|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Information Security (116U01L602)** | **Semester:** | **VI** |
| **Date of Performance:** | **11-02-2025** | **DIV/ Batch No:** | **C - 3** |
| **Student Name:** | **Romil Lodaya** | **Roll No:** | **16010122096** |

|  |
| --- |
| **Title: Implementation of CAPTCHA for Security of systems** |

|  |
| --- |
| **Objectives:** |
| * To understand the role of CAPTCHA in securing online systems. * To implement a CAPTCHA system for user verification. * To analyze the effectiveness of CAPTCHA against automated attacks. * To compare CAPTCHA with other network security mechanisms. |

|  |
| --- |
| **Expected Outcome of Experiment:** |
| **CO4:** Illustrate and Compare network security mechanisms |

|  |
| --- |
| **Books/ Journals/ Websites referred:** |
| Wikipedia  GeeksforGeeks |

|  |
| --- |
| **Pre Lab/ Prior Concepts:** |
| * Basics of web security and authentication. * Understanding of HTML, CSS, and JavaScript for CAPTCHA implementation. * Knowledge of how bots attempt to bypass authentication. |

|  |
| --- |
| **New Concepts to be learned:** |
| * Different types of CAPTCHA (Image-based, Text-based, Audio-based, reCAPTCHA). * How CAPTCHA prevents automated bots from accessing systems. * Implementing a CAPTCHA system using JavaScript. * Security mechanisms that complement CAPTCHA. |

|  |
| --- |
| **Abstract:** |
| CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a widely used security measure to differentiate between human users and automated scripts. This experiment focuses on implementing an image-based CAPTCHA system using HTML, CSS, and JavaScript. The effectiveness of CAPTCHA is evaluated against various attack techniques, and a comparison is made with other security mechanisms such as multi-factor authentication and bot detection algorithms. |

|  |
| --- |
| **Related Theory:** |
| * **Types of CAPTCHAS:**   + **Text-based CAPTCHA:** Requires users to recognize and enter distorted characters.   + **Image-based CAPTCHA:** Users select specific images based on a given prompt.   + **Audio CAPTCHA:** Provides an audio challenge for visually impaired users.   + **reCAPTCHA:** Google’s advanced CAPTCHA system using AI to distinguish bots. * **Working of CAPTCHA:**   + A CAPTCHA system presents a challenge to the user.   + The user must correctly complete the challenge to proceed.   + The system verifies the response to allow or deny access. * **Comparison with Other Network Security Mechanisms:**   + **Multi-Factor Authentication (MFA):** CAPTCHA protects against bots, while MFA adds an extra layer by requiring verification via OTP, email, or biometric authentication.   + **Honeypots:** Unlike CAPTCHA, honeypots attract and analyze bots rather than block them outright.   + **Behavioral Analysis:** AI-based models can analyze user behavior for bot detection, offering a more seamless experience compared to CAPTCHA. |

