Cyber Security 2

1 Cyber security, risico versus impact

Inleiding

In cybersecurity 1 hebben we gekeken naar een aantal soorten cyberaanvallen.

We gaan ons lijstje uitbreiden.

?? Soorten Cyberaanvallen – Overzicht

Ken je al dzee Cyberaanvallen?

Onder het plaatje worden al deze soorten Cyberaanvallen kort uitgelegd.

859932b3-7212-45af-a6dc-588ccea4d8fa.jpg

1. **Phishing,** "*The Digital Fishing Rod*"
   iemand probeert jou te misleiden om je wachtwoord af te geven (bv. via een nep-mail van je bank)
2. **Adware:** software dat ongevraagd advertenties toont.
3. **Virus**: speciaal soort malware dat zichzelf kan vermenigvuldigen (net als het Corona virus).
4. **SQL-injection:** via een formulier wordt je database gehackt
5. **Man-in-the-middle:** iemand onderschept je gegevens tussen jou en een website
6. **Trojan Horse**: Lijkt op legitieme software, maar voert op de achtergrond schadelijke acties uit.
7. **Randsomware**: Blokkeert je bestanden of systeem en eist losgeld om weer toegang te krijgen.
8. **Spyware**: Volgt je activiteiten op je apparaat zonder dat je het merkt, vaak om wachtwoorden of surfgedrag te stelen.
9. **Keylogger**: Registreert alles wat je typt, zoals wachtwoorden en privéberichten.

10. **Distributed Denial-of-Service (DDoS)**: Een aanval waarbij een website of server overspoeld wordt met verkeer en daardoor onbruikbaar wordt.
11. **Brute Force Attack**: Het systematisch proberen van alle mogelijke wachtwoorden tot het juiste gevonden is.
12. **SQL Injection**: Aanval via een invoerveld waarin schadelijke databasecode wordt ingevoerd.
13. **Cross-Site Request Forgery (CSRF**): De gebruiker wordt misleid om zonder het te weten een actie uit te voeren, zoals een betaling.

□□□ Test je kennis – Cyberaanvallen

Opdracht: Lees elk voorbeeld en probeer te raden om welke cyberaanval het gaat. Klik op de vraag om jezelf te controleren.

1. Je krijgt een e-mail van “de bank” met een link waarin je gevraagd wordt opnieuw in te loggen, maar het is een nepwebsite.

Phishing

2. Zodra je een gratis app installeert, verschijnen er overal pop-ups met reclame, zelfs als je de app niet gebruikt.

Adware

3. Een programma op je computer verspreidt zich automatisch naar andere bestanden en computers, net als een griepvirus.

Virus

4. Een website is opeens onbereikbaar omdat duizenden bots tegelijk verzoeken sturen naar de server.

DDoS-aanval

5. Je klikt op een “update”-melding, maar je installeert ongemerkt een programma dat op de achtergrond je bestanden versleutelt en geld vraagt om ze terug te krijgen.

Ransomware

6. Je vult een formulier in op een website, en een hacker gebruikt die invoer om toegang te krijgen tot de database.

SQL-injection

7. Iemand op hetzelfde wifi-netwerk leest mee met de gegevens die jij naar een website stuurt, zonder dat jij het merkt.

Man-in-the-middle

8. Je downloadt een handig ogend programma, maar zodra je het opent blijkt het kwaadaardige acties uit te voeren op je computer.

Trojan Horse

9. Je toetsenbordactiviteit wordt in het geheim vastgelegd zodat iemand je wachtwoord kan achterhalen.

Keylogger

10. Een hacker probeert tienduizenden verschillende wachtwoorden totdat hij de juiste vindt en toegang krijgt tot een account.

Brute Force Attack

11. Een kwaadaardige link zorgt ervoor dat jij zonder het te weten geld overmaakt via een formulier dat je niet hebt ingevuld.

