Phishing Detector Browser Extension

1. Project Overview

Phishing attacks remain a major cybersecurity threat where attackers impersonate trusted websites to steal user credentials or sensitive data. This project presents a browser extension designed to detect potentially malicious websites in real-time by analyzing URLs using VirusTotal and WhoisXML APIs, combined with heuristic checks.

2. Objective

Create a lightweight browser extension that:

- Scans the active tab URL for phishing indicators.
- Queries VirusTotal to check URL reputation.
- Fetches domain age using WhoisXML API to spot newly created suspicious domains.
- Provides a user-friendly popup warning about suspicious websites.

3. Tools & Technologies Used

- **JavaScript** (ES6)
- **Chrome Extension APIs**
- **VirusTotal API** – URL scanning and reputation
- **WhoisXML API** – Domain registration data
- **HTML/CSS** – Popup UI
- **Fetch API** – For asynchronous network calls

4. File Structure

```
/phishing-detector-extension/
├── manifest.json
├── background.js
├── popup.html
└── popup.js
```

5. Workflow

1. User navigates to a website.
2. Popup retrieves current tab URL.
3. Performs heuristic checks on URL keywords and protocol (HTTP/HTTPS).
4. Sends message to background script to:
   - Submit URL to VirusTotal API and retrieve scan results.
   - Fetch domain creation date from WhoisXML API.
5. Background script sends back risk indicators.

6. Popup updates UI with risk status and domain info.
7. Visual status bar changes color based on risk level (green/yellow/red).

6. Code Breakdown

1. manifest.json

Defines extension metadata, permissions, and files loaded.

```json
{
  "manifest_version": 3,
  "name": "Phishing Detector",
  "version": "1.0",
  "description": "Warns about suspicious websites",
  "permissions": ["tabs", "activeTab", "scripting"],
  "host_permissions": [
    "https://www.virustotal.com/*",
    "https://www.whoisxmlapi.com/*"
  ],
  "background": {
    "service_worker": "background.js"
  },
  "action": {
    "default_popup": "popup.html"
  }
}
```

2. background.js

Handles asynchronous API calls to VirusTotal and WhoisXML, processes results, and sends back risk data.

js

```
const API_KEY = "<YOUR_VIRUSTOTAL_API_KEY>";  // VirusTotal API key placeholder
const WHOIS_API_KEY = "<YOUR_WHOISXML_API_KEY>"; // WhoisXML API key
placeholder

// Extract domain from full URL
function extractDomain(url) {
  try {
    const { hostname } = new URL(url);
    return hostname.replace(/^www\./, '');
  } catch {
```

```
    return null;
  }
}

// Fetch domain creation date and calculate domain age (days)
function getDomainAge(domain) {
  return
fetch(`https://www.whoisxmlapi.com/whoisserver/WhoisService?apiKey=${WHOIS_API_KEY}&
domainName=${domain}&outputFormat=JSON`)
    .then(res => res.json())
    .then(data => {
      const createdDate = data.WhoisRecord?.createdDate;
      if (!createdDate) return { ageDays: null, createdDate: null };

      const created = new Date(createdDate);
      const now = new Date();
      const diffTime = Math.abs(now - created);
      const diffDays = Math.ceil(diffTime / (1000 * 60 * 60 * 24));
      return { ageDays: diffDays, createdDate };
    })
    .catch(err => {
      console.error("Whois error:", err);
      return { ageDays: null, createdDate: null };
    });
}

// Listener for messages from popup.js
chrome.runtime.onMessage.addListener((message, sender, sendResponse) => {
  if (message.action === "check_url") {
    const url = message.url;
    const domain = extractDomain(url);

    // Step 1: Submit URL to VirusTotal for scanning
    fetch("https://www.virustotal.com/api/v3/urls", {
      method: "POST",
      headers: {
        "x-apikey": API_KEY,
        "Content-Type": "application/x-www-form-urlencoded"
      },
      body: `url=${encodeURIComponent(url)}`
    })
    .then(res => res.json())
    .then(data => {
      const scanId = data.data.id;
```

```
      // Step 2: Fetch scan report using scanId
      return fetch(`https://www.virustotal.com/api/v3/analyses/${scanId}`, {
        headers: { "x-apikey": API_KEY }
      });
    })
    .then(res => res.json())
    .then(report => {
      const malicious = report.data.attributes.stats.malicious;

      // Step 3: Get domain age info
      return getDomainAge(domain).then(ageResult => {
        sendResponse({
          malicious,
          domainAge: ageResult.ageDays,
          createdDate: ageResult.createdDate
        });
      });
    })
    .catch(err => {
      console.error("VT or Whois error:", err);
      sendResponse({ error: "VirusTotal or Whois lookup failed." });
    });

    return true; // Keeps response channel open for async
  }
});
```

3. popup.html

Popup user interface displaying URL status and risk results.

html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Phishing Detector</title>

