
    X = df[["dst_port", "count_per_minute"]].astype(float)

    model = IsolationForest(contamination=0.2,
                             random_state=42)
    df["anomaly"] = model.fit_predict(X) # -1 anomaly, 1
    normal
    df["score"] = model.decision_function(X) # semakin
    kecil = semakin aneh
```

```

        table = Table(title="AI IDS Sederhana (Anomaly
Detection)")
        table.add_column("src_ip")
        table.add_column("dst_port", justify="right")
        table.add_column("count/min", justify="right")
        table.add_column("anomaly")
        table.add_column("score", justify="right")

        for _, r in df.sort_values("score").iterrows():
            tag = "[bold red]ALERT[/bold red]" if r["anomaly"]
            == -1 else "OK"
            table.add_row(
                str(r["src_ip"]),
                str(int(r["dst_port"])),
                str(int(r["count_per_minute"])),
                tag,
                f"{r['score']:.3f}"
            )

        print(table)

        alerts = df[df["anomaly"] == -1]
        if len(alerts) > 0:
            print("\n[bold]Suggested Investigation
Steps:[/bold]")
            print("- Cek apakah IP itu user normal atau device
tak dikenal")
            print("- Cek log auth (/var/log/auth.log) jika
port 22 dominan")
            print("- Jika banyak port berbeda: kemungkinan
port scanning")
        else:
            print("\n[bold green]No anomaly detected[/bold
green]")

if __name__ == "__main__":
    main()

```

Run:

```
python src/ids_anomaly.py
```

Bonus nilai: integrasikan dengan log nyata dari **auth.log** atau **ufw.log** (tanpa data pribadi).

Tahap 3 — Output Wajib: Tool + Laporan + Demo

1. Tool (Wajib)

Tool kamu minimal bisa:

- menerima input (file / teks)
- menghasilkan output (label/alert/report)
- punya cara menjalankan yang jelas: `python src/...`

2. Laporan (Template singkat tapi kuat)

Buat `reports/report.md` (minimal isi):

- **Latar belakang masalah**
- **Threat model ringkas** (siapa attacker, target, impact)
- **Desain sistem**
- **Dataset & pengamanan data** (pakai GPG? masking?)
- **Hasil uji** (contoh output, confusion matrix / alert list)
- **Limitasi & potensi kesalahan**
- **Rekomendasi perbaikan**

3. Demo (Wajib)

Demo 3–5 menit:

- jelaskan masalah
- jalankan tool live
- tunjukkan output
- jelaskan “kenapa” hasilnya begitu

Rubrik Penilaian

Kalau mau nilai tinggi, fokus ke ini:

- **End-to-end berfungsi** (bukan potongan kode)
- output ada **risk score + alasan**
- ada **evaluasi & limitation** (AI bisa salah di mana)
- data sensitif ditangani: **mask/encrypt (GPG)**
- dokumentasi rapi: README + report

Checklist Final (Sebelum Submit)

- `src/` berisi script utama
- `data/` aman (dummy atau terenkripsi GPG)
- `models/` ada model kalau proyek ML
- `reports/report.md` ada
- Demo bisa jalan di Ubuntu 24.04 dengan perintah jelas
- Semua **open-source**, tanpa proprietary

BAB 14: Review, Etika & Masa Depan

Pada pertemuan penutup ini, mahasiswa diajak **berhenti sejenak dari teknis murni** dan melihat gambaran besar: **bagaimana AI security mempengaruhi manusia, organisasi, dan masyarakat**. Setelah belajar *bagaimana membangun, bagaimana menyerang, dan bagaimana melindungi* sistem AI, kini pertanyaannya adalah “**haruskah?**”, “**siapa yang bertanggung jawab?**”, dan “**ke mana profesi ini akan bergerak?**”.

Tujuannya bukan membuat mahasiswa takut, tetapi **menumbuhkan kesadaran profesional**—bahwa keahlian teknis harus selalu dibarengi **etika, tata kelola, dan visi masa depan**.

Teori 1 — Ethics of AI Security

Etika AI Security membahas **batasan moral** dalam merancang, menguji, dan menerapkan sistem keamanan berbasis AI. AI itu **powerful**, tapi **tidak netral**—ia membawa nilai, bias, dan konsekuensi.

Prinsip inti yang wajib dipegang profesional AI security:

- **Accountability (Tanggung jawab)**
Siapa yang bertanggung jawab jika AI salah mendeteksi dan merugikan orang?
- **Transparency (Keterbukaan)**
Apakah keputusan AI bisa dijelaskan (*explainability*)?
- **Privacy by Design**
Data pribadi **bukan bahan bakar gratis** untuk melatih model.
- **Fairness (Keadilan)**
AI tidak boleh mendiskriminasi kelompok tertentu.

Contoh nyata (real-world):

- Sistem deteksi penipuan AI salah menandai **akun UMKM** sebagai fraud → rekening diblokir → bisnis lumpuh.
- Model keamanan internal perusahaan memantau karyawan **terlalu invasif**, melanggar privasi.

Pesan kunci:

AI yang aman tapi tidak etis tetap berbahaya.

Teori 2 — AI Abuse & Governance

AI tidak hanya alat pertahanan—ia juga bisa **disalahgunakan (AI abuse)**. Professional security **harus memahami sisi gelap ini** agar bisa mencegahnya.

Bentuk penyalahgunaan AI yang umum:

- *Automated phishing* dengan bahasa sangat natural

- *Deepfake* untuk penipuan, pemerasan, atau manipulasi politik
- AI untuk *password guessing* dan *social engineering*
- Model AI bocor data sensitif (*model leakage*)

Di sinilah peran **AI governance**:

- Aturan **siapa boleh melatih model**
- Aturan **data apa yang boleh dipakai**
- Audit model & dataset secara berkala
- *Human-in-the-loop* untuk keputusan kritikal

Mini checklist governance (praktis):

- ☒ Apakah dataset sudah disaring PII?
- ☒ Apakah ada *logging & audit trail*?
- ☒ Apakah ada prosedur *incident response* khusus AI?
- ☒ Apakah pengguna diberi *consent* yang jelas?

Pesan kunci:

Tanpa governance, AI security berubah dari pelindung menjadi ancaman.

Teori 3 — Masa Depan Profesi Security

Profesi keamanan **tidak akan hilang**, tapi **akan berubah drastis**. AI bukan menggantikan manusia—AI **menggeser peran manusia ke level yang lebih strategis**.

Perubahan besar yang akan terjadi:

- Dari *manual analyst* → **AI security orchestrator**
- Dari *rule writer* → **model & data curator**
- Dari *incident responder* → **risk & ethics advisor**

Peran baru yang mulai muncul:

- **AI Security Engineer**
- **AI Risk & Compliance Officer**
- **Model Auditor**
- **Privacy Engineer**
- **AI Red Team Specialist**

Skill yang makin penting ke depan:

- *Critical thinking* (bukan sekadar coding)
- Pemahaman regulasi & etika
- Data literacy
- Kolaborasi lintas disiplin (legal, bisnis, sosial)

Pesan kunci:

Profesional security masa depan adalah penjaga kepercayaan (*guardian of trust*).

Review & Refleksi (Bagian Fun & Menantang)

Ajak mahasiswa berdiskusi dan refleksi:

- Jika AI Anda **akurat tapi melanggar privasi**, apakah boleh dipakai?
- Jika klien meminta AI untuk *surveillance* berlebihan, apa sikap Anda?
- Apakah *open-source AI security* lebih etis daripada *closed-source*?
- Siapa yang salah jika AI mengambil keputusan keliru: developer, user, atau organisasi?

Latihan refleksi singkat:

Tuliskan **1 prinsip etika** yang akan selalu Anda pegang sebagai profesional AI security.

Penutup Inspiratif

AI security bukan sekadar soal **menangkap serangan** atau **meningkatkan akurasi model**. Ini adalah **profesi berbasis kepercayaan**.

Di tangan profesional yang bertanggung jawab, AI menjadi **pelindung masyarakat**.

Di tangan yang salah, AI menjadi **alat penindasan dan manipulasi**.

Ingat satu hal penting:

Skill membuat Anda dibutuhkan. Etika membuat Anda dipercaya.

PRAKTEK 14: Evaluasi & Etika

Fokus pertemuan ini adalah: setelah bikin tool keamanan berbasis AI (phishing detector / audit PDP / IDS / privacy checker), mahasiswa **belajar jadi profesional: mengukur kualitas, menguji keamanan, dan mengambil keputusan etis** yang bisa dipertanggungjawabkan. Ini bukan “bab teori doang”—ini tahap yang bikin proyek kalian **layak demo, layak dipakai, dan tidak membahayakan orang lain**.

Bayangin kamu mau deploy tool ke WA bot kampus / SOC mini / audit data dosen—kalau tool salah deteksi, bisa bikin panik, fitnah, atau kebocoran data. Maka mindset-nya: bayangkan kamu **security engineer + data steward**. Kalian akan bikin evaluasi yang rapi, lalu menulis refleksi profesional yang jujur, tajam, dan bisa dipresentasikan.

Tujuan

Setelah modul ini, mahasiswa mampu:

- **Mengevaluasi performa** sistem (akurasi, *precision*, *recall*, *F1*, false positive/false negative).
- **Mengevaluasi keamanan** sistem (risiko *model leakage*, input berbahaya, penyalahgunaan).
- **Mengevaluasi privasi & kepatuhan** (data minimization, masking, enkripsi).
- Membuat **refleksi etika profesional**: trade-off, mitigasi, batasan, dan rencana perbaikan.

Output wajib: 1 dokumen “Refleksi & Evaluasi Proyek” (2–4 halaman, 1 spasi) + lampiran hasil uji (screenshot/log) + repo/zip kode.

Aktivitas Utama (Flow Praktikum)

Kita pakai flow sederhana yang bisa dipakai untuk semua jenis proyek akhir:

Tahap 1 — Freeze Versi Proyek (Supaya Hasil Evaluasi Konsisten)

Tujuannya: saat dievaluasi, kode & environment tidak berubah-ubah.

Checklist:

- **Tag versi** di git (mis. `v1.0-eval`)
- Catat environment (Ubuntu 24.04, versi Python, dependency)
- Simpan dataset uji (tanpa data pribadi asli)

Contoh:

