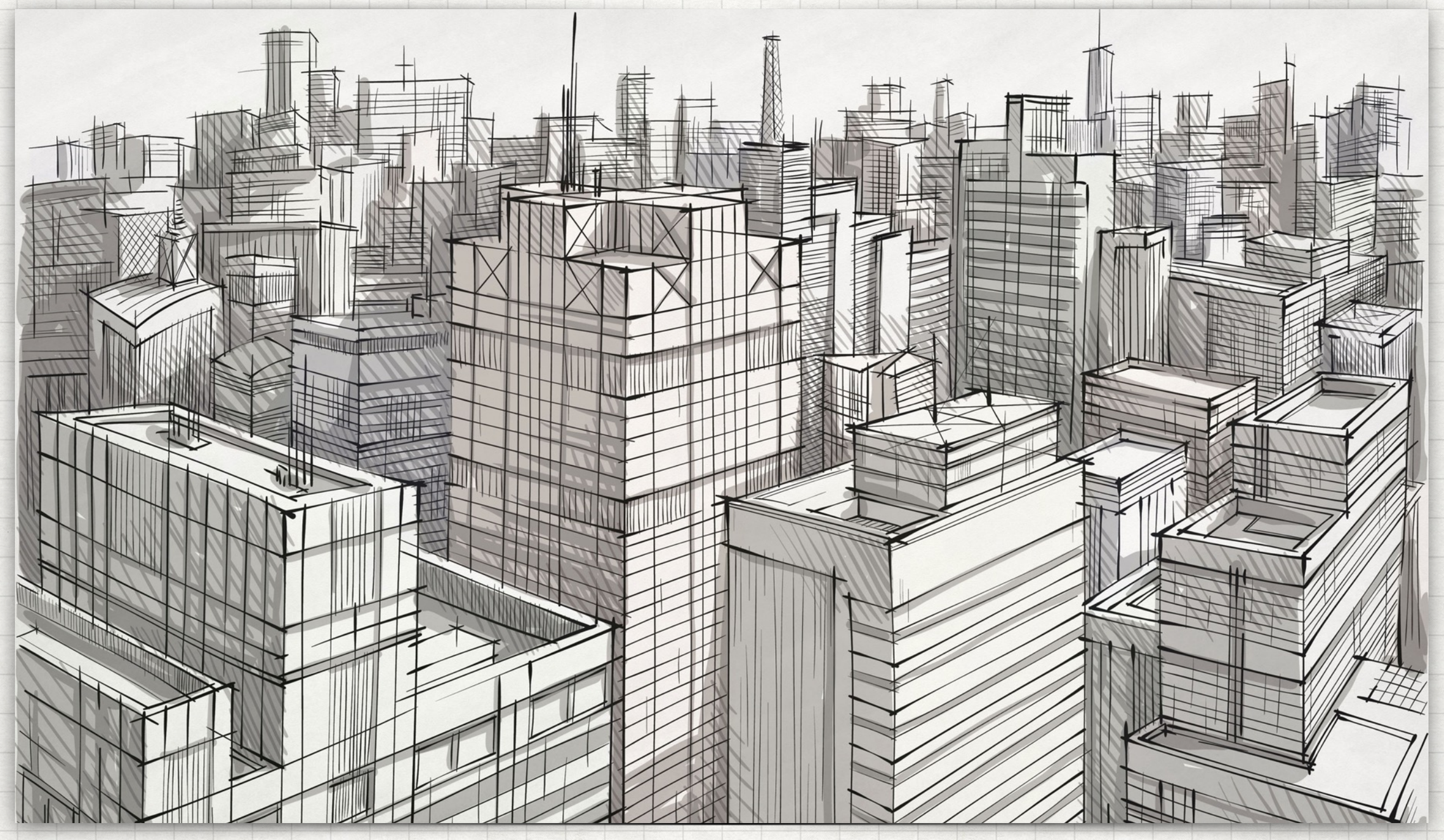
SECURING LEGACY CFML

PETE FREITAG, FOUNDEO INC.



foundeo

ABOUT PETE

- My Company: Foundeo Inc.
    - **Consulting**: Code Reviews, Server Reviews, Development
    - **FuseGuard**: Web App Firewall for CFML
    - **HackMyCF**: Server Security Scanner
- Blog (petefreitag.com), Twitter (@pfreitag), #CFML Slack
- Guy behind cfdocs.org community sourced CFML docs.