|  |
| --- |
| **Implementation Details:** |
| **1] Text CAPTCHA:**  <!DOCTYPE html>  <html lang="en">    <head>      <meta charset="UTF-8" />      <meta name="viewport" content="width=device-width, initial-scale=1.0" />      <title>Text-based CAPTCHA</title>      <style>        \* {          margin: 0;          padding: 0;          box-sizing: border-box;        }        body {          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;          background: #f4f7fc;          color: #333;          padding: 40px;          display: flex;          justify-content: center;          align-items: center;          height: 100vh;          flex-direction: column;        }        h2 {          color: #4f4f4f;          margin-bottom: 20px;        }        .captcha-container {          background-color: #fff;          padding: 20px 30px;          border-radius: 10px;          box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);          text-align: center;          width: 300px;        }        .captcha-text {          font-size: 36px;          font-weight: bold;          letter-spacing: 6px;          background-color: #f5f5f5;          padding: 15px;          border-radius: 6px;          display: inline-block;          margin-bottom: 20px;          color: #555;          user-select: none;        }        input,        button {          width: 100%;          padding: 12px;          margin: 10px 0;          border-radius: 8px;          border: 1px solid #ccc;          font-size: 16px;          transition: all 0.3s ease;        }        input:focus,        button:focus {          outline: none;          border-color: #007bff;          box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);        }        button {          background-color: #007bff;          color: #fff;          font-weight: bold;          cursor: pointer;        }        button:hover {          background-color: #0056b3;        }        #result {          font-size: 14px;          font-weight: 600;          margin-top: 15px;        }        #result.correct {          color: green;        }        #result.incorrect {          color: red;        }      </style>    </head>    <body>      <h2>Text-based CAPTCHA</h2>      <div class="captcha-container">        <div id="captcha" class="captcha-text">ABC123</div>        <input type="text" id="captchaInput" placeholder="Enter CAPTCHA" />        <button id="submitCaptcha">Submit</button>        <button id="refreshCaptcha">Refresh CAPTCHA</button>        <p id="result"></p>      </div>      <script>        function generateCaptcha() {          const chars =            "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";          let captcha = "";  *for* (let i = 0; i < 6; i++) {            captcha += chars.charAt(Math.floor(Math.random() \* chars.length));          }  *return* captcha;        }        function setCaptcha() {          const captchaText = generateCaptcha();          document.getElementById("captcha").textContent = captchaText;        }        document          .getElementById("refreshCaptcha")          .addEventListener("click", function () {            setCaptcha();            document.getElementById("result").textContent = "";          });        document          .getElementById("submitCaptcha")          .addEventListener("click", function () {            const captchaText = document.getElementById("captcha").textContent;            const userInput = document.getElementById("captchaInput").value;            const resultElem = document.getElementById("result");  *if* (userInput === captchaText) {              resultElem.textContent = "CAPTCHA verified successfully!";              resultElem.classList.add("correct");              resultElem.classList.remove("incorrect");            } *else* {              resultElem.textContent = "Incorrect CAPTCHA. Please try again.";              resultElem.classList.add("incorrect");              resultElem.classList.remove("correct");            }          });        setCaptcha();      </script>    </body>  </html>  **2] Distorted Text CAPTCHA:**  <!DOCTYPE html>  <html lang="en">    <head>      <meta charset="UTF-8" />      <meta name="viewport" content="width=device-width, initial-scale=1.0" />      <title>Distorted Text CAPTCHA</title>      <style>        \* {          margin: 0;          padding: 0;          box-sizing: border-box;        }        body {          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;          background: url("https://via.placeholder.com/800x600") no-repeat center            center fixed;          background-size: cover;          color: #fff;          padding: 40px;          display: flex;          justify-content: center;          align-items: center;          height: 100vh;          flex-direction: column;          text-align: center;        }        h2 {          color: #000;          margin-bottom: 20px;        }        .captcha-container {          background-color: rgba(255, 255, 255, 0.8);          padding: 30px 40px;          border-radius: 10px;          box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);          width: 400px;          text-align: center;        }        canvas {          margin: 20px 0;          border-radius: 