```
# di Ubuntu 24.04
python3 --version
pip3 freeze > requirements.txt

git status
git tag -a v1.0-eval -m "Evaluation baseline"
git show v1.0-eval
```

Tahap 2 — Uji Kualitas Model/Detektor (Metrics yang “security-minded”)

Di keamanan, **akurasi saja tidak cukup**. Yang paling penting seringkali:

- **False Negative (FN)**: serangan lolos (bahaya besar)
- **False Positive (FP)**: alarm palsu (bikin capek, menurunkan trust)

Minimal yang harus dihitung:

- Confusion matrix
- *precision, recall, F1*
- (opsional) ROC-AUC kalau model probabilistik

Contoh Kode Evaluasi (Umum, untuk classifier)

Misal output prediksi proyek kamu berupa label 0/1 dan ground-truth 0/1.

```
# eval_metrics.py
from sklearn.metrics import classification_report,
confusion_matrix

def evaluate(y_true, y_pred, label_positive=1):
    cm = confusion_matrix(y_true, y_pred, labels=[0,
label_positive])
    print("Confusion Matrix [TN FP; FN TP]:")
    print(cm)
    print("\nClassification Report:")
    print(classification_report(y_true, y_pred, digits=4))

if __name__ == "__main__":
    # contoh dummy (ganti dengan hasil proyekmu)
    y_true = [1,0,1,1,0,0,1,0,1,0]
    y_pred = [1,0,0,1,0,1,1,0,1,0]
    evaluate(y_true, y_pred)
```

Jalankan:

```
python3 -m pip install -U scikit-learn
python3 eval_metrics.py
```

Yang harus ditulis di laporan: kalau FN masih tinggi, kamu harus jelaskan **strategi mitigasi** (mis. threshold diturunkan, rules tambahan, review manual).

Tahap 3 — Uji Robustness (Input “Nakal” ala Attacker)

Sekarang masuk bagian yang fun: kalian jadi “attacker” buat ngetes sistem sendiri.

1. Untuk phishing detector

Coba variasi teks:

- obfuscation: “l0g1n”, “p4ssw0rd”, spasi aneh, unicode
- link shortener palsu
- “urgent tone” (“segera”, “akun diblokir”)

2. Untuk audit PDP / PII detector

Coba PII yang disamarkan:

- email: `nama [at] domain [dot] com`
- nomor: `08 12-3456-7890`
- NIK dengan spasi

3. Untuk IDS log detector

Coba log injection: masukkan karakter aneh, payload panjang, format rusak.

Contoh *fuzzer* input sederhana:

```
# fuzz_inputs.py
tests = [
    "login sekarang di http://bit.ly/xxx",
    "l0g1n skrg di hxxp://example[dot]com",
    "EMAIL: budi [at] mail [dot] com",
    "NIK: 3174 1234 5678 9012",
    "!!!" * 1000,
]

for t in tests:
    print("TEST:", t[:80], "...")
```

Target evaluasi: sistem kamu **tidak crash**, **tidak error**, dan tetap kasih output yang masuk akal (walau mungkin “unknown”).

Tahap 4 — Uji Privasi & Data Handling (PDP-friendly)

Di proyek keamanan AI, **data handling** itu separuh dari kualitas. Minimal harus ada:

- **Data minimization:** hanya ambil field yang perlu
- **Masking** saat logging/printing
- **Enkripsi** untuk file sensitif (kalau ada)

Contoh: Masking PII sebelum masuk log

