COMP6843 - Topic 3

More types of Injection

A NOTE ON ETHICS / LEGALITY

- UNSW hosting this course is an extremely important step forward.
- We expect a high standard of professionalism from you, meaning:
  - Respect the property of others and the university
  - Always abide by the law and university regulations
  - Be considerate of others to ensure everyone has an equal learning experience
  - Always check that you have written permission before performing a security test on a system

Always err on the side of caution. If you are unsure about anything ask one of the course staff!

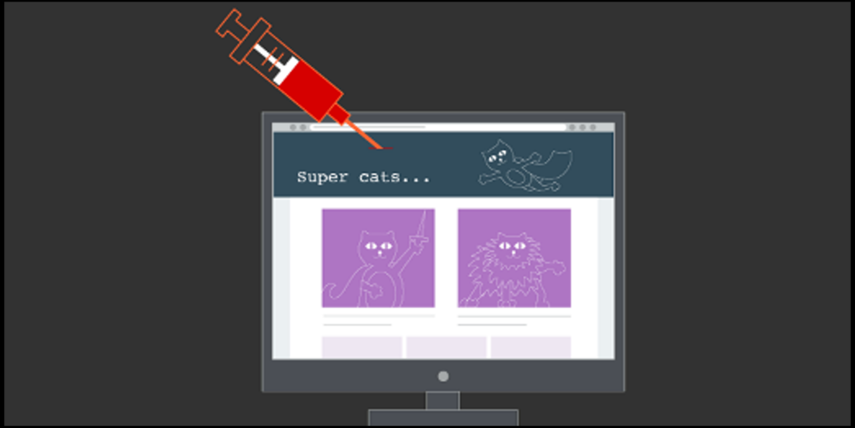
What are we going to learn today

Template Injection

CSV Injection

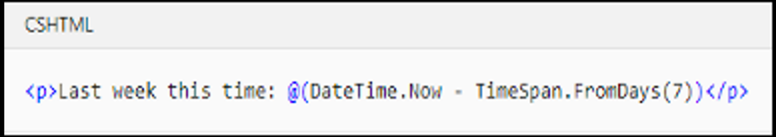
Play with Command Injection &
LFI

Understanding template injections



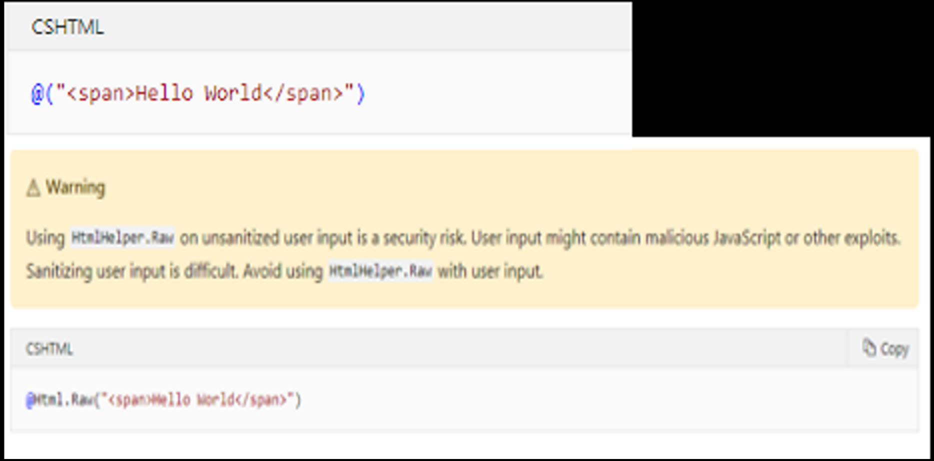
What are templates?

Server Side Templates



CSHTML

```
<p>Last week this time: @(DateTime.Now - TimeSpan.FromDays(7))</p>
```



CSHTML

```
@("<span>Hello World</span>")
```

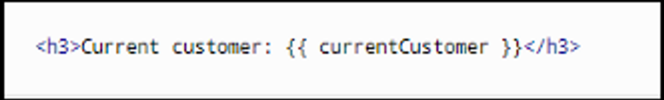
⚠ Warning

Using `HtmlHelper.Raw` on unsanitized user input is a security risk. User input might contain malicious JavaScript or other exploits. Sanitizing user input is difficult. Avoid using `HtmlHelper.Raw` with user input.