Cross-Site Request Forgery (CSRF)

12. Je surft naar een website en zonder te klikken wordt automatisch schadelijke software op je computer gezet.

Drive-by Download

13. Je bezoekt een vertrouwde website, maar deze is ongemerkt gehackt en stuurt jou door naar een foute pagina.

Watering Hole Attack

14. Een hacker onderschept je login-sessie en gebruikt jouw sessie-ID om toegang te krijgen tot je account.

Session Hijacking

15. Je krijgt een sms van “de pakketdienst” met een link naar een track-and-trace pagina, maar die blijkt nep te zijn.

Smishing

16. Iemand installeert een app op jouw telefoon die in het geheim foto’s, locatie en berichten doorspeelt.

Spyware

17. Je gebruikt overal hetzelfde wachtwoord en een hacker probeert deze combinatie op andere websites.

Credential Stuffing

? Uitleg: Wat is het verschil tussen kans en impact?

Kans = Hoe groot is de kans dat het gebeurt?

Impact = Wat zijn de gevolgen als het gebeurt?

Risico is het product maal de impact.

> **Risico = Kans X Impact**

Dus **risico 2 x** zo hoog en **impact 2 x** zo hoog dan is het risico **4x hoger!**

image.png

□□ Voorbeelden

- ### Fiets wordt gestolen bij school
  - *Kans*: Hoog → je laat hem vaak ongeopend buiten staan.
  - *Impact*: Middel → je baalt, moet lopen, misschien nieuwe kopen.
- ### Telefoon valt in water
  - *Kans*: Laag → je past goed op.
  - *Impact*: Hoog → je bent alles kwijt: foto’s, apps, schoolwerk.
- ### Je vergeet een wachtwoord
  - *Kans*: Middel → gebeurt wel eens.
  - *Impact*: Laag → je kunt meestal resetten.

?? Opdracht

Lees de 6 scenarios en bepaal vervolgens van elke scenario het **risico** en de **impact** en licht je keuze toe.

image.png	### Scenario 1 Phishing Stel je voor: je krijgt een e-mail van “de Rabobank” met als onderwerp **“Dringende actie vereist – je account wordt geblokkeerd”**. In het bericht staat dat je onmiddellijk moet inloggen om je rekening te verifiëren. Er staat een link bij die lijkt op de echte site, maar eigenlijk leidt naar een nepwebsite. Zodra je daar je gegevens invult, krijgt een oplichter direct toegang tot je bankaccount.

image.png	## Scenario 2 Adware Je downloadt een gratis spelletje van een onbekende website. Na de installatie verschijnen er ineens overal pop-ups en reclames, zelfs als je de browser niet open hebt. Je wordt telkens doorgestuurd naar webwinkels of vage aanbiedingen. Je computer wordt trager, en sommige advertenties proberen je te laten klikken op gevaarlijke links. Dit is een typische **adware-infectie**: ongewenste software die reclame toont en soms zelfs je surfgedrag volgt.
image.png	## Scenario 3 Man in the Middle Je zit op een openbaar wifi-netwerk in de trein en logt in op een webshop. Wat je niet weet, is dat een hacker op hetzelfde netwerk zit en al het internetverkeer onderschept. Hij leest ongemerkt mee met de gegevens die jij verstuurt, zoals je inlognaam en wachtwoord. Dit is een **man-in-the-middle-aanval**: iemand zit letterlijk “tussen jou en de website in” en vangt alles op wat je verstuurt, zonder dat je het doorhebt.
image.png	## Scenario 4 Ransomware Je opent een bijlage in een e-mail die lijkt te komen van je school, bijvoorbeeld “roosterwijziging.pdf”. Maar zodra je het opent, wordt je scherm zwart en verschijnt er een melding: **“Je bestanden zijn versleuteld. Betaal €300 in bitcoins om ze terug te krijgen.”** Alle documenten, foto’s en schoolopdrachten op je laptop zijn geblokkeerd. Dit is een **ransomware-aanval**: je bestanden zijn gegijzeld en je moet losgeld betalen om ze terug te krijgen.
image.png	## Scenario 5 DDos aanval Je wilt je favoriete webshop bezoeken, maar de site laadt niet of geeft een foutmelding. Op dat moment voeren hackers een **DDoS-aanval** uit: duizenden computers sturen tegelijk nepverzoeken naar de server van de webshop, waardoor die overbelast raakt en voor echte bezoekers onbereikbaar wordt. De website stort tijdelijk in, terwijl er eigenlijk niets “stuk” is — hij wordt gewoon overspoeld met verkeer.