```
# pii_mask.py
import re

EMAIL =
re.compile(r'\b[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\b')
)
PHONE = re.compile(r'\b(\+62|62|0)8[0-9][0-9 -]{7,12}\b')

def mask(text: str) -> str:
    text = EMAIL.sub("[EMAIL_REDACTED]", text)
    text = PHONE.sub("[PHONE_REDACTED]", text)
    return text

if __name__ == "__main__":
    sample = "Hubungi 0812-3456-7890 atau email budi@mail.com"
    print(mask(sample))
```

Contoh: Enkripsi laporan/dataset dengan *GnuPG* (Open Source)

Install:

```
sudo apt update
sudo apt install -y gnupg
```

Encrypt file (pakai passphrase):

```
gpg --symmetric --cipher-algo AES256 laporan_evaluasi.pdf
# hasil: laporan_evaluasi.pdf.gpg
```

Decrypt:

```
gpg --decrypt laporan_evaluasi.pdf.gpg >
laporan_evaluasi.pdf
```

Di laporan, tulis: **file apa yang dienkripsi, kenapa, dan siapa yang boleh akses.**

Tahap 5 — Etika AI & Keamanan (Diskusi yang “Real”, bukan normatif)

Di sini kalian bahas kasus nyata dan kaitkan ke proyek sendiri.

5 Studi Kasus yang Cocok untuk Diskusi Kelas

1. **False Positive bikin korban:** tool phishing kamu salah tandai email dosen → reputasi rusak.

Pertanyaan: *bagaimana mekanisme appeal/review?* **Siapa yang memutuskan?**

2. **False Negative bikin kebobolan:** IDS kamu gagal deteksi brute force.
Pertanyaan: *mana yang lebih kamu prioritaskan: FN kecil atau FP kecil?*
Kenapa?
3. **Data pribadi “nyangkut” di log:** tool audit menulis NIK/email ke log server.
Pertanyaan: *log retention berapa lama? siapa akses? Kenapa tidak dimasking?*
4. **Dual-use:** tool audit/IDS bisa dipakai untuk memata-matai.
Pertanyaan: *fitur apa yang harus dibatasi? apakah perlu role-based access?*
5. **Overclaiming:** presentasi demo bilang “AI kami akurat 99%” padahal dataset kecil.
Pertanyaan: *apa definisi “akurat”?* *apakah datanya representatif?* **Apa batasanmu?**

Output diskusi harus mengarah ke **aksi**: “kami akan menambahkan X untuk mencegah Y”.

Output Akhir: Template “Refleksi Tertulis”

Dokumen refleksi kalian sebaiknya punya struktur ini (biar terlihat profesional):

1. Ringkasan Proyek (1 paragraf)

- Apa toolsnya, input/output, target pengguna.

2. Dataset & Metodologi Evaluasi

- Sumber data (dummy/sintetik), ukuran, pembagian train/test (kalau ada).
- Skenario uji: normal vs adversarial.

3. Hasil Evaluasi (wajib angka)

- Confusion matrix + *precision/recall/F1*
- Jelaskan **FN & FP**: mana yang paling bahaya di konteks proyekmu.

4. Uji Robustness & Temuan

- Contoh input nakal
- Apakah tool crash? apakah output menyesatkan?

5. Privasi & Keamanan Data

- Apa yang di masking, apa yang dienkripsi (pakai *GnuPG*), kebijakan retensi.

6. Risiko Etika & Mitigasi

- Minimal 3 risiko + mitigasi nyata (fitur/prosedur)
- **Batasan**: apa yang tool kamu belum bisa

7. Rencana Perbaikan (Next steps)

- 3–5 poin yang realistis (mis. tambah dataset, tuning threshold, UI peringatan, audit log aman)

Di seluruh dokumen, bold hal penting seperti **FN/FP trade-off**, **batasan sistem**, **mitigasi**, **privacy handling**. Istilah asing tulis italic: *precision*, *recall*, *false positive*, *data minimization*, *dual-use*.

Checklist Penilaian (Rubrik Sederhana)

Kalau mau grading cepat, gunakan rubrik ini:

- **Evaluasi kuantitatif ada** (metrics jelas) — 30%
- **Robustness testing** (input nakal, tidak crash, ada analisis) — 20%
- **Privasi & pengamanan data** (masking/enkripsi dengan *GnuPG*) — 20%
- **Etika & mitigasi nyata** (bukan slogan) — 20%
- **Kerapian laporan & reproducibility** (requirements.txt, cara run) — 10%

Mini-Challenge (Supaya Menantang)

Pilih salah satu:

1. Turunkan **false negative** 20% tanpa menaikkan FP “gila-gilaan”.
2. Tambahkan “mode aman”: output **UNKNOWN** kalau confidence rendah (lebih etis daripada asal nuduh).
3. Semua file laporan/dataset uji terenkripsi *GnuPG* + ada SOP akses 5 baris.

Penutup

Buku *Keamanan Informasi & AI* ini menegaskan bahwa keamanan bukanlah sekadar kumpulan alat teknis, melainkan **cara berpikir sistematis dan berkelanjutan** dalam merancang, mengelola, dan mengevaluasi sistem digital. Melalui fondasi *CIA Triad*, manajemen risiko, kriptografi, keamanan jaringan, perlindungan data pribadi, hingga keamanan sistem berbasis AI, pembaca diajak memahami bahwa setiap sistem selalu memiliki risiko, dan tugas profesional keamanan adalah **mengelola risiko tersebut secara sadar dan bertanggung jawab**, bukan mengklaim sistem yang sepenuhnya aman .

Seiring berkembangnya pemanfaatan *artificial intelligence*, ruang lingkup keamanan informasi mengalami perluasan yang signifikan. AI tidak hanya menjadi alat bantu pertahanan—seperti dalam deteksi anomali, audit data, dan analitik keamanan—tetapi juga menjadi *attack surface* baru yang rentan terhadap *adversarial attack*, *data poisoning*, dan kebocoran model. Buku ini menunjukkan bahwa **AI dapat memperkuat keamanan, namun juga dapat memperbesar dampak kegagalan jika aspek integritas, privasi, dan tata kelola diabaikan**, sehingga keamanan AI harus diperlakukan sebagai bagian integral dari keamanan informasi secara keseluruhan .

Pendekatan pembelajaran yang digunakan menekankan keseimbangan antara teori dan praktik, sejalan dengan prinsip *Outcome-Based Education (OBE)*. Praktikum, studi kasus, dan proyek akhir dirancang agar pembaca—khususnya mahasiswa—tidak hanya memahami konsep, tetapi juga mampu **menerjemahkan regulasi, prinsip etika, dan kerangka keamanan ke dalam implementasi teknis yang nyata, terukur, dan dapat diaudit**. Dengan demikian, keamanan dipahami sebagai kompetensi profesional yang dapat diuji melalui bukti kerja, bukan sekadar pengetahuan konseptual .

Akhirnya, buku ini menekankan bahwa masa depan keamanan informasi dan keamanan AI sangat bergantung pada **mentalitas dan tanggung jawab manusia di balik teknologi**. Teknologi akan terus berubah, ancaman akan semakin adaptif, dan sistem akan semakin kompleks. Namun, dengan pola pikir kritis, kesadaran etika, dan komitmen terhadap praktik keamanan yang baik, para pembaca diharapkan mampu berkontribusi dalam membangun sistem digital yang **tidak hanya canggih, tetapi juga layak dipercaya, adil, dan berkelanjutan** bagi masyarakat luas .

LAMPIRAN: Membuat Data Training Set

Source Code