AGENDA

- Legacy Code Challenges
- How do you get started?
    - Low Hanging Fruit
    - Things to focus on
- What's Next?
- Disclaimer: This approach may not be appropriate for all scenarios. This is a generalized approach which I have found can work well for many.

DO YOU HAVE TO WORK WITH LEGACY CODE?

LEGACY CODE

TYPICALLY

- Has a large codebase (thousands of source code files)
- Has code you hope you don't have to see again.
- Can take weeks, but often months of work to properly secure.
- Can be hard to fix, brittle
- Probably uses outdated techniques

HOW TO APPROACH

FIXING A LARGE CODEBASE

- **Beast Mode** - Spend several weeks dedicated to identifying & fixing vulnerabilities.
- **Prioritize** - Spend time identifying the most critical vulnerabilities and patch less critical vulnerabilities as you see them.
- **As you go** - As you work on files fix vulnerabilities as you see them. You may not ever fix some vulnerabilities with this approach.

HOW DO YOU START?

SECURING THAT LEGACY CODE

STEP 1: DELETE THE CODE!

LEGACY CODEBASES ARE LARGE BUT…

MUCH OF THE CODE PROBABLY NEVER RUNS

YOU MIGHT BE USING...

HOMEMADE VERSION CONTROL

- index_2.cfm
- index.old.cfm
- index-backup.cfm
- index-2007-03-04.cfm
- index-copy.cfm
- folder_backup2009/

VERSION CONTROL

- Those backup folders and files are probably full of vulnerabilities.
- Version Control Server keeps backups of all your code and all changes you have ever made to it.
- Sync server source code with version control.
    - Identify if someone changed something on the server.

VERSION CONTROL

IDENTIFY UNUSED CODE

- Spend some time to identify unused code.
- Delete it!
- Version control has your back, if you deleted something you can recover it from the repository.

“

THERE ARE LOTS OF FADS IN SOFTWARE DEVELOPMENT, VERSION CONTROL IS NOT ONE OF THEM.

”

PATCH THAT SERVER

WELCOME TO THE 90'S

- Use ColdFusion 10 or greater (CF9 and below are no longer supported and no longer patched by Adobe).
- Railo has not been touched since 2014, use Lucee (it is very easy to switch).
- Windows 2008 (EOL 2015)
- Java 8+, Java 7 (EOL 2015), Java 6 (EOL 2013)

PATCH THAT SERVER

FIX VULNERABILITIES

- Multiple Denial of Service Vulnerabilities in old versions of Java
- Path Traversal via Null Byte injection JVM
- CRLF Injection (CF10+)
- File Uploads “somewhat” more secure (CF10+)
- TLS / SSL Protocol Implementations
- Java 8 Not supported on CF9 and below

LOCKDOWN THE SERVER

MITIGATES POTENTIAL IMPACT OF A VULNERABILITY

- If your CFML server is running as SYSTEM or root then the attacker can do a lot more harm.
- If CFML server user has read only access to web root.

IMPLEMENT A WAF

WEB APPLICATION FIREWALLS

- Inspect HTTP Request or Response
    - Block or log malicious requests
- Several options
    - Hardware
    - Web Server Level - ModSecurity
    - Application Level - FuseGuard

HOW DO YOU START

SECURING THAT LEGACY CFML?

STEP 2: IDENTIFY HIGH RISK VULNERABILITIES IN YOUR CODE.

