Security Issues in Web Applications

Vitaly Shmatikov
User Input Validation

Web applications need to reject invalid inputs
- “Credit card number should be 15 or 16 digits”
- “Expiration date in the past is not valid”

Traditionally done at the server
- Round-trip communication, increased load

Better idea (?): do it in the browser using client-side JavaScript code
Client-Side Validation

```javascript
onSubmit =
validateCard();
validateQuantities();
```

Validation Ok?

- Yes
  - send inputs to server

- No
  - reject inputs

[Bisht et al.]
Problem: Client Is Untrusted

Previously rejected values sent to server

Inputs must be re-validated at server!
Online Shopping

Client-side constraints:
- \( \text{quantity1} \geq 0 \)
- \( \text{quantity2} \geq 0 \)

Server-side code:
- \( \text{total} = \text{quantity1} \times \text{price1} + \text{quantity2} \times \text{price2} \)

Vulnerability: malicious client submits negative quantities for unlimited shopping rebates

Two items in cart: \( \text{price1} = $100 \), \( \text{price2} = $500 \)
- \( \text{quantity1} = -4 \), \( \text{quantity2} = 1 \)
- \( \text{total} = $100 \) (rebate of $400 on price2)
Online Banking

Client-side constraints:
from IN (Accnt1, Accnt2)
to IN (Accnt1, Accnt2)

Server-side code:
transfer money from \rightarrow to

Vulnerability: malicious client submits arbitrary account numbers for unauthorized money transfers
IT Support

Client-side constraints:
- `userId == 96` (hidden field)

Server-side code:
Update profile with id 96 with new details

Vulnerability: update arbitrary account

Inject a cross-site scripting (XSS) payload in admin account, cookies stolen every time admin logged in

[Bisht et al.]
Content Management

Server-side code:

```plaintext
privilege = non-admin;
if ( _COOKIE['make_install_prn'] == 1 )
    privilege = admin;
```

Vulnerability: malicious client sets make_install_prn cookie, creates fake admin account
Cashier-as-a-Service

Anyone can register an Amazon seller account, so can Chuck
- Purchase a $25 MasterCard gift card by cash, register under a fake address and phone number
- Create seller accounts in PayPal, Amazon and Google using the card

Chuck’s trick
- Check out from Jeff, but pay to “Mark” (Chuck himself)
- Amazon tells Jeff that payment has been successful
- Jeff is confused, ships product

Chuck, pay in Amazon with this signed letter:
Dear Amazon,
order#123 is $10, when it is paid, text me at 425-111-2222.
[Jeff’s signature]

Amazon, I want to pay with this letter
Dear Amazon,
order#123 is $10, when it is paid, text me at 425-111-2222.
[Jeff’s signature]
[Mark’s signature]

Hi, $10 has been paid for order#123.
[Amazon’s signature]

Great, I will ship order#123!

Anyone can register an Amazon seller account, so can Chuck
• Purchase a $25 MasterCard gift card by cash, register under a fake address and phone number
• Create seller accounts in PayPal, Amazon and Google using the card

Chuck’s trick
• Check out from Jeff, but pay to “Mark” (Chuck himself)
• Amazon tells Jeff that payment has been successful
• Jeff is confused, ships product

Hi, $10 has been paid for order#123.
[Amazon’s signature]
Interspire + PayPal Express

Session 1: pay for a cheap order (orderID1), but prevent the merchant from finalizing it by holding Message B

Session 2: place an expensive order (orderID2), but skip the payment step

Message A redirects to store.com/finalizeOrder?[orderID1]store

Message B calls store.com/finalizeOrder?[orderID1]store

Expensive order is checked out but the cheap one is paid!
Side-Channel Leaks


encrypted!
privacy problems solved?

Attacker can still see the number of packets, size of each packet, time between packets...
Search using encrypted Wi-Fi (WPA / WPA2)
Example: user types “l-i-s-t” on his laptop...

Consequence: any eavesdropper knows our search queries

Attacker’s effort linear in the size of query

Consequence: any eavesdropper knows our search queries
Online Medical Application

◆ Entering health records
  - By typing – auto-suggestion
  - By mouse – a tree structure of elements

◆ Finding a doctor
  - Dropdown list

[Chen et al.]

2000x reduction in ambiguity

Uniquely identify the specialty
Tax Preparation Application

- Wizard-style questionnaire
  - Tailor the questions based on previous inputs
- Which forms you work on reveal filing status, big medical bills, adjusted gross income...
- Knowing the state machine of the application, the eavesdropper can infer sensitive information
  - Especially by combining information learned from multiple state machines

[Chen et al.]
Child Credit State Machine

- Entry page of deductions & credits
- Summary of deductions & credits
- Partial credit
- Full credit
- Not eligible

All transitions have unique traffic patterns

Consult the IRS instruction:
$1000 for each child
Phase-out starting from $110,000. For every $1000 income, lose $50 credit.
Even worse, most decision procedures for credits/deductions have asymmetric paths: eligible – more questions, not eligible – no more questions.
Some Identifiable AGI Thresholds

[Chen et al.]