```
import os, json, random, string, zipfile, pandas as pd
from datetime import datetime, timedelta

base = "/mnt/data/ki_ai_praktikum_datasets"
os.makedirs(base, exist_ok=True)

random.seed(42)

def rand_ip(private_prob=0.7):
    if random.random() < private_prob:
        # private ranges
        choice = random.choice([10, 172, 192])
        if choice == 10:
            return
            f"10.{random.randint(0,255)}.{random.randint(0,255)}.{random.randint(1,254)}"
        if choice == 172:
            return
            f"172.{random.randint(16,31)}.{random.randint(0,255)}.{random.randint(1,254)}"
        return
            f"192.168.{random.randint(0,255)}.{random.randint(1,254)}"
    return
    f"{random.randint(1,223)}.{random.randint(0,255)}.{random.randint(0,255)}.{random.randint(1,254)}"

def rand_domain():
    tlds = ["com", "id", "net", "org"]
    names =
["kampus", "siakad", "market", "billing", "helpdesk", "cloud", "cdn", "api", "login", "portal"]
    return
    f"{random.choice(names)}.{random.choice(['example', 'demo', 'lab'])}.{random.choice(tlds)}"

def rand_email(name):
    return f"{name.lower().replace(' ', '_')}@{random.choice(['mail.demo', 'example.com', 'kampus.demo.id'])}"

def rand_name():
    first =
["Ayu", "Budi", "Citra", "Dimas", "Eka", "Fajar", "Gita", "Hadi", "Indah", "Joko", "Kiki", "Lia", "Maya", "Nanda", "Putra", "Rina", "Sari", "Tono", "Wulan", "Yusuf"]
```

```

last =
["Saputra", "Wijaya", "Pratama", "Siregar", "Halim", "Kusuma", "Setiawan", "P
ermata", "Nugroho", "Ramadhan"]
return f"{random.choice(first)} {random.choice(last)}"

def rand_nik():
    # 16 digits (synthetic)
    return "".join(str(random.randint(0,9)) for _ in range(16))

def rand_phone():
    return "08" + "".join(str(random.randint(0,9)) for _ in range(10))

def rand_token(n=24):
    return "".join(random.choice(string.ascii_letters+string.digits)
for _ in range(n))

# 1) Asset inventory + data classification
assets = []
systems = [
    ("SIKAD", "Web App", "10.10.2.15", "PostgreSQL", "PDP Tinggi",
    "Data akademik + identitas"),
    ("PAYROLL", "Web App", "10.10.2.20", "MySQL", "PDP Sangat Tinggi",
    "Gaji + NIK"),
    ("HELPDESK", "SaaS/On-Prem", "10.10.3.10", "MongoDB", "PDP
Sedang", "Tiket dukungan"),
    ("PORTAL UMKM", "Web App", "10.10.4.12", "MariaDB", "PDP Tinggi",
    "Customer & transaksi"),
    ("FILE SERVER", "Infra", "10.10.1.9", "N/A", "PDP Tinggi",
    "Dokumen internal"),
]
owners = ["BAK", "HRD", "IT Ops", "Unit UMKM", "Sekretariat"]
for i, (name, typ, ip, db, klass, desc) in enumerate(systems, start=1):
    assets.append({
        "asset_id": f"A-{i:03d}",
        "system_name": name,
        "asset_type": typ,
        "ip_address": ip,
        "data_store": db,
        "data_classification": klass,
        "owner_unit": owners[i-1],
        "description": desc
    })
pd.DataFrame(assets).to_csv(os.path.join(base, "01_asset_inventory.csv"
), index=False)

# 2) Network flow dataset (for anomaly detection)
start = datetime(2026, 1, 1, 8, 0, 0)
rows = []
protocols = ["TCP", "UDP"]
ports_common = [80, 443, 22, 53, 123, 3306, 5432, 8080, 8443]
for i in range(3500):
    ts = start + timedelta(seconds=random.randint(0, 8*3600))
    src = rand_ip()

```

```

dst = rand_ip(private_prob=0.9)
proto = random.choice(protocols)
dport = random.choice(ports_common)
sport = random.randint(1024,65535)
bytes_out = int(abs(random.gauss(2500,1200)))+100
packets = max(1,int(bytes_out/ random.randint(200,900)))
# Inject anomalies: bursts to single dst, unusual ports, huge
bytes
label = 0
if random.random() < 0.03:
    label = 1
    dport = random.choice([4444, 3389, 5900, 6667, 9001])
    bytes_out = random.randint(200000, 2000000)
    packets = random.randint(300, 5000)
rows.append({
    "timestamp": ts.isoformat(),
    "src_ip": src,
    "dst_ip": dst,
    "protocol": proto,
    "src_port": sport,
    "dst_port": dport,
    "bytes": bytes_out,
    "packets": packets,
    "label_anomaly": label
})
pd.DataFrame(rows).to_csv(os.path.join(base,"02_network_flows.csv"),
index=False)

# 3) Web access logs (CSV simplified)
paths=["/", "/login", "/dashboard", "/api/v1/orders", "/api/v1/profile", "/
admin", "/export", "/reset-password"]
agents=["Mozilla/5.0", "curl/8.0", "python-requests/2.31", "sqlmap/1.7", "
Go-http-client/1.1"]
status=[200,200,200,302,401,403,404,500]
w=[]
for i in range(1800):
    ts=start + timedelta(seconds=random.randint(0, 6*3600))
    ip=rand_ip()
    p=random.choice(paths)
    st=random.choice(status)
    ua=random.choice(agents)
    # inject suspicious: many 401 to /admin, sqlmap UA, export
attempts
suspicious=0
if random.random()<0.05:
    suspicious=1
    p=random.choice(["/admin", "/export", "/api/v1/profile"])
    st=random.choice([401,403,500])
    ua=random.choice(["sqlmap/1.7", "curl/8.0"])
w.append({
    "timestamp": ts.isoformat(),
    "client_ip": ip,
    "method": random.choice(["GET", "POST"]),

```

```

        "path": p,
        "status": st,
        "user_agent": ua,
        "suspicious": suspicious
    })
pd.DataFrame(w).to_csv(os.path.join(base, "03_web_access_logs.csv"),
index=False)