8px;          box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2);        }        input,        button {          width: 100%;          padding: 12px;          margin: 10px 0;          border-radius: 8px;          border: 1px solid #ccc;          font-size: 16px;          transition: all 0.3s ease;        }        input:focus,        button:focus {          outline: none;          border-color: #007bff;          box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);        }        button {          background-color: #007bff;          color: #fff;          font-weight: bold;          cursor: pointer;        }        button:hover {          background-color: #0056b3;        }        #result {          font-size: 14px;          font-weight: 600;          margin-top: 15px;        }        #result.correct {          color: green;        }        #result.incorrect {          color: red;        }      </style>    </head>    <body>      <h2>Distorted Text CAPTCHA</h2>      <div class="captcha-container">        <canvas id="captchaCanvas" width="300" height="100"></canvas>        <br />        <input type="text" id="captchaInput" placeholder="Enter CAPTCHA text" />        <button id="submitCaptcha">Submit</button>        <button id="refreshCaptcha">Refresh CAPTCHA</button>        <p id="result"></p>      </div>      <script>        function generateCaptchaText() {          const chars =            "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";          let captcha = "";  *for* (let i = 0; i < 6; i++) {            captcha += chars.charAt(Math.floor(Math.random() \* chars.length));          }  *return* captcha;        }        function fillCanvasBackground(*ctx*, *canvas*) {          const gradient = *ctx*.createLinearGradient(            0,            0,  *canvas*.width,  *canvas*.height          );          gradient.addColorStop(0, "#800080");          gradient.addColorStop(0.5, "#4B0082");          gradient.addColorStop(1, "#9370DB");  *ctx*.fillStyle = gradient;  *ctx*.fillRect(0, 0, *canvas*.width, *canvas*.height);  *for* (let i = 0; i < 100; i++) {            const radius = Math.random() \* 12 + 3;  *ctx*.fillStyle = `rgba(${Math.random() \* 255}, ${              Math.random() \* 255            }, ${Math.random() \* 255}, 0.2)`;  *ctx*.beginPath();  *ctx*.arc(              Math.random() \* *canvas*.width,              Math.random() \* *canvas*.height,              radius,              0,              2 \* Math.PI            );  *ctx*.fill();          }  *for* (let i = 0; i < 10; i++) {  *ctx*.strokeStyle = `rgba(0,0,0,${Math.random() \* 0.2})`;  *ctx*.beginPath();  *ctx*.moveTo(              Math.random() \* *canvas*.width,              Math.random() \* *canvas*.height            );  *ctx*.lineTo(              Math.random() \* *canvas*.width,              Math.random() \* *canvas*.height            );  *ctx*.stroke();          }        }        function drawDistortedText() {          const captchaText = generateCaptchaText();          const canvas = document.getElementById("captchaCanvas");          const ctx = canvas.getContext("2d");          ctx.clearRect(0, 0, canvas.width, canvas.height);          fillCanvasBackground(ctx, canvas);          ctx.font = "bold 40px Arial";          ctx.textBaseline = "middle";  *for* (let i = 0; i < captchaText.length; i++) {            const x = 30 + i \* 40;            const y = canvas.height / 2 + (Math.random() - 0.5) \* 10;            const angle = (Math.random() - 0.5) \* 0.6;            ctx.save();            ctx.translate(x, y);            ctx.rotate(angle);            ctx.fillStyle = `rgba(0, 0, 0, ${Math.random() \* 0.8 + 0.2})`;            ctx.fillText(captchaText[i], 0, 0);            ctx.restore();          }  *for* (let i = 0; i < 5; i++) {            ctx.strokeStyle = `rgba(0,0,0,${Math.random() \* 0.3})`;            ctx.beginPath();            ctx.moveTo(              Math.random() \* canvas.width,              Math.random() \* canvas.height            );            ctx.lineTo(              Math.random() \* canvas.width,              Math.random() \* canvas.height            );            ctx.stroke();          }          document.getElementById("captchaInput").dataset.answer = captchaText;        }        drawDistortedText();        function verifyCaptcha() {          const userInput = document.getElementById("captchaInput").value;          const correctAnswer =            document.getElementById("captchaInput").dataset.answer;          const resultElem = document.getElementById("result");  *if* (userInput === correctAnswer) {            resultElem.textContent = "CAPTCHA verified successfully!";            resultElem.classList.add("correct");            resultElem.classList.remove("incorrect");          } *else* {            resultElem.textContent = "Incorrect answer. Please try again.";            resultElem.classList.add("incorrect");            resultElem.classList.remove("correct");          }        }        document          .getElementById("refreshCaptcha")          .addEventListener("click", function () {            drawDistortedText();            document.getElementById("result").textContent = "";            document.getElementById("captchaInput").value = "";          });        document          .getElementById("submitCaptcha")          .addEventListener("click", function () {            verifyCaptcha();          });        drawCaptchaText();      </script>    </body>  </html>  **3] Mathematical CAPTCHA:**  <!DOCTYPE html>  <html lang="en">    <head>      <meta charset="UTF-8" />      <meta name="viewport" content="width=device-width, initial-scale=1.0" />      <title>Mathematical CAPTCHA</title>      <style>        \* {          margin: 0;          padding: 0;          box-sizing: border-box;        }        body {          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;          background: #f4f7fc;          color: #333;          padding: 40px;          display: flex;          justify-content: center;          align-items: center;          height: 100vh;          flex-direction: column;        }        h2 {          color: #4f4f4f;          margin-bottom: 20px;        }        .captcha-container {          background-color: #fff;          padding: 20px 30px;          border-radius: 10px;          box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);          text-align: center;          width: 350px;        }        .captcha-question {          font-size: 28px;          font-weight: bold;          letter-spacing: 2px;          background-color: #f5f5f5;          padding: 15px;          border-radius: 6px;          display: inline-block;          margin-bottom: 20px;          color: #555;        }        input,        button {          width: 100%;          padding: 12px;          margin: 10px 0;          border-radius: 8px;          border: 1px solid #ccc;          font-size: 16px;          transition: all 0.3s ease;        }        input:focus,        button:focus {          outline: none;          border-color: #007bff;          box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);        }        button {          background-color: #007bff;          color: #fff;          font-weight: bold;          cursor: pointer;        }        button:hover {          background-color: #0056b3;        }        #result {          font-size: 14px;          font-weight: 600;          margin-top: 15px;        }        #result.correct {          color: green;        }        #result.incorrect {          color: red;        }      </style>    </head>    <body>      <h2>Complex Mathematical CAPTCHA</h2>      <div class="captcha-container">        <div id="captchaQuestion" class="captcha-question">Loading...</div>        <input          type="number"          id="captchaInput"          placeholder="Enter your answer"          step="any"        />        <button id="submitCaptcha">Submit</button>        <button id="refreshCaptcha">Refresh Question</button>        <p id="result"></p>      </div>      <script>        function generateComplexQuestion() {          const operators = ["+", "-", "\*", "/"];          let num1 = Math.floor(Math.random() \* 20) + 1;          let num2 = Math.floor(Math.random() \* 20) + 1;          let num3 = Math.floor(Math.random() \* 20) + 1;          let op1 = operators[Math.floor(Math.random() \* operators.length)];          let op2 = operators[Math.floor(Math.random() \* operators.length)];  *if* (op1 === "/") {            num2 = Math.floor(Math.random() \* 10) + 1; *// keep small divisor*            num1 = num2 \* (Math.floor(Math.random() \* 10) + 1);          }  *if* (op2 === "/") {            num3 = Math.floor(Math.random() \* 10) + 1;            num2 = num3 \* (Math.floor(Math.random() \* 10) + 1);          }          const expression = `${num1} ${op1} ${num2} ${op2} ${num3}`;          let answer = eval(expression);  *if* (!Number.isInteger(answer)) {            answer = parseFloat(answer.toFixed(2));          }  *return* { expression, answer };        }        function setMathQuestion() {          const { expression, answer } = generateComplexQuestion();          document.getElementById("captchaQuestion").textContent =            expression + " = ?";          document.getElementById("captchaInput").dataset.answer = answer;          document.getElementById("captchaInput").value = "";          document.getElementById("result").textContent = "";        }        document          .getElementById("refreshCaptcha")          .addEventListener("click", function () {            setMathQuestion();          });        document          .getElementById("submitCaptcha")          .addEventListener("click", function () {            const userInput = document.getElementById("captchaInput").value;            const correctAnswer =              document.getElementById("captchaInput").dataset.answer;            const resultElem = document.getElementById("result");  *if* (parseFloat(userInput) === parseFloat(correctAnswer)) {              resultElem.textContent = "CAPTCHA verified successfully!";              resultElem.classList.add("correct");              resultElem.classList.remove("incorrect");            } *else* {              resultElem.textContent = "Incorrect answer. Please try again.";              resultElem.classList.add("incorrect");              resultElem.classList.remove("correct");            }          });        setMathQuestion();      </script>    </body>  </html>  **4] Grid CAPTCHA:**  <!DOCTYPE html>  <html lang="en">    <head>      <meta charset="UTF-8" />      <meta name="viewport" content="width=device-width, initial-scale=1.0" />      <title>Image Grid CAPTCHA</title>      <style>        \* {          margin: 0;          padding: 0;          box-sizing: border-box;        }        body {          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;          background-color: #f4f7fc;          color: #333;          padding: 40px;          display: flex;          justify-content: center;          align-items: center;          height: 100vh;          flex-direction: column;          text-align: center;        }        h2 {          margin-bottom: 20px;        }        .captcha-container {          background-color: #fff;          padding: 30px 40px;          border-radius: 10px;          box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);          width: 600px;          text-align: center;        }        .image-grid {          display: grid;          grid-template-columns: repeat(3, 1fr);          gap: 15px;          margin-bottom: 20px;        }        .image-grid img {          width: 100%;          height: 120px;          object-fit: cover;          border-radius: 8px;          cursor: pointer;          transition: all 0.3s ease;        }        .image-grid img.selected {          border: 4px solid #007bff;          transform: scale(1.02);        }        .image-grid img:hover {          opacity: 0.8;        }        button {          width: 50%;          padding: 12px;          margin-top: 10px;          border-radius: 8px;          background-color: #007bff;          color: #fff;          font-weight: bold;          cursor: pointer;          border: none;        }        button:hover {          background-color: #0056b3;        }        #result {          font-size: 14px;          font-weight: 600;          margin-top: 15px;        }        #result.correct {          color: green;        }        #result.incorrect {          color: red;        }      </style>    </head>    <body>      <h2>Select All Images With Pedestrian Crossings</h2>      <div class="captcha-container">        <div class="image-grid" id="imageGrid">  *<!-- Images will be dynamically inserted here -->*        </div>        <button id="verifyCaptcha">Verify</button>        <button id="refreshCaptcha">Refresh</button>        <p id="result"></p>      </div>      <script>        const imageGrid = document.getElementById("imageGrid");        const verifyBtn = document.getElementById("verifyCaptcha");        const refreshBtn = document.getElementById("refreshCaptcha");        const resultElem = document.getElementById("result");        const imagePaths = [          { src: "./assets/crossing1.webp", correct: true },          { src: "./assets/crossing2.webp", correct: true },          { src: "./assets/road1.webp", correct: false },          { src: "./assets/crossing3.webp", correct: true },          { src: "./assets/road2.webp", correct: false },          { src: "./assets/crossing4.webp", correct: true },          { src: "./assets/road3.webp", correct: false },          { src: "./assets/crossing5.webp", correct: true },          { src: "./assets/crossing6.webp", correct: true },        ];        let selectedImages = new Set();        function createImage(*imagePath*) {          const img = document.createElement("img");          img.src = *imagePath*.src;          img.className = "captcha-image";          img.dataset.correct = *imagePath*.correct;          img.addEventListener("click", () => {            img.classList.toggle("selected");  *if* (img.classList.contains("selected")) {              selectedImages.add(img);            } *else* {              selectedImages.delete(img);            }          });  *return* img;        }        function shuffleImages() {          imageGrid.innerHTML = "";          selectedImages.clear();          resultElem.textContent = "";          resultElem.className = "";          const shuffledImages = [...imagePaths].sort(() => Math.random() - 0.5);          shuffledImages.forEach((*imagePath*) => {            const img = createImage(*imagePath*);            imageGrid.appendChild(img);          });        }        function verifyCaptcha() {          const allImages = Array.from(            imageGrid.getElementsByClassName("captcha-image")          );          const correctImages = allImages.filter(            (*img*) => *img*.dataset.correct === "true"          );          const selectedCorrectImages = Array.from(selectedImages).filter(            (*img*) => *img*.dataset.correct === "true"          );  *if* (            selectedCorrectImages.length === correctImages.length &&            selectedImages.size === correctImages.length          ) {            resultElem.textContent = "CAPTCHA verified successfully!";            resultElem.className = "correct";          } *else* {            resultElem.textContent = "Incorrect selection. Please try again.";            resultElem.className = "incorrect";          }        }        verifyBtn.addEventListener("click", verifyCaptcha);        refreshBtn.addEventListener("click", shuffleImages);        shuffleImages();      </script>    </body>  </html> |

