Når data krysser grenser

Erlend Oftedal, BEKK
Knut Vidar Siem, Objectware

Communities in Action

10. mai 2010

The OWASP Foundation
http://www.owasp.org

Agenda

- Data og grenser
- Kontekster
- XSS
- XSS og kompliserende kontekster
- Injection attacks
- Beskyttelse

Data krysser grenser…

- …når den kommer inn i eller forlater applikasjonen
- …når den forlater et språk og går over i et annet språks kontekst

Når data krysser grenser må vi passe oss

- Innholdet har ikke alltid samme betydning
  - Encoding
  - Kontrolltegn/metategn

En webapplikasjoner er polyglot

- Java? C#? Ruby?
- HTML?
- Javascript?
- SQL?
- LDAP?
- Xpath?
- XML?

Eksempler på kryssing av grenser

- Applikasjon til bakenforliggende systemer
  - Java til SQL
  - C# til LDAP
  - Ruby til XPath
- Applikasjon til browser
  - C# til HTML
  - Java til javascript
  - Ruby til JSON

Grensevakt

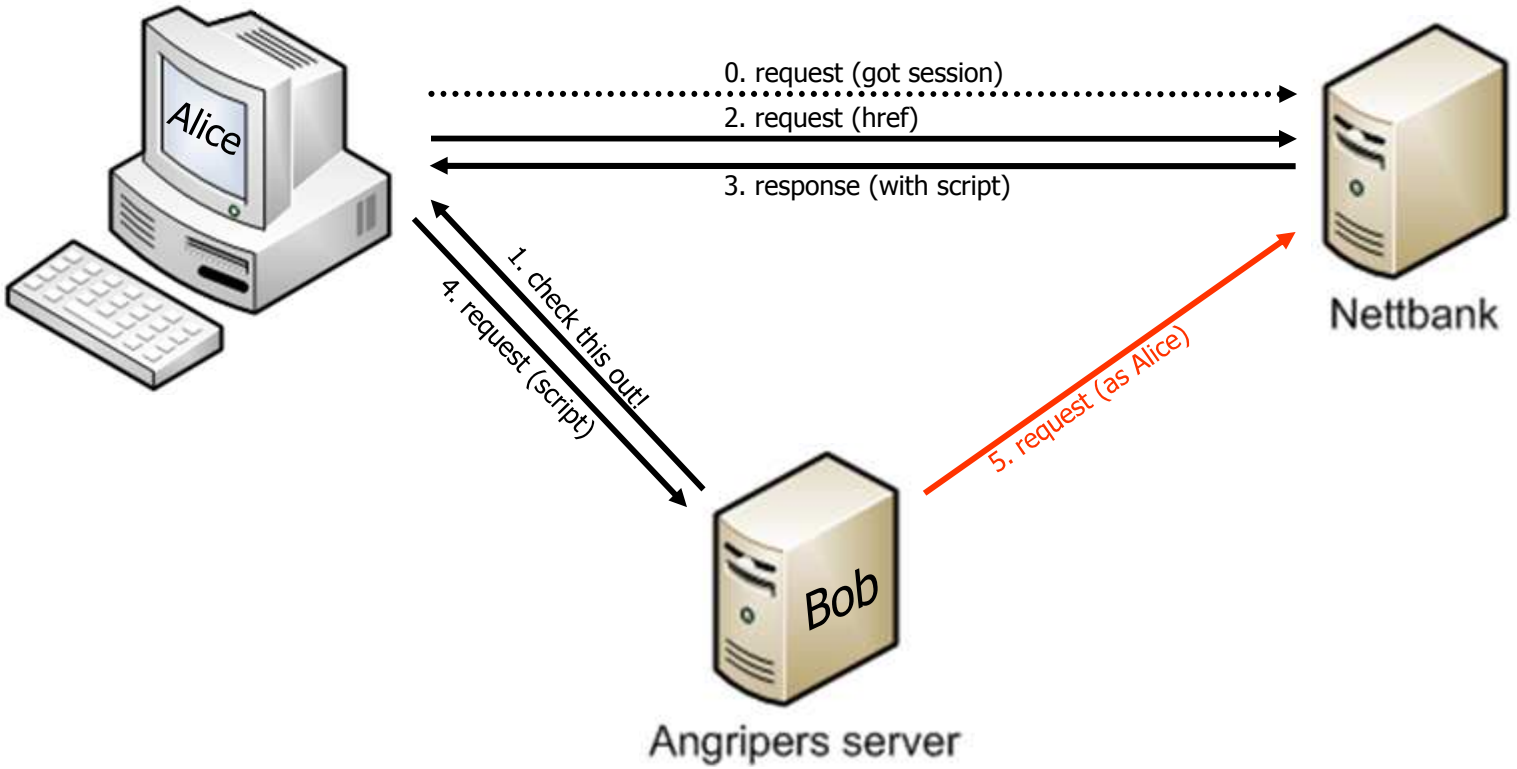
- Når vi krysser en grense må vi være på vakt
- Tilsynelatende like grenser skal nemlig ikke alltid behandles på samme måte