# 4) Auth logs (bruteforce + success)
users=["admin","operator","student","finance","hrd","support"]
a=[]
for i in range(1200):
    ts=start + timedelta(seconds=random.randint(0, 5*3600))
    u=random.choice(users)
    ip=rand_ip()
    outcome=random.choices(["FAIL","SUCCESS"], weights=[92,8])[0]
    reason = "invalid_password" if outcome=="FAIL" else "ok"
    suspicious=0
    if random.random()<0.06:
        # brute force block
        u=random.choice(["admin","finance","hrd"])
        outcome="FAIL"

reason=random.choice(["invalid_password","user_not_found","locked"])
suspicious=1
a.append({
    "timestamp": ts.isoformat(),
    "username": u,
    "source_ip": ip,
    "outcome": outcome,
    "reason": reason,
    "mfa_used": random.choice([0,1]),
    "suspicious": suspicious
})
pd.DataFrame(a).to_csv(os.path.join(base, "04_auth_logs.csv"),
index=False)

# 5) PII structured dataset (synthetic customer records)
cities=["Bekasi","Depok","Bogor","Tangerang","Jakarta"]
pii=[]
for i in range(500):
    name=rand_name()
    pii.append({
        "customer_id": f"C{i+1:05d}",
        "full_name": name,
        "email": rand_email(name),
        "phone": rand_phone(),
        "nik": rand_nik(),
        "city": random.choice(cities),
        "join_date":
            (datetime(2025,1,1)+timedelta(days=random.randint(0,360))).date().isoformat(),
        "balance_idr": random.randint(0, 25000000)
    })

```

```

    })
pd.DataFrame(pii).to_csv(os.path.join(base, "05_pii_customer_records.csv"), index=False)

# 6) Text dataset with mixed PII (JSONL tickets)
topics=["Reset password", "Pengaduan transaksi", "Perubahan alamat", "Refund", "Akun terkunci", "Lupa PIN"]
tickets_path=os.path.join(base, "06_helpdesk_tickets.jsonl")
with open(tickets_path, "w", encoding="utf-8") as f:
    for i in range(600):
        name=rand_name()
        email=rand_email(name)
        phone=rand_phone()
        nik=rand_nik()
        include_pii = random.random() < 0.55
        msg = f"Halo, saya {name}. Saya butuh bantuan untuk {random.choice(topics).lower()}."
        if include_pii:
            msg += f" Email: {email}. No HP: {phone}. NIK: {nik}."
            msg += f" OrderID: ORD{random.randint(100000, 999999)}. Terima kasih."
            obj={"ticket_id": f"T{i+1:06d}",
                "created_at":
                    (datetime(2025, 12, 1)+timedelta(minutes=random.randint(0, 60*24*20))).isoformat(),
                "channel": random.choice(["email", "whatsapp", "webform"]),
                "message": msg,
                "contains_pii": 1 if include_pii else 0}
            f.write(json.dumps(obj, ensure_ascii=False)+"\n")

# 7) Phishing email dataset
subjects=[
    "Verifikasi Akun Diperlukan",
    "Pembaruan Sistem Payroll",
    "Invoice Tertunda - Segera Bayar",
    "Reset Password SIAKAD",
    "Undangan Rapat Mendadak",
    "Paket Anda Tertahan"
]
phish=[]
for i in range(800):
    is_phish = 1 if random.random()<0.35 else 0
    sender =
        random.choice(["it-support@kampus.demo.id", "billing@market.demo.com", "noreply@portal.demo.net", "hr@kampus.demo.id"])
    if is_phish:
        sender =
            random.choice(["it-support@kampus-dem0.id", "billing@marke7.demo.com", "noreply@portaI.demo.net", "hr@kampus-security.demo.org"])
        link = f"https://{rand_domain()}/" +
            random.choice(["login", "verify", "invoice", "reset", "download"])
        if is_phish:

```

```

        link =
f"http://{random.choice(['secure','account','verify'])}.{random.choice
(['update','check','validate'])}.{random.choice(['ru','top','xyz'])}/{
rand_token(8)}"
        body = "Mohon lakukan tindakan berikut: " + ("verifikasi akun" if
random.random()<0.5 else "unduh dokumen") + f". Klik: {link}"
        if is_phish and random.random()<0.4:
            body += f" Kode OTP Anda: {random.randint(100000,999999)}"
        phish.append({
            "email_id": f"E{i+1:06d}",
            "timestamp":
(datetime(2025,11,15)+timedelta(minutes=random.randint(0,
60*24*30))).isoformat(),
            "from": sender,
            "subject": random.choice(subjects),
            "body": body,
            "label_phishing": is_phish
        })
pd.DataFrame(phish).to_csv(os.path.join(base,"07_phishing_emails.csv")
, index=False)

# 8) Incident scenario timeline (JSON)
incident = {
    "scenario_name": "Kebocoran Data Portal UMKM",
    "overview": "Terjadi akses tidak sah ke endpoint /export yang
menghasilkan ekstraksi data customer berisi email, telepon, dan NIK.",
    "timeline": [
        {"time":"2026-01-03T09:12:04","event":"Lonjakan request 401 ke
/admin dari IP eksternal","source":"03_web_access_logs.csv"},
        {"time":"2026-01-03T09:25:10","event":"Login SUCCESS untuk user
finance dari IP tidak biasa (kemungkinan kredensial
bocor)","source":"04_auth_logs.csv"},
        {"time":"2026-01-03T09:26:02","event":"Akses /export status 200
sebanyak 35 kali dalam 3 menit","source":"03_web_access_logs.csv"},
        {"time":"2026-01-03T10:05:44","event":"Outbound traffic besar ke
IP eksternal port 4444 (indikasi
exfil/command)","source":"02_network_flows.csv"},
        {"time":"2026-01-03T11:20:00","event":"Keluhan customer menerima
email mencurigakan","source":"07_phishing_emails.csv"}
    ],
    "student_tasks": [
        "Identifikasi indikator kompromi dari dataset logs.",
        "Tentukan data apa saja yang bocor dan klasifikasikan berdasarkan
PDP.",
        "Buat incident report dan rekomendasi mitigasi teknis +
prosedural.",
        "Buat aturan deteksi (rule) dan model ML sederhana untuk
mendeteksi pola serupa."
    ]
}
with
open(os.path.join(base,"08_incident_scenario_timeline.json"),"w",encod
ing="utf-8") as f:

```

```

    json.dump(incident,f,ensure_ascii=False,indent=2)