image.png	**Scenario 6 Brute Force** Je hebt een simpel wachtwoord zoals “welkom123” voor je schoolaccount. Een hacker probeert via een script duizenden combinaties van veelgebruikte wachtwoorden, één voor één, totdat jouw wachtwoord wordt geraden. Dit heet een **brute force-aanval**: de aanvaller probeert systematisch allerlei wachtwoorden uit tot hij toegang krijgt tot je account. Als je een zwak of kort wachtwoord gebruikt, ben je hier extra kwetsbaar voor.

Maak deze tabel af

Cyberaanval	Kans	Impact	Uitleg
1. Phishing	Laag	Hoog	Risico laag omdat we meestal de nep-email herkennen, maar als we er toch "*intrappen*" dan is de impact groot want dan kan je (veel) geld verliezen.
2. Adware			
3. Man in the Middle			
4. Ransomware			
5. DDoS			
6. Brute Force			

image.png

? Inleveren

Bepaal van scenario 2 t/m 6 de kans en de impact.

Leg daarna uit welke cyberaanval het **grootste risico** is en leg aan de hand van de kans en de impact uit waarom.

2 SQL Injection in PHP

? Leerdoelen

- Je begrijpt wat een SQL Injection is en waarom het gevaarlijk is.
- Je kunt zelf een simpele SQL Injection uitvoeren op een onveilige PHP-pagina.
- Je kunt uitleggen hoe je dit soort aanvallen voorkomt.

? Uitleg

Bij een **SQL Injection** wordt misbruik gemaakt van een fout in je code, waarbij de invoer van een gebruiker direct in een SQL-query wordt gezet. Als je niet oppast, kan een hacker zo gegevens uit je database stelen of zelfs verwijderen.

2.2 SQL-Injection-kl.jpg

Set-up

We gaan zelf hacken doormiddel van een SQL injection. We maken hiervoor een eenvouidge website met een login dialog.

1. Maak form.php

```
<?php
require_once "connection.php";

$message = "";
$username = "";

if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $username = $_POST["username"] ?? "";
    $password = $_POST["password"] ?? "";

    $sql = "SELECT * FROM users WHERE username = '$username' and password = '$password'";
    $stmt = $pdo->query($sql);
    $user = $stmt->fetch();

    echo "SQL Statement: $sql<br>User: $username<br>Password: $password<br>";

    if ($user) {
        $message = "Welkom, je hebt toegang tot de website!";
    } else {
        $message = "Ongeldige login.";
    }
```

```
}
?>

<div class="login-box" style="margin:60px;border:1px solid #000;padding:20px;border-
radius:10px;width:300px;" >
<h2>Login</h2>
<form method="post" action="">
    <label for="username">Username:</label><br>
    <input type="text" id="username" name="username" value="<?php echo
htmlspecialchars($username); ?>" required><br><br>

    <label for="password">Password:</label><br>
    <input type="text" id="password" name="password" required><br><br>

    <button type="submit">Login</button>
</form>
</div>
<?php if ($message !== ""): ?>
    <p><?php echo htmlspecialchars($message); ?></p>
<?php endif; ?>
```

2. Maak connection.php

```
<?php
$host = "localhost";
$db = "testdb";
$user = "root";
$pass = "";

$dsn = "mysql:host=$host;dbname=$db";

$pdo = new PDO($dsn, $user, $pass);
?>
```

3. Maak nu de database aan:

```
CREATE DATABASE IF NOT EXISTS testdb;
USE testdb;

CREATE TABLE IF NOT EXISTS users (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(100) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL
);
```

4. Test of alles werkt.

Maak gebruikersnaam en wachtwoord aan en test of je kunt aanloggen.

Boven de login drukken we de SQL query, de gebruikersnaam en wacthwoord af. Zo kunnen we precies volgen wat er gebeurt.