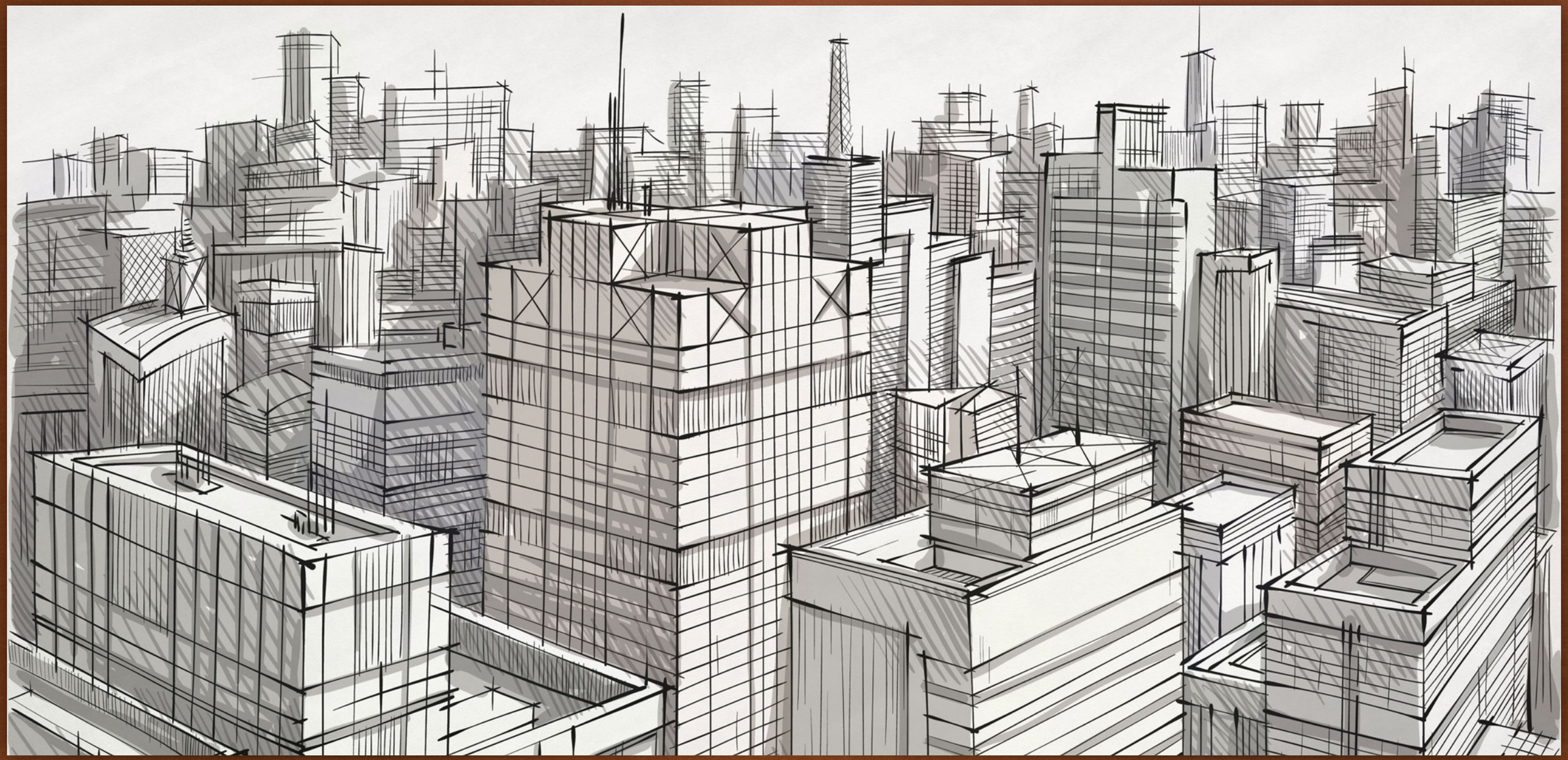
HIGH RISK VULNERABILITIES

TAKE CARE OF THESE FIRST

- File Uploads
- Dynamic Evaluation Issues
- SQL Queries (SQL Injection)
- File System Access / Path Traversals
- Dynamic Process Execution (CFEXECUTE)
- Anything that can fully compromise server

REMOTE CODE EXECUTION VIA

EVALUATE



COMMON LEGACY EVALUATE

CODE EXAMPLE

```
<cfset day_1 = "Wednesday">
<cfset day_2 = "Thursday">
<cfset day_3 = "Friday">

<cfoutput>
  #Evaluate("day_#url.day#")#
</cfoutput>
```

