ACC Association of Corporate Counsel

SOUTHERN CALIFORNIA

ACC SOUTHERN CALIFORNIA
IN HOUSE COUNSEL CONFERENCE

January 30, 2024
Anaheim, California

sponsored by:

Key Risks of Using Open Source

- Patent Infringement Liability
- Security Risk
- Copyright Infringement Risk

Patent Infringement Liability Risk

Begins with Understanding What Open Source Software Is

Open Source Software =

1. Source code that is freely distributed
2. Under a license from the author that
   a. permits derivative works to be created
   b. requires that credit be given to the author
   c. does not discriminate against persons/groups/fields of endeavor
   d. must not be specific to a product
   e. must not restrict other software
   f. is technology neutral

Open Source Initiative

So you can use the code however you want without recrimination from the author/provider of the code, but that doesn't mean you can use it without risk with respect to other parties

Patent Infringement Liability Risk

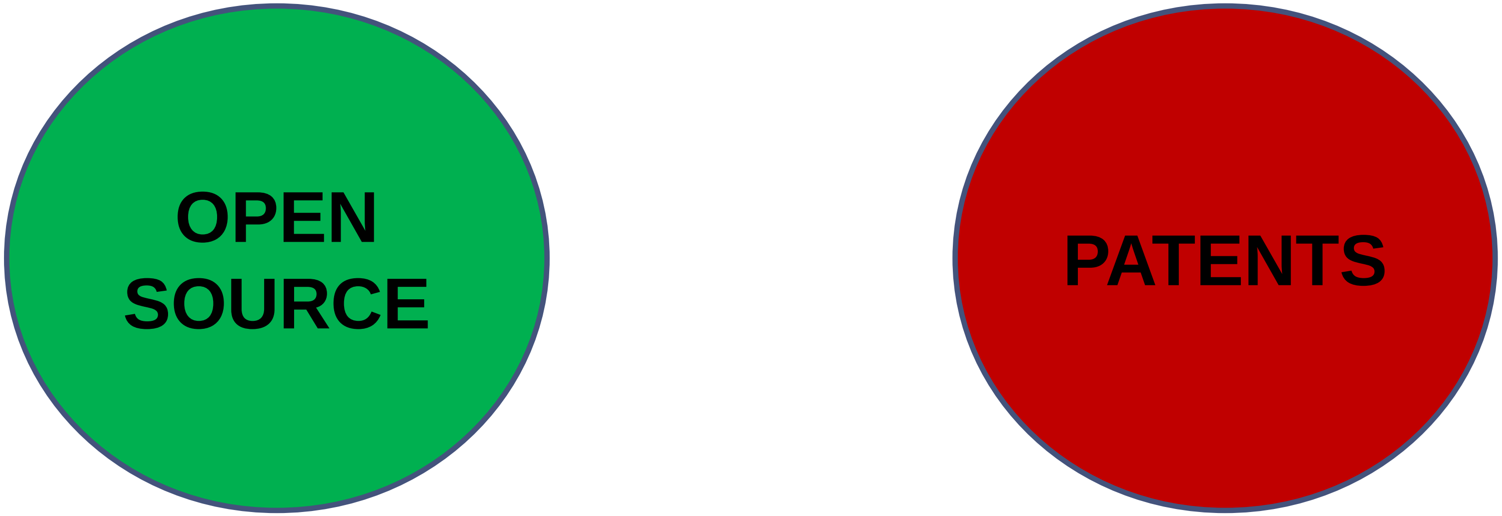
And Continues with an Understanding of What a Patent Is

- A patent for an invention is the grant of a property right to the inventor.
- The property right conferred is *the right to exclude others* from using, making, or selling anything within the scope of the claims of the patent.
- Thus, a patent does NOT give the owner/inventor the right to practice the invention themselves.

Patent Infringement Liability Risk

Patents and Open Source Software Are NOT Mutually Exclusive

- a patent can give a third party owner the right to exclude you from using open source software; and
- the open source license will NOT give you a license under that third party patent



OPEN SOURCE

PATENTS

Patent Infringement Liability Risk

So how do you mitigate that risk?