# 9) Poisoned training data (tabular) for AI security (data poisoning)
# Features: msg_len, num_links, has_urgent_words, sender_domain_score,
contains_attachment
poison=[]
urgent_words=["segera","urgent","darurat","hari ini","akun
diblokir","verifikasi"]
for i in range(1500):
    label = 1 if random.random()<0.3 else 0 # phishing
    msg_len = int(abs(random.gauss(220,90)))+20
    num_links = random.choices([0,1,2,3], weights=[55,30,10,5])[0] if
label==0 else random.choices([0,1,2,3], weights=[10,40,30,20])[0]
    urgent = 1 if (random.random() < (0.15 if label==0 else 0.6)) else
0
    sender_score = round(random.uniform(0.4,1.0),3) if label==0 else
round(random.uniform(0.0,0.6),3)
    attach = 1 if random.random() < (0.08 if label==0 else 0.25) else
0
    poisoned = 0
    # Inject poisoning: flip labels for some phishing-like samples
    if random.random() < 0.04:
        poisoned = 1
        label = 0
        urgent = 1
        num_links = max(1, num_links)
        sender_score = round(random.uniform(0.0,0.4),3)
    poison.append({
        "msg_len": msg_len,
        "num_links": num_links,
        "has_urgent_words": urgent,
        "sender_domain_score": sender_score,
        "contains_attachment": attach,
        "label_phishing": label,
        "poisoned_sample": poisoned
    })
pd.DataFrame(poison).to_csv(os.path.join(base,"09_poisoned_training_da
ta.csv"), index=False)

# 10) Differential privacy demo dataset (aggregate friendly)
dp=[]
for i in range(1200):
    dp.append({
        "user_id": f"U{i+1:05d}",
        "age": random.randint(17,55),
        "city": random.choice(cities),
        "monthly_spend_idr": random.randint(50000, 15000000),
        "category":
random.choice(["makanan","transport","pendidikan","kesehatan","hiburan
","lainnya"])
    })
pd.DataFrame(dp).to_csv(os.path.join(base,"10_dp_demo_spend.csv"),
index=False)

```

```
# README with scenarios + mapping to pertemuan
readme = """\
KI & AI Praktikum Datasets (Synthetic) - 2026
=====
Semua data di folder ini adalah DATA SINTETIS (bukan data nyata),
dibuat untuk latihan praktikum.

Daftar File
-----
01_asset_inventory.csv
  - Inventaris sistem/aset + klasifikasi data (untuk Pertemuan 1,2,6)

02_network_flows.csv
  - Netflow sederhana (bytes, packets, dst_port) + label anomaly
  (untuk Pertemuan 4,5,10,7)

03_web_access_logs.csv
  - Log akses web (path, status, user-agent) + flag suspicious (untuk
  Pertemuan 4,5,7)

04_auth_logs.csv
  - Log autentikasi (FAIL/SUCCESS) + flag suspicious (untuk Pertemuan
  5,7,10)

05_pii_customer_records.csv
  - Rekaman customer (PII: nama, email, telepon, NIK) untuk latihan
  perlindungan data (Pertemuan 6,7,11)

06_helpdesk_tickets.jsonl
  - Tiket helpdesk dalam teks; sebagian berisi PII (Pertemuan 11)

07_phishing_emails.csv
  - Dataset email phishing berlabel (Pertemuan 10,13)

08_incident_scenario_timeline.json
  - Skenario insiden + timeline dan tugas mahasiswa (Pertemuan 7,8,13)

09_poisoned_training_data.csv
  - Dataset tabular dengan sampel poisoning (Pertemuan 9,13)

10_dp_demo_spend.csv
  - Dataset demo untuk differential privacy (Pertemuan 12)

Skenario Latihan (Ringkas)
-----
S1: "Kebocoran Data Portal UMKM" (08_incident_scenario_timeline.json)
  - Gunakan 03,04 untuk deteksi akses tidak sah
  - Gunakan 02 untuk indikasi exfiltration
  - Gunakan 05 untuk memahami jenis data yang bocor dan klasifikasi PDP
  - Hasil: Incident report, rekomendasi mitigasi, aturan deteksi + model
  ML
```

S2: "Deteksi Phishing Berbasis ML"

- Latih model pada 07_phishing_emails.csv (atau ekstrak fitur sederhana)
- Evaluasi precision/recall dan jelaskan false positive/negative
- Hasil: notebook + confusion matrix + rekomendasi implementasi

S3: "Audit PII pada Tiket Helpdesk"

- Scan 06_helpdesk_tickets.jsonl untuk deteksi email/telepon/NIK (regex dan/atau ML)
- Hitung persentase tiket yang mengandung PII dan usulkan kontrol (masking, retention, access control)
- Hasil: laporan audit + contoh masking

S4: "Eksperimen Data Poisoning"

- Latih model sederhana dengan 09_poisoned_training_data.csv
- Deteksi outlier/poisoned_sample dan uji robust training
- Hasil: analisis dampak poisoning

Catatan Etika

- Tidak boleh mengunggah data nyata ke publik.
 - Praktikum OSINT dilakukan pada data simulasi/dummy saja.
- """

```
with open(os.path.join(base, "README.txt"), "w", encoding="utf-8") as f:
    f.write(readme)
```