EVALUATE

EXAMPLE

FIXING LEGACY EVALUATE EXAMPLE

USE BRACKET NOTATION

```
<cfset day_1 = "Wednesday">
<cfset day_2 = "Thursday">
<cfset day_3 = "Friday">

<cfoutput>
  #variables["day_#url.day#"]#
</cfoutput>
```

FIXING EVALUATE ISSUES

SEARCH CODE FOR EVALUATE

- Search Code for "Evaluate"
- In most cases you should not need to use Evaluate at all, use brackets.
    - If the variable is a query you may need to use queryName[row][columnName] notation.
    - Not all cases are super simple to fix, but most are.
- Remove all Evaluate calls from your code.

DO ANY OTHER FUNCTIONS EVALUATE DYNAMICALLY?

IIF

IF YOU ARE USING IIF STOP USING IIF

```
Hi #iif(len(url.name) EQ 0, de("Friend"), de(url.name))#
```

The second and third arguments are evaluated dynamically!

IIF EXAMPLE

FIXING IIF

USE TERNARY OPERATOR (CF9+, LUCEE)

```
Hi #(!len(url.name)) ? "Friend" : url.name#
```

ELVIS OPERATOR (CF11+, LUCEE)

```
Hi #url.name?:"Friend"#
```

Elvis Operator tests to see if url.name is defined / not null

DO ANY OTHER FUNCTIONS EVALUATE DYNAMICALLY?

YES!

The **PrecisionEvaluate** function also dynamically evaluates expressions

DO ANY OTHER FUNCTIONS EVALUATE DYNAMICALLY?

YES!

Lucee 5 has added a **render** function that evaluates tags dynamically.

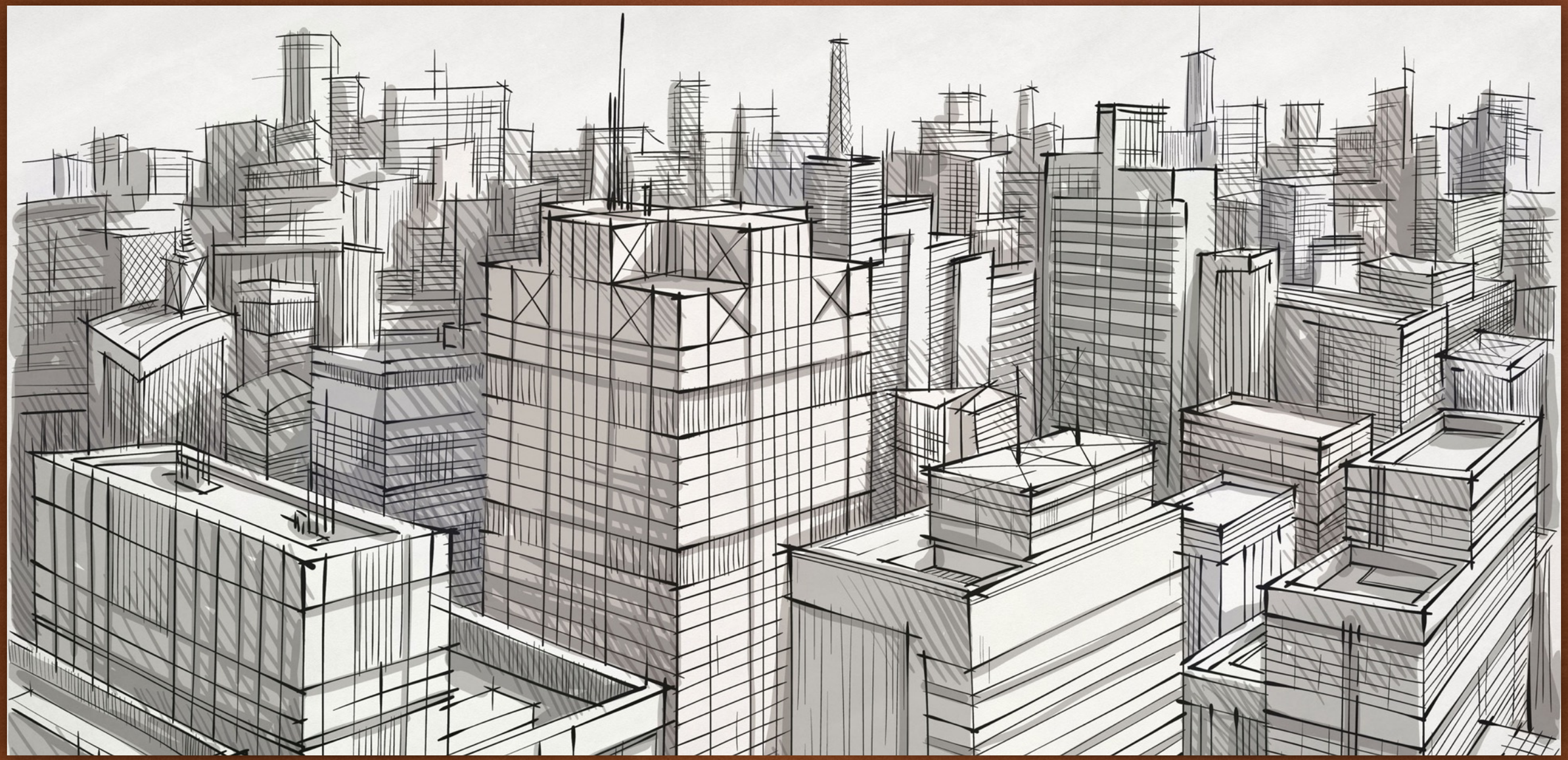
DO ANY OTHER FUNCTIONS EVALUATE DYNAMICALLY?