- **Consider using non-open source alternatives**
  - Vet the alternatives carefully and consider cost of license or development versus potential cost savings of avoiding patent litigation
- **Consider obtaining the open source software from a vendor, e.g., through PaaS, SaaS, or IaaS, rather than implementing the open source software yourself**
  - Vet the vendors carefully and consider the cost of the service versus potential cost savings of avoiding patent litigation
  - Make sure that all service agreements with vendors include sufficient reps and warranties, as well as indemnification provisions **THAT DO NOT CARVE OUT OPEN SOURCE SOFTWARE OR PROVIDE A LOW CAP ON THE INDEMNIFICATION AMOUNT**
  - Make sure that the vendor is financially able to live up to its indemnification obligations

Patent Infringement Liability Risk

Example Contractual Provisions both Good and Bad

"Service Provider will indemnify, defend, and hold harmless Customer, its officers, directors, employees, agents, and other affiliates from and against all damages, costs, penalties, liabilities, or expenses (including reasonable attorneys' fees and costs) directly or indirectly arising out of or related to any third-party claim, demand or lawsuit (collectively a "Claim") based on an allegation that the Service Provider Technology or any other intellectual property furnished or used by Service Provider in connection with this Agreement infringes any copyright, trade secret, patent or trademark of any third party or violates applicable laws."

Patent Infringement Liability Risk

Example Contractual Provisions both Good and Bad

"The indemnity provided by Service Provider under this Agreement does not extend to Claims arising from or relating to Third Party Software."

"IN NO EVENT WILL SERVICE PROVIDER'S AGGREGATE LIABILITY UNDER THIS AGREEMENT EXCEED THE GREATER OF $1,000,000.00 OR TWO TIMES THE AMOUNT OF FEES PAID BY LICENSEE TO SERVICE PROVIDER PURSUANT TO THIS AGREEMENT."

Security Risk

There are large numbers of Open Source components being used

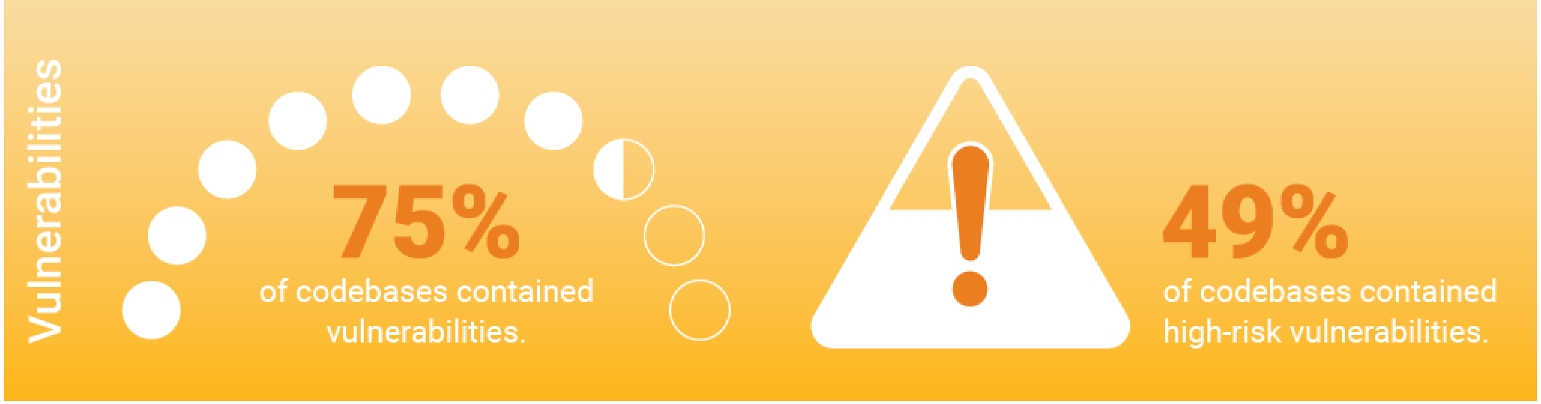
"Black Duck Audits found open source in **nearly 99%** of codebases audited in 2019."

"Black Duck Audits identified an average of **445 open source components** per codebase in 2019. . . . "

https://www.synopsys.com/software-integrity/resources/analyst-reports/2020-open-source-security-risk-analysis.html

Security Risk

- There are many vulnerabilities, some of which are high risk



Vulnerabilities

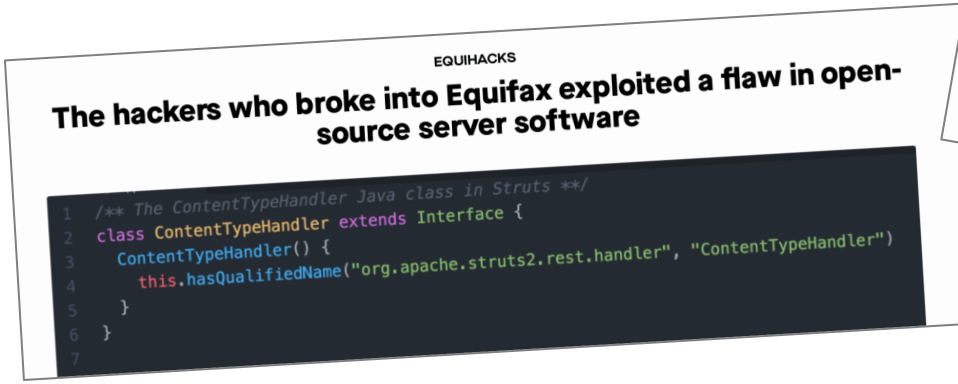
75% of codebases contained vulnerabilities.