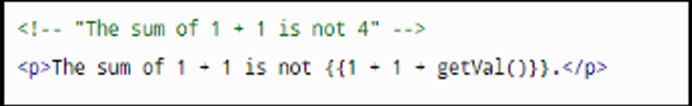
CSHTML Copy

```
@Html.Raw("<span>Hello World</span>")
```

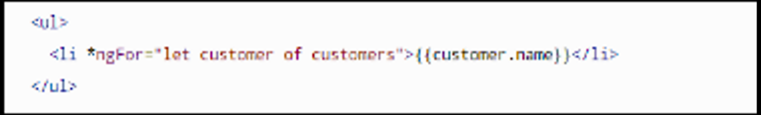
Client Side Templates



```
<h3>Current customer: {{ currentCustomer }}</h3>
```



```
<!-- "The sum of 1 + 1 is not 4" -->
<p>The sum of 1 + 1 is not {{1 + 1 + getVal()}}.</p>
```



```
<ul>
  <li *ngFor="let customer of customers">{{customer.name}}</li>
</ul>
```

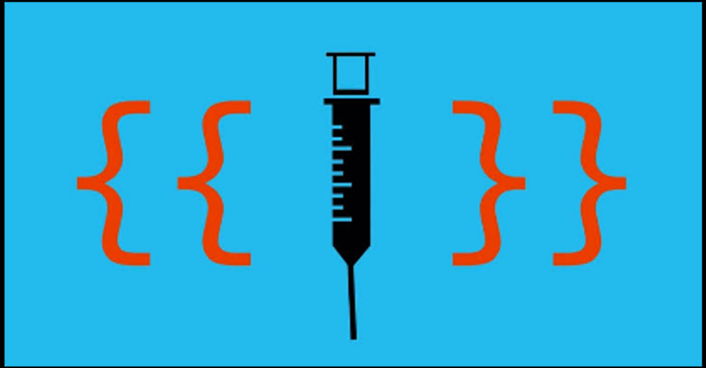
Template Injection

- Inputs from users ending up directly in templates without any validation or sanitisation.
- Could lead to Remote Code Execution (RCE), Cross Site Scripting (XSS).
- Could be classified as Server-Side and Client-Side Template Injection.

Server-Side Template Injection

- User input reflected directly in the server-side template engines.
- Attacker might be able to compromise the server.
- Make sure secure patterns are used when user inputs are passed into templates.

Server-Side Template Injection - DEMO

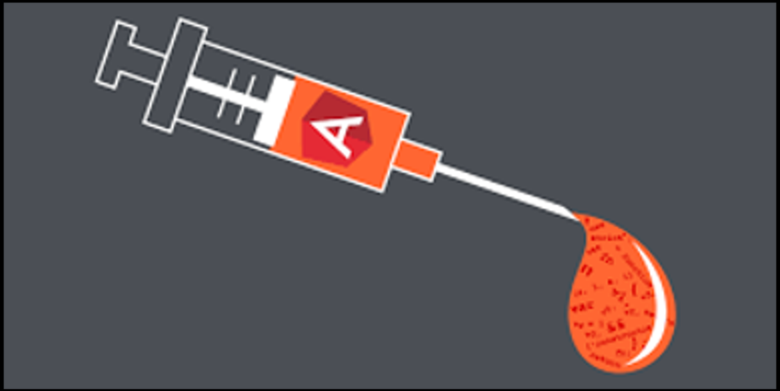


```
{% for x in ().__class__.__base__.__subclasses__() %}
      {% if "warning" in x.__name__ %}
           {{
             x()._module.__builtins__['__import__']
             ('os').popen("cat /etc/passwd").read()
           }}
       {%endif%}
{% endfor %}
```

Client-Side Template Injection

- When user input enters the template context without any validation.
- Would lead to XSS in most cases.
- DOM manipulation.