NO!

Not that I know of

COMMON YET DANGEROUS

FILE UPLOADS



FILE UPLOAD

EXAMPLE

FILE UPLOADS

3 RULES

- The upload **destination** must be outside of the web root
- Always validate the **file extension** against a whitelist
- Don't trust mime type validation in the **accept** attribute

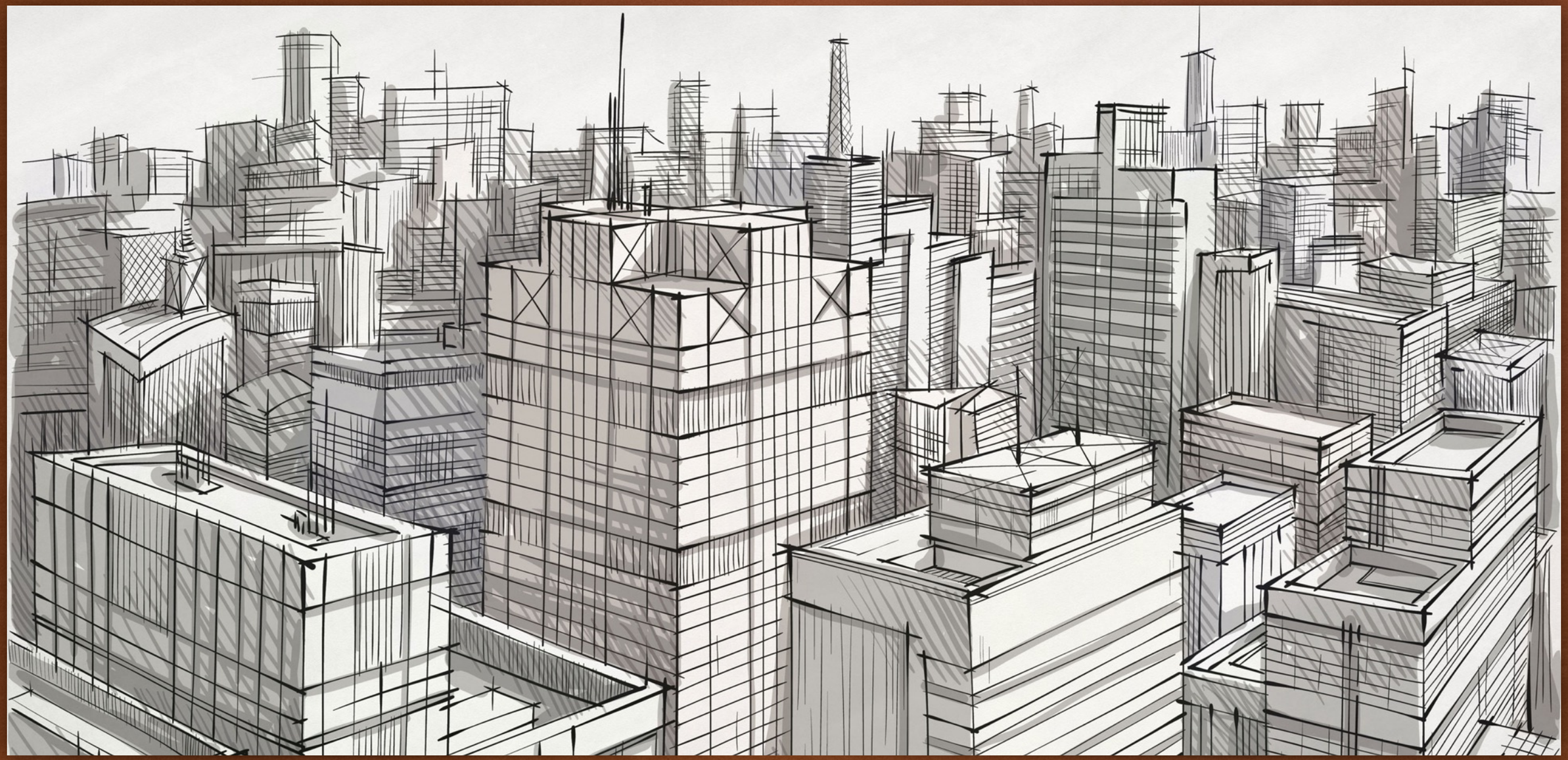
FILE UPLOADS

ADDITIONAL TIPS

- Inspect file content: fileGetMimeType, isImageFile, isPDFFile, etc
- Upload to static content server (s3 for example)
  - Upload directly to s3: https://www.petefreitag.com/item/833.cfm
- Make sure directory serving uploaded files cannot serve dynamic content.
- File Extension Whitelist on Web Server (eg IIS Request Filtering)
- secureupload.cfc: https://github.com/foundeo/cfml-security/

