Lab Exercise: Why Regex Still Matters in the Age of AI (Regex vs GPT Log Forensics)

Learning Outcomes

By the end of this mini-project, students should be able to:

1. Apply regular expressions (Regex) to extract forensic artefacts (e.g., IP addresses, URLs, emails, credit card numbers, phone numbers, USB events, and login results) from large log files in a reproducible way.
2. Use a large language model (e.g., ChatGPT) to perform the same extraction task and reflect on its strengths and weaknesses (e.g., inconsistency, probabilistic output).
3. Compare the two approaches using **precision, recall, F1-score, runtime, and explainability**.
4. Develop an understanding of **forensic soundness**: reproducibility, transparency, and evidence verification.

Prerequisites

- Basic command-line skills (Bash or PowerShell). Familiarity with `grep` / `ripgrep`.
- Basic Regex syntax (groups, quantifiers, character classes, word boundaries, escapes).
- (Optional) Python basics (`re` library) or Autopsy's Keyword/Regex Search module.

Resources

- Dataset: `regex_vs_gpt_lab_dataset.zip` (synthetic logs with ground truth). Includes:
    - `logs/web.log` — Apache-style log with URLs, IPs, emails, credit cards (and near-miss "fake" numbers).
    - `logs/auth.log` — SSH authentication events with successes/failures and IPs.
    - `logs/app.log` — Application events with emails, phone numbers, USB insertions.
    - `ground_truth.csv` — Gold standard annotations (type, value, file, line).
- Tools: `ripgrep (rg)` or `grep`; text editor; optionally Python, Autopsy, or Excel/LibreOffice.
- A GPT interface (e.g., ChatGPT web or API).

Workflow

Part A: Regex Pipeline (Deterministic)

1. **Download and extract dataset**:

```
unzip regex_vs_gpt_lab_dataset.zip -d lab
cd lab/regex_vs_gpt_lab
```

2. **Write Regex patterns** (individually, then refine in groups):
    - IPv4: `\b(?:25[0-5]|2[0-4]\d|1?\d?\d)(?:\.(?:25[0-5]|2[0-4]\d|1?\d?\d)){3}\b`
    - URL: `https?://[^\s"]+`
    - Email: `[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}`
    - Phone: `\+?\d[\d\-\s()]{6,}\d`
    - Credit card: `\b(?:\d[ -]*?){13,16}\b`
    - USB event: `usb\s+\d+-\d+:\s+new\s+high-speed\s+USB\s+device\b`
    - SSH logins: Accepted / Failed password patterns.
3. **Run searches and export matches** using `ripgrep` with `-n` for line numbers.
4. **Optional**: Apply Luhn algorithm in Python to filter out invalid credit card numbers.
5. **Consolidate results** into CSV/TSV with columns `value, file:line`.

Part B: GPT Pipeline (Probabilistic)

1. Provide log chunks to GPT with a structured prompt (JSON output format with type, value, file, line).
2. Save outputs to CSV/JSON.
3. Record **time taken, prompts used, and reproducibility issues**.

Part C: Evaluation

1. Compare results with `ground_truth.csv`.
2. Metrics:
    - Precision = TP / (TP + FP)
    - Recall = TP / (TP + FN)
    - F1-score = 2PR / (P + R)
    - Runtime (human effort + tool execution)
    - Reproducibility (consistent across attempts?)
    - Explainability (can you justify why it matched?)
3. Key observations:
    - Regex: may over-match (false positives), requires refinement.
    - GPT: may miss matches or provide inconsistent line numbers.
    - Filtering (e.g., Luhn) improves accuracy of Regex pipeline.

Assessment Rubric

- **Implementation (40%)**: Functional Regex pipeline; optional credit card filtering.
- **Evaluation (30%)**: Correct metrics against ground truth, with error analysis.
- **Documentation & Reproducibility (20%)**: Clear record of commands, Regex rules, scripts, and timings.
- **Discussion (10%)**: Balanced analysis of Regex vs GPT, supported by experimental evidence.

Teaching Notes

- Encourage students to try Regex independently before group refinement.
- Include “distractor” samples (invalid credit cards) to highlight the need for two-stage detection.
- In GPT stage, have students log different prompts and note changes in results to emphasise variability.

Extensions

- **Autopsy demo**: Import logs as evidence, use Keyword/Regex Search and Timeline tools.
- **Memory/network analysis**: Extend to small `pcap` or RAM dumps with regex/AI extraction.
- **Legal/ethical**: Ask students to write a short expert-witness style statement, reflecting on reproducibility and admissibility.

Quick Reference Regex

- Email: `[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}`
- URL: `https?://[^\s"]+`
- IPv4: `\b(?:25[0-5]|2[0-4]\d|1?\d?\d)(?:\.(?:25[0-5]|2[0-4]\d|1?\d?\d)){3}\b`
- Credit card: `\b(?:\d[ -]*?){13,16}\b` (apply Luhn check)
- Phone: `\+?\d[\d\-\s()]{6,}\d`
- USB insert: `usb\s+\d+-\d+:\s+new\s+high-speed\s+USB\s+device\b`
- SSH success: `Accepted password for\s+(\w+)\s+from\s+(\d+\.\d+\.\d+\.\d+)`
- SSH failure: `Failed password for\s+(\w+)\s+from\s+(\d+\.\d+\.\d+\.\d+)`

Suggested Deliverables

- Optional Group report (4–6 pages) including methodology, Regex rules, GPT prompts, evaluation metrics, and discussion.
- Short presentation (10 minutes) demonstrating findings and reflections.

> Note: Dataset is synthetic, designed for teaching. Contains distractor samples to test robustness of Regex and GPT approaches, highlighting differences in reproducibility and interpretability.