Onder de login staat of je succesvol bent ingelogd.

image.png

5. Aanvallen!

De query die je ziet, is zoiets als:

`SELECT * FROM users WHERE username = 'max' and password = 'max123'`

Als je het wachtwoord niet weet en je zou de query kunnen veranderen in zoiets als:

`SELECT * FROM users WHERE username = 'max' and password = 'xxx' or '1'`

Dan zou de query het wachtwoord xxx altijd goedkeuren want `password='xxx' or '1'` is altijd waar.

?? Opdracht – Voer een SQL Injection uit

□□ Probeer nu een wachtwoord in te vullen zodat de query altijd waar is en je dus met elk password toegnag kan krijgen!

?? Vervolgopdracht – Beveilig je formulier

1. Pas de code aan zodat je gebruik maakt van **prepared statements.**

2. Test of de SQL Injection nu nog werkt. Leg uit waarom het niet meer lukt.

? Reflectie

- Waarom is SQL-injection zo gevaarlijk voor een website?

- Wat zou een reden kunnen zijn dat een software developer geen gebruik maakt van technieken om SQL injection te voorkomen?

? Inleveren

- **Screenshot** van ingevuld form met de juiste gegevens waarmee je via SQL kan inloggen.
  Dit is dus een schermafdruk van de **geslaagde hack-poging** waarbij de query zichtbaar is.
- De aangepaste **code** waarbij de SQL injection niet meer mogelijk is.

3 Cross-Site Scripting (XSS)

? Leerdoelen

- Je weet wat Cross-Site Scripting (XSS) is en hoe het werkt.
- Je kunt zelf een XSS-aanval uitvoeren in een onveilige webpagina.
- Je kunt uitleggen hoe je je website kunt beschermen tegen XSS.

? Uitleg

Wat is XSS?

Cross-Site Scripting (XSS) is een aanval waarbij een aanvaller kwaadaardige scripts (zoals JavaScript) invoert in een formulier of URL. Deze scripts worden dan uitgevoerd in de browser van een andere bezoeker.

Stel: je maakt een gastenboek waar mensen een berichtje kunnen achterlaten. Als je hun invoer niet goed filtert, kan iemand dit invoeren:

```
<script>alert('Ik ben gehackt!')</script>
```

Iedereen die daarna het gastenboek bezoekt, krijgt dan deze melding te zien – het script wordt uitgevoerd alsof het van jouw site komt.

image.png

?? Opdracht – Voer een XSS-aanval uit

1. Maak een PHP-bestand `gastenboek.php` met het volgende formulier en afhandelingscode:

```
<form method="post">
  Naam: <input name="naam"><br>
  Bericht: <textarea name="bericht"></textarea><br>
  <input type="submit" value="Verstuur">
</form>

<?php
if ($_POST) {
  echo "<h3>Bericht ontvangen:</h3>";
  echo "<p>Van: " . $_POST['naam'] . "</p>";
  echo "<p>" . $_POST['bericht'] . "</p>";
}
?>
```

2. Voer nu als bericht het volgende in:

   ```
   <script>alert('XSS test')</script>
   ```

3. Wat gebeurt er?

?? Vervolgopdracht – Beveilig je formulier

1. Pas de PHP-code aan zodat gebruikersinvoer wordt ge-escaped:

```
echo "<p>Van: " . htmlspecialchars($_POST['naam']) . "</p>";
echo "<p>" . htmlspecialchars($_POST['bericht']) . "</p>";
```

2. Voer nogmaals het script in. Wat gebeurt er nu?

? Reflectie

- Leg uit waarom XSS gevaarlijk is en wat je ermee zou kunnen. Geef een concreet voorbeeld en leg dat uit.

? Inleveren

- Screenshot van de popup bij onveilige invoer
- Reflectie

4 Cross-Site Request Forgery (CSRF)

? Leerdoelen

- Je begrijpt wat Cross-Site Request Forgery (CSRF) is en waarom het gevaarlijk is.
- Je kunt uitleggen hoe een CSRF-aanval werkt met een voorbeeld.
- Je leert hoe je je webapplicaties kunt beschermen tegen CSRF.

? Uitleg

Wat is CSRF?

Cross-Site Request Forgery (CSRF) is een aanval waarbij een gebruiker wordt misleid om een actie uit te voeren op een website waarop hij is ingelogd, zonder dat hij het weet.