Client-Side Template Injection - DEMO



Payload:

```
{{constructor.constructor('alert(1)')()}}
```

What is this?

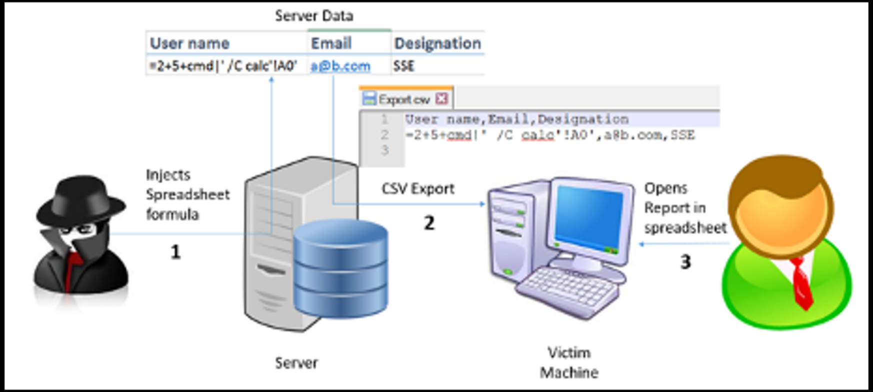
- Angular expressions are evaluated against the Scope object.

  `$scope.constructor.constructor()`

Recommendation

- Do not use user inputs to create dynamic templates.
- Do not mix server-side templates with client-side templates.
- Input validation.
- Follow framework provided recommendations.
- Separation between user input and data.

Understanding CSV injections



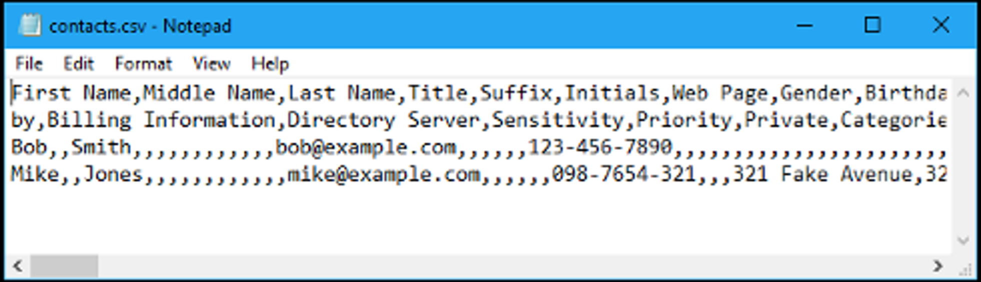
User name	Email	Designation
=2+5+cmd|' /C calc'!A0'	a@b.com	SSE

```
User name,Email,Designation
=2+5+cmd|' /C calc'!A0',a@b.com,SSE
```

What is CSV?

Comma-Separated-Values

- File extension: .csv
- Flat files, defined for data only.

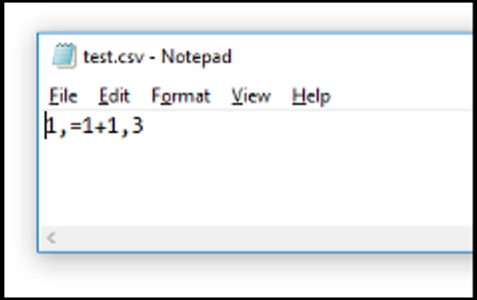


```
First Name,Middle Name,Last Name,Title,Suffix,Initials,Web Page,Gender,Birthda
by,Billing Information,Directory Server,Sensitivity,Priority,Private,Categorie
Bob,,Smith,,,,,,,,,,,,bob@example.com,,,,,,123-456-7890,,,,,,,,,,,,,,,,,,,,,,,,
Mike,,Jones,,,,,,,,,,,,mike@example.com,,,,,,098-7654-321,,,321 Fake Avenue,32
```

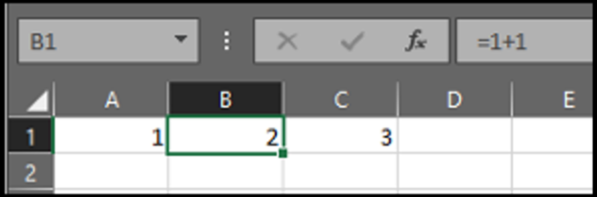
What data can we put in the file?

