**AI-BASED MONITORING AND DETECTION OF PHISHING DOMAINS/URLS RELATED TO CRITICAL SECTOR ENTITIES (CSEs)**

**1. Problem Context**

* **Phishing threats:** The main problem is the rise of sophisticated phishing attacks targeting CSEs (such as banking, energy, healthcare, etc.), which now use techniques like tunnelling services, lookalike / typo squatting domains, parked domains, and content cloned from official sites.
* **Current gaps:** Traditional detection methods fall short against new strategies like non-resemblance of URLs, empty parked domains, and internationalized domain names (IDNs).

**2. Objectives & Output**

* **End Goal:** Develop a scalable, automated AIML-powered engine to continuously crawl, parse, and detect phishing/suspected domains in near real-time, providing timely alerts to prevent breaches and losses.
* **Key Features:**
  + Friendly browser-based dashboard.
  + AIML engine to monitor massive datasets (structured + unstructured).
  + Real-time detection/reporting of phishing and suspected domains.
  + Continuous monitoring for at least 3 months (configurable).
  + Output mapping between CSEs and relevant phishing/suspected domains/URLs.

**3. Detection Artifacts & Attributes**

* **Artifacts to track:** URL features (length, number of dots/special chars, entropy), domain features, subdomain features, path features, favicon/image-based features, registration data, SSL/HTTPS, screenshot comparison, visual similarity, IDN handling, etc.
* **Output report attributes:** Creation date/time, IP/subnet, maliciousness score, registrar/registrant info, ASN, geolocation, MX records, cert transparency, screenshots, external verification—all to be mapped and reported.

**4. Process Flow & Evaluation**

* **Training and evaluation datasets:** Multiple curated datasets (training, mock, shortlist, hold-out) are provided and updated in stages for participant solution development, testing, and final evaluation.
* **Stages:**
  + **Stage 1:** Self-evaluation using mock/shortlist sets. Submission includes detailed logs, reports, and zipped platforms.
  + **Stage 2:** Live demonstration & evaluation on FIIT, IIT Delhi infrastructure, scored via predefined metrics.
  + **Finalists:** Top performers advance and showcase for final selection. Solution demo duration: 5 days (subject to adjustment).

**5. Rules/Requirements**

* **Solution must:**
  + Be dockerized and run on specified Ubuntu hardware (48 cores, 256GB RAM, 500GB storage).
  + Be on-premises, internet-connected.
  + Avoid third-party/proprietary threat intelligence APIs; all calls must be declared and verified.
  + Include comprehensive documentation and implementation details.
  + Be flexible on language/framework (open-source preferred).
* **Participants may:**
  + Use their own data/methods for solution development.
  + Innovate beyond listed artifacts/attributes, but not use unfair means.

**6. Artifacts - Feature List**

* **Annexure A:** Illustrative list of features for domain and URL analysis (lengths, special chars, subdomain stats, favicon hash/color/image similarity, image-based text, registration data, SSL, entropy, etc.). Encouraged to add innovative features/methods.

**7. Timeline & QA**

* QA sessions begin August 15, 2025. Regular site checks for updates are required.
* Datasets released in batches for mock, shortlist, and hold-out stages.
* Platform submission due October 31, 2025.

**8. Key Constraints**

* **No commercial threat intelligence APIs allowed.**
* **Strict monitoring for unfair practices; non-compliance leads to disqualification.**

**Conclusion:**  
This scope demands the design of an efficient, scalable, and deployable AIML-based system specifically tailored to identify rapidly-evolving phishing and suspicious domains targeting CSEs. The system must demonstrate real-time detection, automated reporting, comprehensive logging, feature-rich artifact tracking, and robust evaluation—while adhering to transparency, open-source standards, and hard security constraints. Continuous improvement, innovation, and comprehensive documentation are highly valued throughout the development and evaluation journey.ShortSummaryOnProblemStatement2Doc.pdf