FILE SYSTEM ACCESS &

PATH TRAVERSAL



PATH TRAVERSAL

VULNERABLE CODE EXAMPLE

```
<cfinclude template="path/#fileName#">
```

PATH TRAVERSAL

EXAMPLE

FIXING PATH TRAVERSALS

TIPS

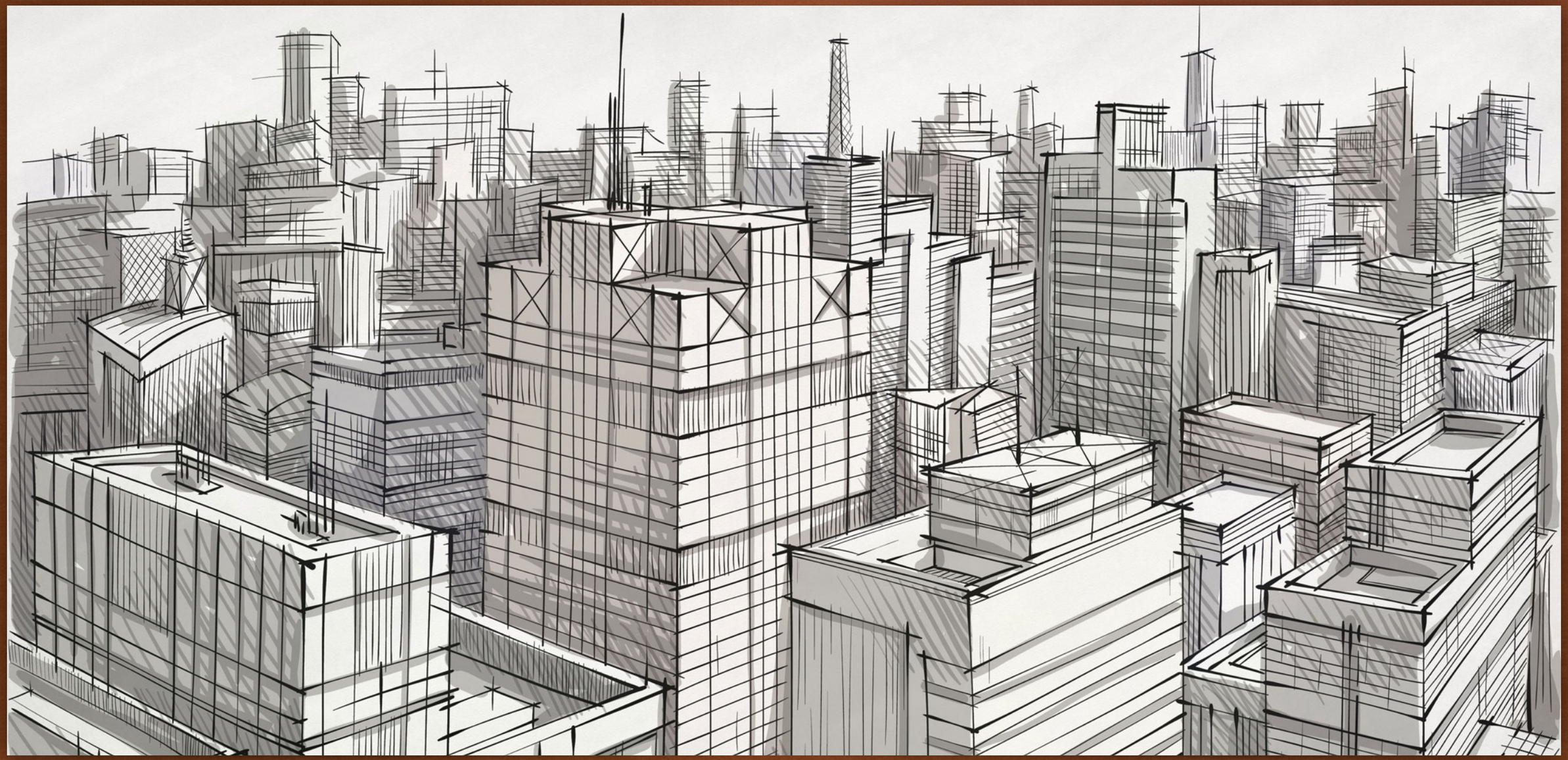
- Avoid variables in paths
    - If you really need to use a variable strip out everything except a-z0-9
- Use the CF11 Application.cfc setting this.compileExtForInclude setting.

FINDING FILE ACCESS ISSUES

CAN BE TIME CONSUMING

- Review all function calls / tags that access file system
  - cffile, cfdocument, cfinclude, cfmodule, cfspreadsheet
  - fileRead, fileWrite, fileOpen, etc

SQL INJECTION



CLASSIC SQL INJECTION

CODE EXAMPLE

```
<cfquery>
  SELECT title, story
  FROM news
  WHERE id = #url.id#
</cfquery>
```

FIXING SQL INJECTION

CODE EXAMPLE

```
<cfquery>
  SELECT title, story
  FROM news
  WHERE id = <cfqueryparam value="#url.id#">
</cfquery>
```

SCRIPT BASED

SQL INJECTION

```
queryExecute("SELECT story FROM news WHERE id = #url.id#");
```

Vulnerable

```
queryExecute("SELECT story FROM news WHERE id = :id", {id=url.id});
```

Not Vulnerable

FINDING SQL INJECTION

DONEC QUIS NUNC

- Search codebase for cfquery, queryExecute, ormExecute query
- Use Static Code Analyzer (CFBuilder 2016)
- Fix when you see one as you work

SECURING LEGACY CFML

STEP 3: FIX ADDITIONAL VULNERABILITIES IN YOUR CODE.

WHAT'S NEXT

TO REVIEW

- Session Handling (sessionRotate, sessionInvalidate)
- Scope Injection
- Authentication / Authorization / Forgot / Remember Me Code
- Cross Site Scripting
- Cross Site Request Forgery
- Timing Attacks
- Visit OWASP.org for tons of info about web application vulnerabilities

THANK YOU

Questions?

Pete Freitag
pete@foundeo.com

foundeo

foundeo.com | fuseguard.com | hackmycf.com