CSV Formula Injection

- Cells beginning with = are interpreted as formulas by Excel (and other applications).



```
1,=1+1,3
```



	A	B	C	D	E
1	1	2	3		
2					

Formulas that hurt!

So why is this dangerous?

Formulas can be used for multiple kinds of malicious payloads, for example:

- Create fake hyperlinks.
- Use Excel DDE (Dynamic Data Exchange) to execute commands (Excel only).

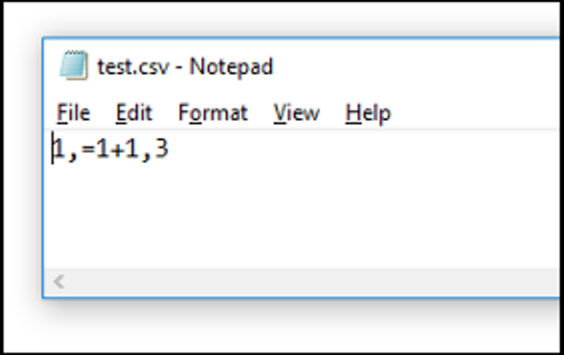
```
=cmd|' /C notepad'!'A1'
```

Cell begins with = (indicates a formula to Excel)

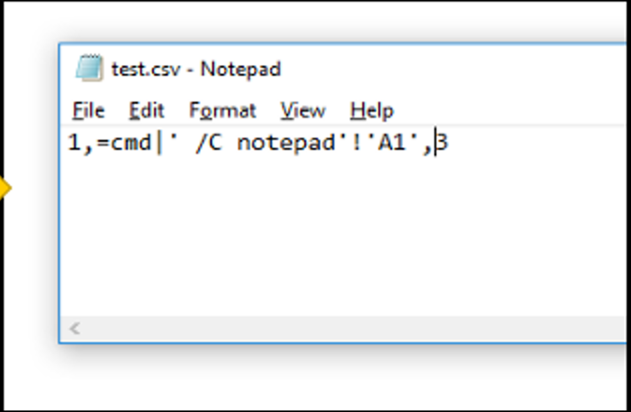
'Filename' gets directly executed as

With arguments of (run command 'notepad')

Cell reference: forces Excel to treat the preceding string as a 'filename'

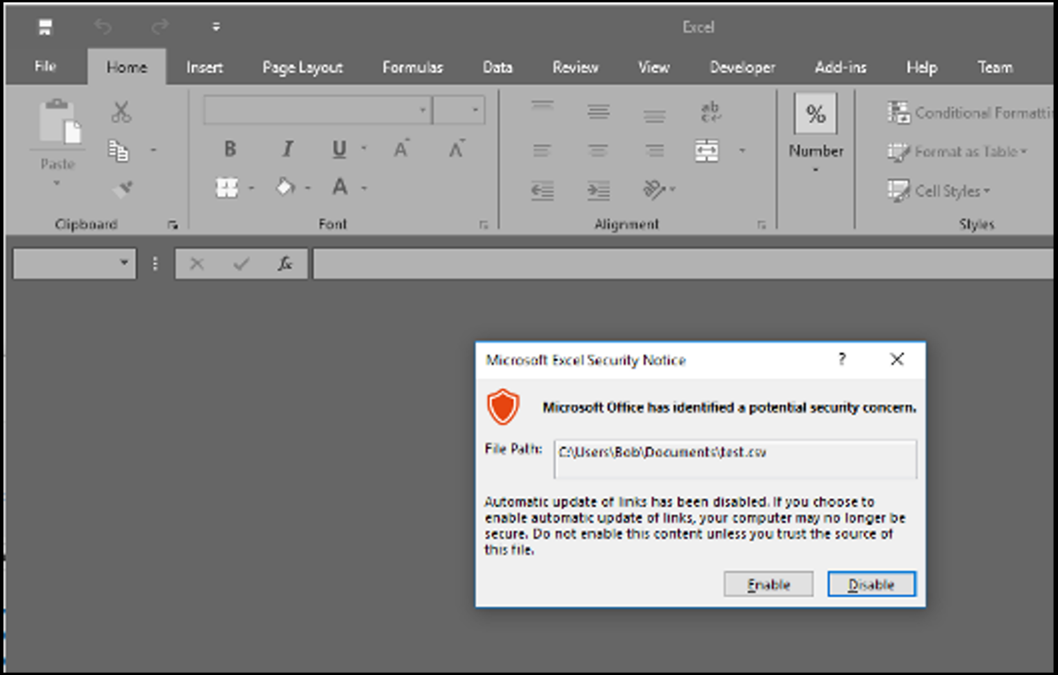


```
1,=1+1,3
```



```
1,=cmd|' /C notepad'!'A1',3
```