- Disabled Credit
  - $0
- Earned Income Credit
  - $24,999
  - $41,646
- Retirement Savings
  - $5,300
- College Expense
  - $53,000
- IRA Contribution
  - $116,000
- Student Loan Interest
  - $85,000
  - $105,000
  - $115,000
- Child Credit
  - $110,000
  - $130,000 or $150,000 or $170,000...
- First-time Homebuyer Credit
  - $150,000
  - $170,000
- Adoption Expense
  - $174,730
  - $214,780
Online Investments

Which funds you invest in?

- Each price history curve is a GIF image from MarketWatch
  - Anyone in the world can get them from this website
- Just compare the image sizes!

Your investment allocation?

- Can see the size of the pie chart, but hundreds of pie charts have the same image...
Change Over Time Is Revealing!

Financial institution updates your pie chart every day after market close. Mutual fund prices are public knowledge.
Still have the asymmetric path problem

Google’s responses are compressed, destination networks may or may not uncompress responses

- For example, Microsoft gateways uncompress and inspect Web traffic, but university does not
- Round before compression – university still sees distinguishable sizes; after compression – Microsoft does

Random padding is not appropriate

- If user checks several times, repeated random padding of the same responses quickly degrades effectiveness
- Images come from MarketWatch, not site itself
Applications rely on OS abstractions to improve their safety and reliability

- “Process”, “User”

Case study: Web browsers


Fork a new process

xbank.com

quickdate.com

OS isolation
Unintended Consequences

**Good**
- Better isolation
- Better reliability
  - Others not affected if one process crashes
- Better safety

**Bad**
-Leaks more info to concurrent processes
ProcFS in Multi-User OS

Introduced in the 1980s

Tom Killian
"Processes as Files" (1984)
“Noone Uses Multi-User OS Anymore”
Multi-User Isolation

UNIX multi-users in the 1980s
Android Sandboxing

Android “multi-users” today
Android Apps as “Users”

• Different apps run as different users

Android uses OS “user” abstraction to isolate applications
ProcFS Did Not Go Away

ProcFS API is still unchanged!

Android “multi-users” today
This Is Not Just About Android
What Can Be Learned from ProcFS?

- No permissions needed to read any world-readable file in ProcFS ...
  - IP addresses of network connections
  - Value of stack pointer
  - Various statistics
    - Packet counters
    - Number of context switches / CPU scheduling statistics
    - Memory usage
  - TCP sequence number inference
  - Keystroke sniffing
  - “Memento” attacks
Putting Memory Streams Together
Memprint: Stream of Memory Usage

10568 KB  15976 KB  49380 KB
65948 KB  11632 KB  48996 KB
60280 KB  60820 KB  59548 KB
Sniffing Memory Footprints

browser process

Parsing JavaScript...
Rendering images...

alloc 1
alloc 2

OS free page pool

OS isolation

memprint

zero-permission malicious process

used page count

2050
Sniffing Memory Footprints

- Parsing JavaScript...
- Rendering images...
- browser process
- alloc 1
- alloc 2
- OS free page pool
- brk/mmap
- memprint
- used page count
- zero-permission malicious process
- 2050-2056
- 2056

OS isolation
Sniffing Memory Footprints

- Browser process
- Alloc 1
- Alloc 2
- OS free page pool
- Parsing JavaScript...
- Rendering images...
- Brk/mmap
- OS isolation
- Used page count
- Memprint
- Zero-permission malicious process
- 2050, 2056, 2080
Loading BeNaughty.com in Chrome
Loading BeNaughty.com in Chrome
Loading BeNaughty.com in Chrome
Full Attack

OS isolation

browser

zero-permission app

memprint

/database

/proc/pid/statm

Why the Attack Works

◆ Memprints are **unique** - for up to 43% of Alexa top 100,000 pages
  • Can tune recognition to achieve zero false positives
◆ Memprints are **stable** across repeated visits to the same page

memprints are OS/browser-dependent but machine-independent
Cross-Page Similarity

similarity = Jaccard index of memprints

distinguishable

Similar to themselves

Different from others
Other Privacy Leaks

◆ Fine-grained memory dynamics reveal membership in dating sites, interest in medical conditions, etc.

◆ Dynamics of CPU scheduling reveal individual keystrokes

◆ General problem: fine-grained resource usage statistics are correlated with secrets
  • These statistics are visible across isolation boundary
  • Their dynamics are a high-bandwidth side channel