Bijvoorbeeld: je bent ingelogd op je bankwebsite in tabblad 1. In tabblad 2 bezoek je een foute website die automatisch een formulier verstuurt waarmee geld wordt overgemaakt vanaf jouw bankaccount.

Voorbeeld

```
<img src="https://bank.nl/maakover?naar=crimineel&bedrag=1000" style="display:none">
```

Als jij op dat moment bent ingelogd bij de bank, wordt dit verzoek uitgevoerd zonder dat je het merkt!

Jij voert dit script uit in de browser en de browser is (op een ander tabblad) ingelogt bij de bank. Het script wordt uitgeoverd en omdat je bent ingelogt zal de bank het script uitvoeren. De bank weet niet dat het script op een ander tabblad staat en een gevaarlijk script is.

2.4 CSRF2.jpg

?? Opdracht – Begrijp een CSRF-aanval

1. Maak een simpele bank applicatie

```
<?php
session_start();

if (!isset($_SESSION["betalingen"])) {
    $_SESSION["betalingen"] = [];
}

if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST["clear_all"])) {
    $_SESSION["betalingen"] = [];
    header("Location: " . $_SERVER["PHP_SELF"]);
    exit;
}

if ($_SERVER["REQUEST_METHOD"] === "POST" && !isset($_POST["clear_all"])) {
    $naar = $_POST["naar"] ?? "";
    $bedrag = $_POST["bedrag"] ?? "";

    $_SESSION["betalingen"][] = [
        "naar" => $naar,
        "bedrag" => $bedrag,
    ];
}

if ($_SERVER["REQUEST_METHOD"] === "GET" && (isset($_GET["naar"]) || isset($_GET["bedrag"])))
{
    $naar = $_GET["naar"] ?? "";
    $bedrag = $_GET["bedrag"] ?? "";

    $_SESSION["betalingen"][] = [
        "naar" => $naar,
        "bedrag" => $bedrag,
    ];
}
?>

<form method="post" action="">
  <input name="naar" placeholder="Naar">
  <input name="bedrag" placeholder="Bedrag">
```

```
  <input type="submit" value="Opslaan">
</form>
<form method="post" action="" style="margin-top: 8px;">
  <button type="submit" name="clear_all" value="1">Clear all</button>
</form>

<h3>Opgeslagen inzendingen</h3>
<?php if (count($_SESSION["betalingen"]) > 0): ?>
  <ul>
    <?php foreach ($_SESSION["betalingen"] as $betaling): ?>
      <li>
        Overgemaakt naar: <?php echo htmlspecialchars($betaling["naar"]); ?>
        voor €<?php echo htmlspecialchars($betaling["bedrag"]); ?>
      </li>
    <?php endforeach; ?>
  </ul>
<?php else: ?>
  <p>Nog geen inzendingen.</p>
<?php endif; ?>
```

Tests of die werkt.

2. Maak nu een pagina waarmee je een aanval gaat doen.

```
<h2>Welkom op de site voor gratis .....</h2>
<img src="http://localhost/Cyber2/CSRF/bank_form.php?naar=hacker&bedrag=1000"
style="display:none">
```

Het gaat erom dat er op deze site een verborgen hack zit, namelijk een plaatje dat geen plaatje is, maar een plaatje dat stiekum een bedrag overboekt. Om dit te laten werken moet je ervoor zorgen dat het path `//localhost/Cyber2/CSRF/bank_form.php` goed staat en naar de bank-applicatie wijst.

3. Voer de hack uit!

1. Ga naar de 'bank applicatie'.
2. Maak een bedrag over
3. Ga nu naar de pagina waarmee je de aanval wil doen, en laadt deze pagina. Er lijkt niets te gebeuren?
4. Ga terug naar de back appplicatie en voer nog een transactie uit.
5. Wat zie je?

□□ Maak een schermafdruk!