```
# Zip everything
```

```
zip_path="/mnt/data/ki_ai_praktikum_datasets.zip"
```

```
with zipfile.ZipFile(zip_path, "w", zipfile.ZIP_DEFLATED) as z:
```

```
    for root, _, files in os.walk(base):
```

```
        for fn in files:
```

```
            fp=os.path.join(root, fn)
```

```
            arc=os.path.relpath(fp, base)
```

```
            z.write(fp, arcname=f"ki_ai_praktikum_datasets/{arc}")
```

```
zip_path, sorted(os.listdir(base))[:5], len(os.listdir(base))
```

Berikut **paket dataset sintetis + skenario latihan** yang langsung nyambung ke modul praktikum per pertemuan (1–14). Semua data **bukan data nyata** (aman untuk lab).

Download paket dataset (ZIP):

`ki_ai_praktikum_datasets.zip`

Isi Dataset (di dalam ZIP)

1. **01_asset_inventory.csv**
Inventaris sistem/aset + klasifikasi data (buat Pertemuan 1,2,6)
2. **02_network_flows.csv**
Netflow sederhana (bytes/packets/port) + label anomali (buat 4,5,7,10)
3. **03_web_access_logs.csv**
Log akses web (path/status/user-agent) + flag suspicious (buat 4,5,7)
4. **04_auth_logs.csv**
Log autentikasi (FAIL/SUCCESS, MFA) + flag suspicious (buat 5,7,10)
5. **05_pii_customer_records.csv**
Rekaman customer berisi PII sintetis (nama/email/telepon/NIK) (buat 6,7,11)
6. **06_helpdesk_tickets.jsonl**
Tiket helpdesk teks (sebagian mengandung PII) (buat 11)
7. **07_phishing_emails.csv**
Dataset email phishing berlabel (buat 10,13)
8. **08_incident_scenario_timeline.json**
Skenario insiden + timeline & tugas (buat 7,8,13)
9. **09_poisoned_training_data.csv**
Dataset tabular untuk eksperimen **data poisoning** (buat 9,13)
10. **10_dp_demo_spend.csv**
Dataset demo untuk **differential privacy** (buat 12)
11. **README.txt**
Panduan mapping dataset → pertemuan + ringkasan skenario

Skenario Latihan (Lengkap & Siap Dipakai)

Skenario A — “Kebocoran Data Portal UMKM”

Tujuan: audit kebocoran data pribadi + incident response + deteksi berbasis AI

File dipakai: 02, 03, 04, 05, 08

Narasi

Terdeteksi aktivitas mencurigakan pada aplikasi “Portal UMKM”: banyak akses ke endpoint **/export** dan outbound traffic besar. Setelah itu muncul keluhan pengguna menerima email aneh.

Tugas Mahasiswa (Output Praktikum)

1. **Cari indikator kompromi (IOC)** dari:
 - akses web (03): lonjakan 401/403, user-agent “sqlmap/curl”, akses ke `/export`
 - auth log (04): login sukses dari IP tidak biasa / pola brute-force
 - netflow (02): outbound besar ke port tidak umum (indikasi exfil)
2. **Tentukan data apa yang kemungkinan bocor**
 - cocokkan endpoint “export” dengan data (05_pii_customer_records.csv)
 - klasifikasikan berdasarkan PDP (umum vs sensitif)
3. **Buat Incident Report** (template kampus)
 - ringkasan insiden
 - timeline
 - dampak (CIA + PDP)
 - akar masalah (hipotesis)
 - rekomendasi teknis + prosedural
4. **Buat deteksi lanjutan**
 - aturan (rule) sederhana: threshold request `/export` / burst logins
 - model ML sederhana: anomaly detection dari 02 + 03

Kriteria sukses: mahasiswa bisa membuktikan “kenapa ini breach”, bukan sekadar dugaan.

Skenario B — “Deteksi Phishing Berbasis ML”

Tujuan: AI untuk keamanan (klasifikasi phishing)

File dipakai: 07 (opsional 04 untuk korelasi akun)

Tugas

1. Feature extraction sederhana dari `subject + body`:
 - jumlah link
 - ada kata mendesak (“segera/urgent/verifikasi”)
 - domain mencurigakan (TLD aneh, http bukan https, typo)
2. Train model baseline:
 - Logistic Regression / Naive Bayes
 - Laporkan precision/recall/F1
3. Analisis **false positive** dan **false negative**:
 - email yang aman tapi terdeteksi phish
 - phish yang lolos
4. Rekomendasi deployment:
 - di gateway email? di SOC? di helpdesk?

Kriteria sukses: ada evaluasi metrik + pembahasan bias/error.

Skenario C — “Audit PII di Tiket Helpdesk”

Tujuan: audit kebocoran data pribadi + AI/regex untuk deteksi PII

File dipakai: 06_helpdesk_tickets.jsonl

Tugas

1. Buat detektor PII:
 - regex email
 - regex nomor HP Indonesia
 - regex NIK 16 digit
 - (opsional) ML/NLP sederhana untuk konteks
2. Hasil audit:
 - persentase tiket mengandung PII
 - 10 contoh temuan (tanpa menampilkan penuh—masking)
3. Rekomendasi kontrol:
 - masking otomatis sebelum masuk database
 - role-based access untuk staff
 - retention policy (berapa lama disimpan)
 - logging akses tiket

Kriteria sukses: deteksi berjalan + ada usulan kontrol teknis.

Skenario D — “Eksperimen Data Poisoning”

Tujuan: keamanan AI (AI sebagai target)

File dipakai: 09_poisoned_training_data.csv

Tugas

1. Train model klasifikasi phishing dari fitur tabular
2. Bandingkan performa:
 - data “apa adanya”
 - data setelah cleansing sederhana (outlier/poison filtering)
3. Deteksi poisoning:
 - cari sampel aneh (poisoned_sample)
 - analisis dampak terhadap decision boundary
4. Rekomendasi:
 - data validation pipeline
 - provenance data
 - robust training / monitoring drift

Kriteria sukses: mahasiswa melihat “poisoning bikin model meleset”.

Skenario E — “Differential Privacy untuk Laporan Statistik”

Tujuan: privacy-preserving analytics

File dipakai: 10_dp_demo_spend.csv

Tugas

1. Buat laporan agregat (rata-rata spend per kota/kategori)
2. Tambahkan noise (DP sederhana):
 - bandingkan utility (selisih hasil)
 - diskusikan trade-off privacy vs accuracy
3. Buat kebijakan:
 - kapan DP wajib dipakai (laporan publik, dataset riset, dsb.)

Kriteria sukses: ada angka sebelum-sesudah + interpretasi.

Mapping Cepat: Dataset → Pertemuan

- **P1–P2:** 01 (inventaris & risiko)
- **P4–P5:** 02, 03, 04 (traffic/log/IDS)
- **P6:** 01, 05 (klasifikasi PDP)
- **P7–P8:** 02, 03, 04, 05, 08 (audit breach + integrasi)
- **P9:** 09 (AI poisoning)
- **P10:** 02/07 (AI detection)
- **P11:** 06/05 (AI audit PII)
- **P12:** 10 (DP)
- **P13:** gabungan sesuai proyek
- **P14:** review + etika

Glossary

A

Adversarial Attack

Serangan terhadap sistem AI dengan cara memanipulasi input secara halus sehingga model menghasilkan keputusan yang salah tanpa terlihat rusak secara teknis.

Adversarial Example

Contoh input yang telah dimodifikasi secara sengaja untuk menipu model AI agar salah mengklasifikasikan atau mengambil keputusan keliru.

AI Security

Bidang keamanan yang berfokus pada perlindungan sistem kecerdasan buatan, termasuk data, model, pipeline training, dan output AI.

Anomaly Detection

Teknik pendeteksian pola tidak normal dalam data atau aktivitas sistem yang dapat mengindikasikan serangan atau kegagalan.

Asset (Aset)

Segala sesuatu yang bernilai dan perlu dilindungi, seperti data, sistem, layanan, atau pengguna.