Cross Site Scripting

- .. også kjent som XSS
- Ukritisk behandling av (inn)data resulterer i kjøring av skadelig JavaScript i HTTP-responsen
- Muliggjør blant annet:
  - stjeling av sesjoner
  - phishing
  - endring av innhold på sider
  - utnyttelse av klientsårbarheter

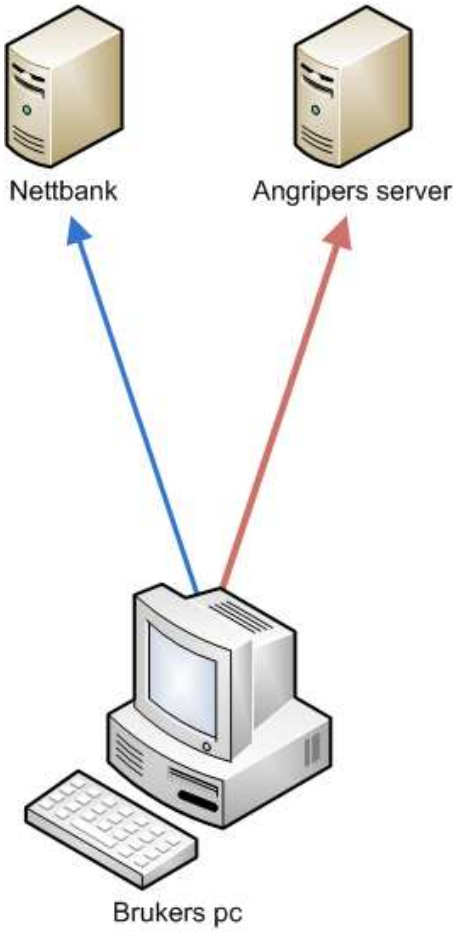
Attack Vectors	Security Weakness		Technical Impacts
Exploitability AVERAGE	Prevalence VERY WIDESPREAD	Detectability EASY	Impact MODERATE

Cross Site Scripting (illustrert #1)




```
nettbank.no/search.action?searchTerms=javaZone
nettbank.no/search.action?searchTerms=<script>.....</script>
```

Cross Site Scripting (illustrert #2)



Cross Site Scripting - Eksempel

- Søkefelt

- Bruker søker på:

test

- HTML:

```
...<input type="tex
```

- Bruker søker på:

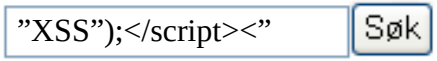
test”

- HTML:

```
...<input type="tex
```

- Bruker søker på:

test”><script>alert(”XSS”);</script><”



- HTML:

```
...<input type="text" value="test"><script>alert("XSS");</script><"">...
```

Demo

Hva gjør vi for å unngå dette da?

- Svaret man får er ofte enten
  - ”Valider all input!”
  - ”Encode all output!”
- Ingen av svarene er direkte feil, men…

Inputvalidering er ikke nok

- Hvordan validerer man rik input?
  - En kommentar på stackoverflow

Man kan klare seg med outputencoding, men…

- Man må encode riktig
- I HTML finnes det seks forskjellige områder
  1. Mellom tagger
  2. HTML attributt-verdier
  3. Javascript variabler
  4. CSS
  5. URLer (<a href="…">)
  6. Alle andre steder (kommentarer, attributtnavn, tagnavn osv.)
- Disse må håndteres på forskjellig måte!

Eksempel på problematikken

```
<html>
<body>
<script>
var a = "\x3C/script\x3E\x3Cscript\x3Ealert('xss');\x3C/script\x3E";
</script>
</body>
</html>
```

```
<html>
    <head>  </head>
    <body>
        <script>
            1
            2  var a = "
        </script>
        <script>
            1  alert('xss');
        </script>
      ";
    </body>
</html>
```

Demo av kontekstproblematikk

Hvordan gjør man det riktig?

1. Mellom tagger
   - Vanlig HTML escaping (”<” blir til ”<” osv.)
2. HTML-attributter
   - Escape alle ikke-alfanumeriske tegn med &#xHH;
3. Javascript-variabler
   - Escape alle ikke-alfanumeriske tegn med \xHH
4. CSS
   - Escape alle ikke-alfanumeriske tegn med \xHH

Hvordan gjør man det riktig?