|  |
| --- |
| **Results/Output:** |
| **1] Text CAPTCHA:**    **2] Distorted Text CAPTCHA:**    **3] Mathematical CAPTCHA:**    **4] Grid CAPTCHA:** |

|  |
| --- |
| **Conclusion:** |
| The experiment demonstrates CAPTCHA's effectiveness in blocking bots. While useful, it may impact usability. Combining CAPTCHA with MFA and AI-based security enhances protection. |

|  |
| --- |
| **Post-Lab Questions:** |
| **Q1] Discuss how CAPTCHA helps in improving web security.**  CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) enhances web security by preventing automated bots from performing malicious activities such as brute-force attacks, spam submissions, fake account creation, and credential stuffing. It ensures that only human users can access certain functionalities, thereby reducing fraud, data scraping, and unauthorized access. CAPTCHA also helps protect online services from Distributed Denial of Service (DDoS) attacks by adding a verification step before granting access.  **Q2] What are the different types of CAPTCHA? Is re-CAPTCHA different from CAPTCHA? Justify your answer.**  There are several types of CAPTCHA, including:   * **Text-based CAPTCHA**: Users identify and type distorted text. * **Image-based CAPTCHA**: Users select images matching a given category (e.g., "Select all images with traffic lights"). * **Audio CAPTCHA**: Users listen to a spoken phrase and type it out. * **Math-based CAPTCHA**: Simple arithmetic problems for users to solve. * **3D CAPTCHA**: Uses 3D rendering to create complex challenges. * **reCAPTCHA**: An advanced version of CAPTCHA developed by Google that uses AI to improve security and user experience.   Yes, reCAPTCHA is different from traditional CAPTCHA. While standard CAPTCHA relies on distorted text or images, reCAPTCHA leverages machine learning and behavioral analysis to verify users with minimal input. Versions like reCAPTCHA v2 require users to check a box ("I'm not a robot") or solve an image-based challenge, while reCAPTCHA v3 operates invisibly by analyzing user interactions to detect bots.  **Q3] List the limitations of CAPTCHA. Mention the alternatives to CAPTCHA which can overcome these limitations.**  **Limitations of CAPTCHA:**   * **User Inconvenience**: Some CAPTCHAs are difficult to read or solve, frustrating users. * **Accessibility Issues**: Audio and visual CAPTCHAs may be challenging for visually or hearing-impaired users. * **Automated Solving Techniques**: Advanced bots and AI-based solvers can bypass traditional CAPTCHA. * **Slow Performance**: CAPTCHA adds extra steps, slowing down user interactions. * **Mobile Usability Problems**: Small screens make CAPTCHAs harder to solve on mobile devices.   **Alternatives to CAPTCHA:**   * **reCAPTCHA v3**: Works in the background without user interaction, analyzing behavior. * **Biometric Authentication**: Uses fingerprints, facial recognition, or voice recognition for verification. * **HoneyPot Fields**: Invisible fields in forms that only bots fill out, helping detect automated scripts. * **Behavioral Analysis**: AI-driven techniques analyzing user behavior, such as mouse movements and keystrokes. * **One-Time Passwords (OTP)**: Sending an OTP to a registered device to verify human users. * **WebAuthn (FIDO2 Authentication)**: Using security keys or device-based authentication instead of CAPTCHA. |