Attack Surface

Seluruh titik atau komponen sistem yang dapat diakses dan berpotensi dieksploitasi oleh penyerang.

C

CIA Triad

Kerangka dasar keamanan informasi yang terdiri dari Confidentiality, Integrity, dan Availability.

Confidentiality (Kerahasiaan)

Prinsip yang menjamin bahwa informasi hanya dapat diakses oleh pihak yang berwenang.

Consent

Persetujuan eksplisit dari subjek data terkait pengumpulan, pemrosesan, dan penggunaan data pribadinya.

Cryptography (Kriptografi)

Ilmu dan teknik untuk melindungi informasi melalui enkripsi, hashing, dan mekanisme kunci.

D

Data Breach

Insiden kebocoran data akibat kegagalan keamanan, baik karena serangan maupun kelalaian.

Data Lifecycle

Siklus hidup data mulai dari pengumpulan, pemrosesan, penyimpanan, penggunaan, hingga penghapusan.

Data Poisoning

Serangan terhadap AI dengan cara merusak atau memanipulasi data pelatihan agar model menghasilkan output yang bias atau berbahaya.

Denial of Service (DoS / DDoS)

Serangan yang bertujuan melumpuhkan layanan sehingga tidak dapat diakses oleh pengguna yang sah.

Differential Privacy

Pendekatan privasi yang melindungi data individu dengan menambahkan *noise* terkontrol pada hasil analisis.

E

Elevation of Privilege

Ancaman ketika pengguna biasa berhasil memperoleh hak akses yang lebih tinggi secara tidak sah.

Encryption (Enkripsi)

Proses mengubah data menjadi bentuk tidak terbaca tanpa kunci yang sesuai.

F

Federated Learning

Pendekatan pembelajaran AI terdistribusi dimana data tetap berada di sisi pengguna dan tidak dikumpulkan secara terpusat.

Failover

Mekanisme pemindahan layanan secara otomatis ke sistem cadangan saat sistem utama gagal.

G

GnuPG (GNU Privacy Guard)

Perangkat lunak open-source untuk enkripsi, dekripsi, dan tanda tangan digital berbasis kriptografi asimetris.

H

Hashing

Proses satu arah untuk mengubah data menjadi nilai hash tetap, umumnya digunakan untuk penyimpanan password.

I

Incident Response

Serangkaian langkah sistematis untuk menangani, mengendalikan, dan memulihkan insiden keamanan.

Integrity (Integritas)

Prinsip yang menjamin data tetap utuh, akurat, dan tidak berubah tanpa izin.

Intrusion Detection System (IDS)

Sistem yang memantau aktivitas jaringan atau sistem untuk mendeteksi potensi serangan.

K

Key Management

Proses pengelolaan siklus hidup kunci kriptografi, termasuk pembuatan, penyimpanan, distribusi, dan pencabutan.

L

Log Monitoring

Pemantauan log sistem untuk mendeteksi aktivitas mencurigakan atau kegagalan.

M

Model Leakage

Kondisi ketika informasi sensitif tentang data pelatihan atau model AI dapat diekstraksi oleh pihak luar.

Machine Learning (ML)

Pendekatan AI di mana sistem belajar dari data untuk membuat prediksi atau keputusan.

P

PII (Personally Identifiable Information)

Informasi yang dapat digunakan untuk mengidentifikasi individu secara langsung maupun tidak langsung.

Privacy-Preserving AI

Pendekatan AI yang dirancang untuk meminimalkan risiko pelanggaran privasi.

R

Risk Assessment

Proses identifikasi, analisis, dan evaluasi risiko keamanan berdasarkan kemungkinan dan dampaknya.

Risk Scoring

Pemberian nilai kuantitatif pada risiko untuk membantu penentuan prioritas mitigasi.

S

Security Analytics

Analisis data keamanan untuk menemukan pola, tren, dan indikator ancaman.

Signature-Based Detection

Metode deteksi serangan berdasarkan pola atau tanda tangan yang telah dikenal.

STRIDE

Kerangka threat modeling yang mengklasifikasikan ancaman menjadi Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, dan Elevation of Privilege.

T

Threat

Potensi kejadian atau tindakan yang dapat merusak aset.

Threat Modeling

Pendekatan sistematis untuk memahami bagaimana sebuah sistem dapat diserang atau gagal.

V

Availability (Ketersediaan)

Prinsip yang menjamin sistem dan data dapat diakses saat dibutuhkan oleh pengguna yang sah.

Vulnerability (Kerentanan)

Kelemahan dalam sistem yang dapat dieksploitasi oleh ancaman.

.

Referensi

- Bierbrauer, D. A. (2021). *Cybersecurity anomaly detection in adversarial environments*. arXiv. <https://arxiv.org/abs/2105.06742>[arxiv]
- Chang, J. M., & Di, X. (2025). *Privacy-preserving machine learning*. Manning.[manning]
- Pemerintah Indonesia. (2022). *Undang-Undang Republik Indonesia Nomor 27 Tahun 2022 tentang Perlindungan Data Pribadi*. Lembaran Negara Republik Indonesia.[research.e-greenation]
- Stallings, W. (2024). *Cryptography and network security: Principles and practice* (8th global ed.). Pearson.[books.google.co]
- Stamp, M. (2019). *Information security: Principles and practice* (3rd ed.). Wiley.[wiley]
- Beerliová-Trubíniová, Z., & Hubáček, P. (Eds.). (2024). *Adversarial learning and secure AI*. Cambridge University Press.[assets.cambridge]
- Lutrianto, I. (2025). Legal problems of personal data protection in the digital era. *Global Indonesian Journal of Law and Social Sciences*, 3(2), 1–15.[research.e-greenation]
- Mirna, M. (2023). Analisis pengaturan keamanan data pribadi di Indonesia. *Living Law: Jurnal Ilmiah Fakultas Hukum Universitas Sultan Ageng Tirtayasa*, 15(1), 45–60.[ojs.unida.ac]

Tentang Penulis



Onno W. Purbo, saat ini bertugas sebagai rektor di Institut Teknologi Tangerang Selatan (ITTS). Onno memperoleh gelar Ph.D bidang Electrical Engineering dari University of Waterloo, Canada, adalah seorang copyleft, educator dan ICT evangelist. Dia sudah mempublikasikan 50++ buku, termasuk free ICT ebook untuk sekolah tahun 2008. Beberapa buku terakhirnya adalah "Internet-TCP/IP: Konsep Dan Implementasi", 2018; "Sistem Operasi, Konsep Dan Membuat Linux OpenWRT Dan ROM Android", 2019; "IPv6 Untuk Mendukung Operasi Jaringan Dan Domain Name System", 2019; "Kubernetes untuk Pemula", 2024 dan "Membuat Operator Seluler 5G Sendiri", 2024. Dia memimpin sambungan pertama Internet di Institut Teknologi Bandung, tahun 1993-2000, dan menggunakannya untuk membuat jaringan Internet pendidikan yang pertama di

Indonesia. Dia membebaskan frekuensi WiFi, memperkenalkan RT/RW-net, antenna Wajanbolic dan jaringan selular OpenBTS dan private 5G sendiri. Dia memimpin jaringan telepon pertama di atas Internet, VoIP Merdeka, yang kemudian hari dikenal sebagai VoIP Rakyat berbasis SIP dan menggunakan kode area +62520 dan +62521. Dia saat ini aktif memperkenalkan e-Learning, dan menjalankan server e-Learning di <http://lms.onnocenter.or.id/moodle/> 66,000++ siswa / mahasiswa dan <https://opencourse.itts.ac.id> 39.000+ mahasiswa secara gratis. Email: onno@indo.net.id