What happens next?



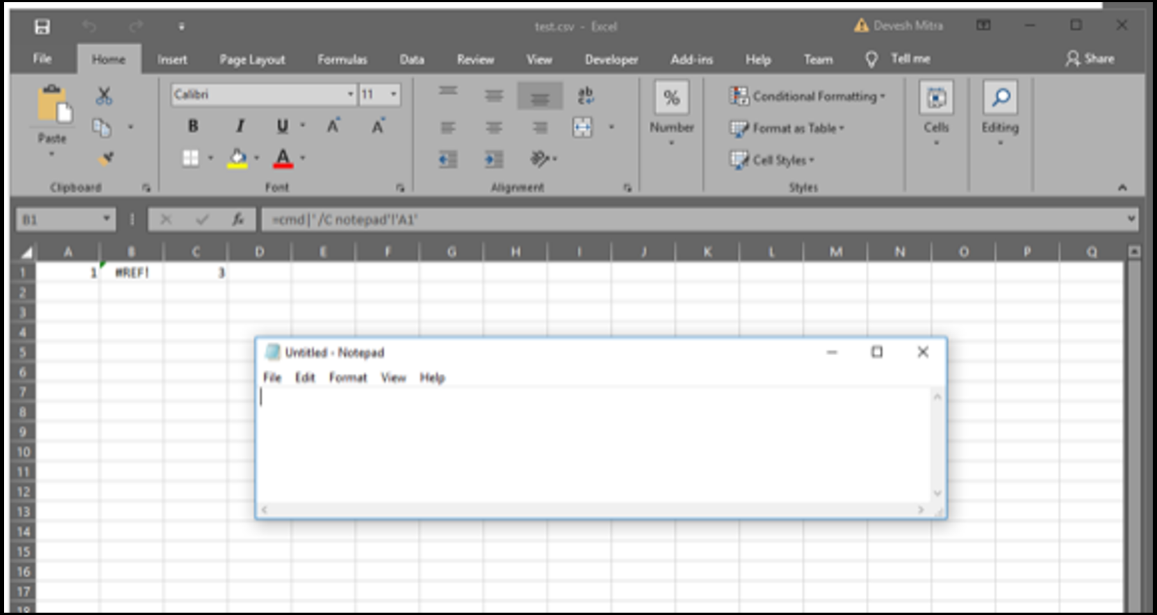
Microsoft Excel Security Notice

Microsoft Office has identified a potential security concern.

File Path: C:\Users\Bob\Documents\test.csv

Automatic update of links has been disabled. If you choose to enable automatic update of links, your computer may no longer be secure. Do not enable this content unless you trust the source of this file.

and… .



```
=cmd|' /C notepad'!'A1'
```

	A	B	C
1	1	#REF!	3

Remediations

Application exporting CSV files must sanitise the output!
The following characters are known to be dangerous:

= + - @

- Cells beginning with these characters should have a single quote character (') inserted at the beginning.
- This forces Excel to interpret the cell as text.
- Make sure commas are removed from data!
- Commas can be used to start a new cell, which then evades the single quote remediation above.
- If a different delimiter other than commas is used, modify the remediation accordingly.

THANKS FOR LISTENING TO US RANT!

questions? slack / email

Thanks to @sy for all the contributions