5. URL-er
   - Bruk URL-encoding (%HH)
6. Alle andre steder
   - Unngå å legge data her med mindre du har en VELDIG god HTML-vasker (for eksempel OWASP AntiSamy)

http://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet

Injection attacks

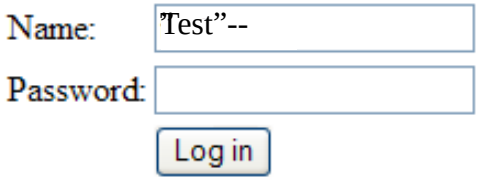
- SQL-injection
- LDAP injection
- Xpath injection
- QL injection
- Command-line injection

Attack Vectors	Security Weakness		Technical Impacts
Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE

SQL-injection

- Inndata brukes ukritisk i en SQL-spørring
- Gir angriper mulighet til å endre SQL-spørringen som kjøres mot databasen
- Muliggjør uautorisert tilgang til data:
  - Lese
  - Endre
  - Slette

SQL-injection – Eksempel



- Bruker skriver inn brukernavn og passord
- SQL:

```
SELECT * FROM User
WHERE Username="Test" AND Password="test123"
```

- Hva skjer hvis bruker skriver inn brukernavnet " ?
- SQL:

```
SELECT * FROM User
WHERE Username="" AND Password=""
```

- Hvordan kan dette utnyttes?
- -- eller # er SQL-kommentartegn
- SQL:

```
SELECT * FROM User
WHERE Username="Test" --" AND Password=""
```

- SQL - Alternativ:

```
SELECT * FROM User
WHERE Username="" OR "1"="1" AND Password="" OR "1"="1"
```

SQL-injection - Eksempel

- I mange programmeringsspråk kan en SQL-spørring inneholde flere spørringer
- SQL – tap av data:

```
SELECT * FROM Pages
WHERE PageId=9; DELETE FROM Users; -- ORDER BY Date
```

- SQL – henting av uautorisert data:

```
SELECT Title, Body FROM Forum
WHERE Author="" UNION ALL SELECT Username AS Title,
Password AS Body FROM Users WHERE ""=""
```

Demo

Hva gjør vi for å unngå dette da?

- Svaret man får er ofte
  - ”Valider all input!”
- Igjen – dette er ikke direkte feil, men…

Det handler om kontekst

- Hva gjør vi med navnet O'Brian når vi gjør inputvalidering?

- Vi må behandle kontrolltegn i det data blir en del av en spørring
- Heldigvis har noen løst dette for oss

Prepared statements

- Escaper kontrolltegn for oss…
- …så lenge vi bruker dem riktig

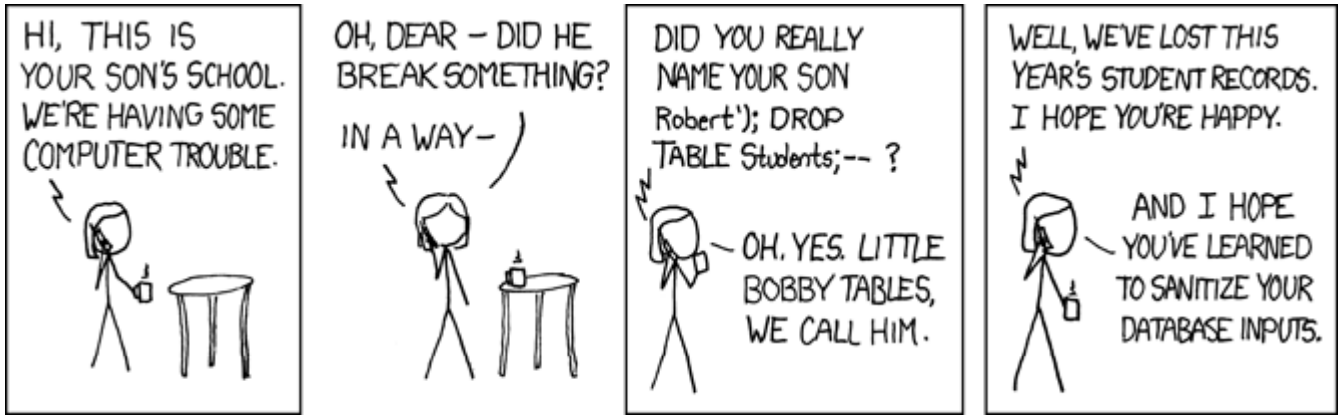
```
PreparedStatement updateSales =
con.prepareStatement("UPDATE COFFEES SET SALES = ? WHERE COF_NAME LIKE ? ");
updateSales.setInt(1, 75);
updateSales.setString(2, "Colombian");
updateSales.executeUpdate();
```

```
PreparedStatement updateSales =
con.prepareStatement("UPDATE COFFEES SET SALES = ? WHERE COF_NAME LIKE \""
+ coffeeName + "\"");
updateSales.setInt(1, 75);
updateSales.executeUpdate();
```

Generelt om injection angrep

- Escaping må gjøres i det man bytter kontekst
- Sjekk om det finnes ferdige bibliotek som løser problemet
  - OWASP ESAPI har et encoding/escaping-bibliotek og finnes for flere plattformer (.NET, Java ++)
- Hvis du skal escape selv
  - Få med alle kontrolltegn
  - Husk at escapetegnet i seg selv er et kontroll tegn
    - ’ escapes til \’
    - \’ escapes til \\\’

Spørsmål?



[xkcd.com]

Bli med i OWASP Norway og lær mer om sikkerhet i webapplikasjoner. Det er gratis å være med!

http://www.owasp.org/index.php/Norway