 <style>
    body { font-family: Arial; padding: 10px; }

    #status-bar {
      height: 10px;
```

```
    width: 100%;
    margin-bottom: 10px;
    border-radius: 4px;
   }

   .safe {
    background-color: #4CAF50; /* Green */
   }

   .warning {
    background-color: #FFC107; /* Yellow */
   }

   .danger {
    background-color: #F44336; /* Red */
   }

   .warning-text {
    color: red;
   }

   #result {
    font-size: 14px;
    padding: 4px;
   }
  </style>
</head>
<body>
  <h3>Phishing Detector</h3>
  <div id="status-bar"></div>
  <div id="result">Checking...</div>

  <script src="popup.js"></script>
</body>
</html>
```

4. popup.js

Client-side logic for heuristic URL checks and communicating with background script.

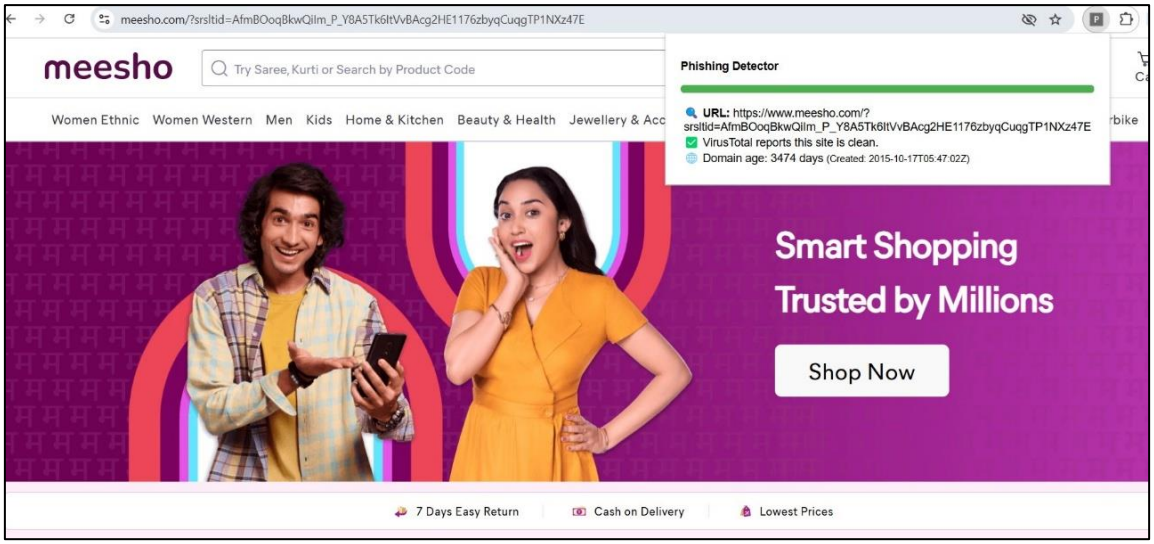
js

```
chrome.tabs.query({ active: true, currentWindow: true }, (tabs) => {
  const url = tabs[0].url;
```

```
  document.getElementById("result").innerHTML = `🔍 <strong>URL:</strong> ${url}`;

  const suspiciousWords = ["login", "verify", "update", "bank", "secure"];
  const isSuspicious = suspiciousWords.some(word => url.toLowerCase().includes(word));
  const isHTTP = !url.startsWith("https://");

  let riskLevel = 0;

  if (isSuspicious) {
    document.getElementById("result").innerHTML += `<br><span class="warning">⚠ Suspicious keywords detected!</span>`;
    riskLevel += 1;
  }

  if (isHTTP) {
    document.getElementById("result").innerHTML += `<br><span class="warning">⚠ This site does not use HTTPS!</span>`;
    riskLevel += 1;
  }

  // Request background.js for VirusTotal and Whois results
  chrome.runtime.sendMessage({ action: "check_url", url }, (response) => {
    if (response?.error) {
      document.getElementById("result").innerHTML += `<br>⚠ ${response.error}`;
      setStatusBar("warning");
      return;
    }

    if (response.malicious > 0) {
      document.getElementById("result").innerHTML += `<br><span class="warning">⚠ Reported as malicious by VirusTotal (${response.malicious} engines)!</span>`;
      riskLevel += 2;
    } else {
      document.getElementById("result").innerHTML += `<br>✅ VirusTotal reports this site is clean.`;
    }

    if (response.domainAge !== null) {
      document.getElementById("result").innerHTML += `<br>🌐 Domain age: ${response.domainAge} days <small>(Created: ${response.createdDate})</small>`;

      if (response.domainAge < 30) {
```

```
        document.getElementById("result").innerHTML += `<br><span class="warning">⚠ Domain is very new! Might be suspicious.</span>`;
        riskLevel += 1;
      }
    } else {
      document.getElementById("result").innerHTML += `<br>⚠ Could not determine domain age.`;
      riskLevel += 1;
    }

    if (riskLevel >= 3) {
      setStatusBar("danger");
    } else if (riskLevel > 0) {
      setStatusBar("warning");
    } else {
      setStatusBar("safe");
    }
  });
});

function setStatusBar(level) {
  const statusBar = document.getElementById("status-bar");
  statusBar.className = level;
}
```

7. Screenshots



8. Summary

This project combines browser extension development with real-time cybersecurity APIs to provide users with instant phishing detection. The layered approach (heuristic + VirusTotal + domain age) improves detection accuracy, helping users avoid dangerous websites.