?? Vervolgopdracht – Bescherm je formulier

1. Gebruik een CSRF-token in je formulier:

Plaats deze code vlak nadat je de sessie hebt gestart.

```
if (!isset($_SESSION['csrf_token'])) {
  $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}
$token = $_SESSION['csrf_token'];
```

2 Controleer in bank.php of het token klopt

Plaats deze code na de if waar je controleert of je een bedrag wilt overmaken. Dus vlak voordat je de variabelen `$naar`, `$bedrag` een waarde geeft.

```
if ($_GET['csrf_token'] !== $_SESSION['csrf_token']) {
    die("CSRF aanval gedetecteerd!");
}
```

3. Test nu nog een keer of de hack nog werkt.

Als het goed is en je probeert nu een nieuwe aanval uit te voeren dan zie je:

`CSRF aanval gedetecteerd!`

□□ Maak een schermafdruk!

? Reflectie

- probeer in eigen woorden uit te leggen hoe SCRF protectie werkt.

? Inleveren

- Screenshot van de gelukte aanval waarbij de hacker geld heeft 'gestolen'
- Screenshot van de niet gelukte aanval.
- Beantwoord de refelctievraag in eigen woorden.

5 Hoe denken hackers en beveiligers?

? Inleiding

Om goed te begrijpen hoe je een systeem beveiligt, moet je ook snappen hoe een hacker denkt. Cybersecurity is eigenlijk een spel tussen aanvaller en verdediger. De één zoekt zwakke plekken, de ander probeert ze te dichten.

?? Twee perspectieven

De hacker (offensief)	De beveiliger (defensief)
Zoekt zwakke plekken in formulieren, wachtwoorden en instellingen.	Probeert kwetsbaarheden te herkennen vóórdat ze misbruikt worden.
Gebruikt tools als Burp Suite, Nmap of SQLMap om systemen te scannen.	Gebruikt logging, firewalls en updates om aanvallen te voorkomen of te blokkeren.
Denkt in mogelijkheden: “Wat gebeurt er als ik dit invoerveld misbruik?”	Denkt in risico’s: “Hoe kan ik dit voorkomen of beperken?”

? Denk als een hacker (maar handel als een ethisch hacker)

Een **ethisch hacker** (of *penetration tester*) gebruikt dezelfde technieken als een kwaadwillende hacker, maar met toestemming, om bedrijven te helpen hun beveiliging te verbeteren. In Nederland mag je **alleen testen met toestemming** (Responsible Disclosure). Zonder toestemming is het strafbaar onder de Computerwetgeving.

?? Opdracht – “Denk als een hacker”

Lees de volgende situaties en bedenk:

- Welke kwetsbaarheid wordt hier misbruikt?
- Hoe kun je dat voorkomen?

1. Een website toont direct wat je invoert zonder controle.

→ XSS, oplossing: `htmlspecialchars()`.

2. Een loginformulier stuurt gegevens via `GET` in plaats van `POST`.

→ Slechte beveiliging van wachtwoorden in URL, oplossing: gebruik POST en HTTPS.

3. Een student laat zijn laptop openstaan in de kantine.

→ Fysieke beveiliging, oplossing: schermvergrendeling, bewustzijn.

□□ Responsible Disclosure

Als je een kwetsbaarheid ontdekt, meld dit netjes bij de organisatie in plaats van het openbaar te maken. Veel bedrijven hebben een “Responsible Disclosure”-beleid: meldingen worden gewaardeerd en soms zelfs beloond.

□□ Wat mag niet

- Testen op systemen zonder toestemming
- Data kopiëren of bekijken die niet van jou is
- Andermans wachtwoorden gebruiken

□ Wat mag wél

- Testen binnen je eigen oefenomgeving (zoals localhost of testserver)
- Fouten melden op een veilige manier
- Anderen bewust maken van veilig gedrag

? Reflectie

- Wat is het verschil tussen een ethische hacker en een criminele hacker?
- Waarom is het belangrijk dat cybersecurityspecialisten ook creatief denken?
- Wat zou je doen als je per ongeluk een kwetsbaarheid ontdekt bij je stagebedrijf?

? Inleveren

- Beantwoord de refelctievragen in eigen woorden.

Revision #32
Created 2025-06-22 08:11:57 UTC by Max
Updated 2026-06-03 06:01:18 UTC by Max