**Recommended AI/ML Model & Architecture**

**1. Modalities Involved**

* **Lexical / Textual (URL, domain, path, subdomains)**
* **Metadata (WHOIS, SSL, MX records, ASN, registrar info, etc.)**
* **Image (screenshots, favicons, logos, visual similarity to CSEs)**
* **Behavioral (domain age, parking detection, DNS activity, CT logs, entropy)**

**2. Model Choices per Modality**

**🔹 A. URL / Domain Text**

* **Model**: Pretrained Transformer (e.g., BERT, DistilBERT, URLNet, or DNABERT-style models adapted for domain tokens).
* **Why**: Captures semantic + character-level patterns (typosquatting, homoglyphs, IDNs).
* **Input**: Tokenized URL/domain sequences.
* **Output**: URL risk embedding (vector).

**🔹 B. Metadata (Tabular)**

* **Model**: Gradient Boosted Trees (XGBoost/LightGBM) or Random Forest.
* **Why**: Tabular ML models handle structured categorical/numeric metadata better than deep nets.
* **Features**: Registrar country, ASN, SSL validity, domain age, DNS records, entropy, number of dots/hyphens, etc.
* **Output**: Metadata risk score.

**🔹 C. Image (Screenshot / Favicon / Logo)**

* **Model**: CNN (ResNet-50 or EfficientNet) + Perceptual Hash (pHash/dHash).
* **Why**: CNN extracts **visual layout similarity**, while hash detects near-duplicates of known CSE logos/pages.
* **Input**: Rendered webpage screenshot, favicon.
* **Output**: Visual similarity embedding.

**🔹 D. Behavioral / Temporal**

* **Model**: LSTM/GRU or Temporal GNN.
* **Why**: Can capture temporal signals like sudden DNS changes, parking → content activation, CT log patterns.
* **Input**: Domain time-series (age, DNS/IP shifts, cert issuance).
* **Output**: Temporal risk embedding.

**3. Fusion Architecture (Multi-Modal Ensemble)**

Recommended design: **Late Fusion + Meta-Ensemble**

* Each modality outputs a **risk embedding or score**.
* A **meta-classifier** (e.g., Logistic Regression or shallow MLP) combines these into a **final phishing/suspected/benign decision**.

**Architecture Flow:**

1. **Input** (Domain/URL, Metadata, Screenshot, Logs).
2. **Parallel Processing**:
   * Transformer → URL embedding.
   * XGBoost → Metadata score.
   * CNN + pHash → Image similarity score.
   * LSTM → Temporal risk score.
3. **Fusion Layer**:
   * Concatenate embeddings + scores.
   * Pass through **dense layers** with dropout + batchnorm.
4. **Output**:
   * Final classification: {Phishing, Suspected, Legit}.
   * Confidence score.

**4. Deployment Architecture**

* **Real-time pipeline**:
  + Input → Feature Extractors → Modal Models → Fusion Layer → Dashboard.
* **Batch pipeline**:
  + Periodic re-evaluation of parked domains (every 24h).
* **Explainability**:
  + Use SHAP for tabular features, attention visualization for URL text, Grad-CAM for screenshots.

**5. Why This Hybrid Approach Works**

* **Transformers** excel in text/URL similarity and typosquatting detection.
* **XGBoost** is best for tabular metadata (WHOIS/SSL).
* **CNNs** detect visual cloning of CSE websites.
* **LSTM/GNN** captures evolving parked → active phishing behavior.
* The **fusion meta-classifier** ensures robustness by combining strengths across modalities.

**Final Recommendation:**  
Use a **multi-modal late fusion ensemble** with:

* **Transformer (BERT/URLNet) for URLs**
* **XGBoost for metadata**
* **CNN (ResNet-50) for screenshots/favicons**
* **LSTM/GNN for temporal features**
* **Meta-MLP or Logistic Regression fusion layer**

This ensures **high accuracy, interpretability, and scalability** on the competition datasets.