49% of codebases contained high-risk vulnerabilities.

https://www.synopsys.com/software-integrity/resources/analyst-reports/2020-open-source-security-risk-analysis.html

Security Risk

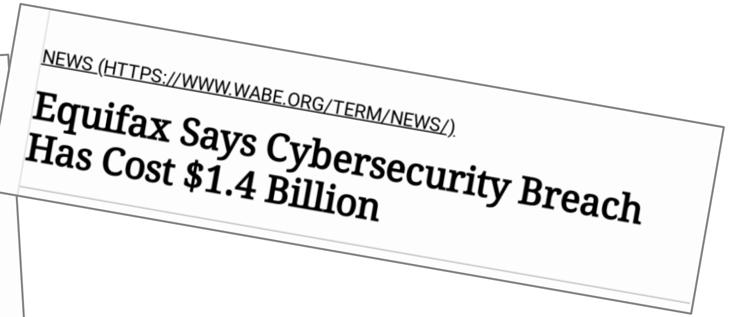
- The security risk is compounded when it intersects with privacy rights



EQUIHACKS

The hackers who broke into Equifax exploited a flaw in open-source server software

```
/** The ContentTypeHandler Java class in Struts **/
class ContentTypeHandler extends Interface {
  ContentTypeHandler() {
    this.hasQualifiedName("org.apache.struts2.rest.handler", "ContentTypeHandler")
  }
}
```



NEWS (HTTPS://WWW.WABE.ORG/TERM/NEWS/)

Equifax Says Cybersecurity Breach Has Cost $1.4 Billion

https://qz.com/1073221/the-hackers-who-broke-into-equifax-exploited-a-nine-year-old-security-flaw/#:~:text=That%20vulnerability%2C%20according%20to%20a,same%20day%20it%20was%20announced
https://www.wabe.org/equifax-says-cybersecurity-breach-has-cost-1-4-billion/

Security Risk

There are several known Open Source security risks

- **Notice files provide hackers with a list components that are being used**

OpenSSL Heartbleed Bug

Linux GNU C Lib Ghost

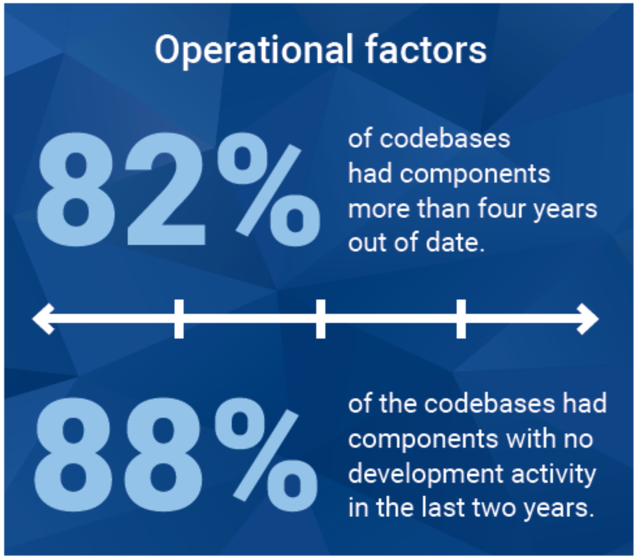
GNU Bash Shellshock

Fake jQuery Injections

Security Risk

Challenges of Managing Security Risk

1. Need to proactively look for Open Source updates
   - Updates have to be “pulled”
     - No commercial vendor proactively reaching out with warning messages
     - No automatic installation
   - Should also flag stale components where there is no one providing updates



Operational factors

82% of codebases had components more than four years out of date.

88% of the codebases had components with no development activity in the last two years.

https://www.synopsys.com/software-integrity/resources/analyst-reports/2020-open-source-security-risk-analysis.html

Security Risk

Challenges of Managing Security Risk

2. There is often a time lag in the security databases

    - Vulnerabilities are not published in the National Vulnerabilities Database right away
    - “[S]ome research reporting an average 27 days between initial announcement and NVD publication”*
    - Black Duck’s Security Advisories attempt to provide earlier notification.

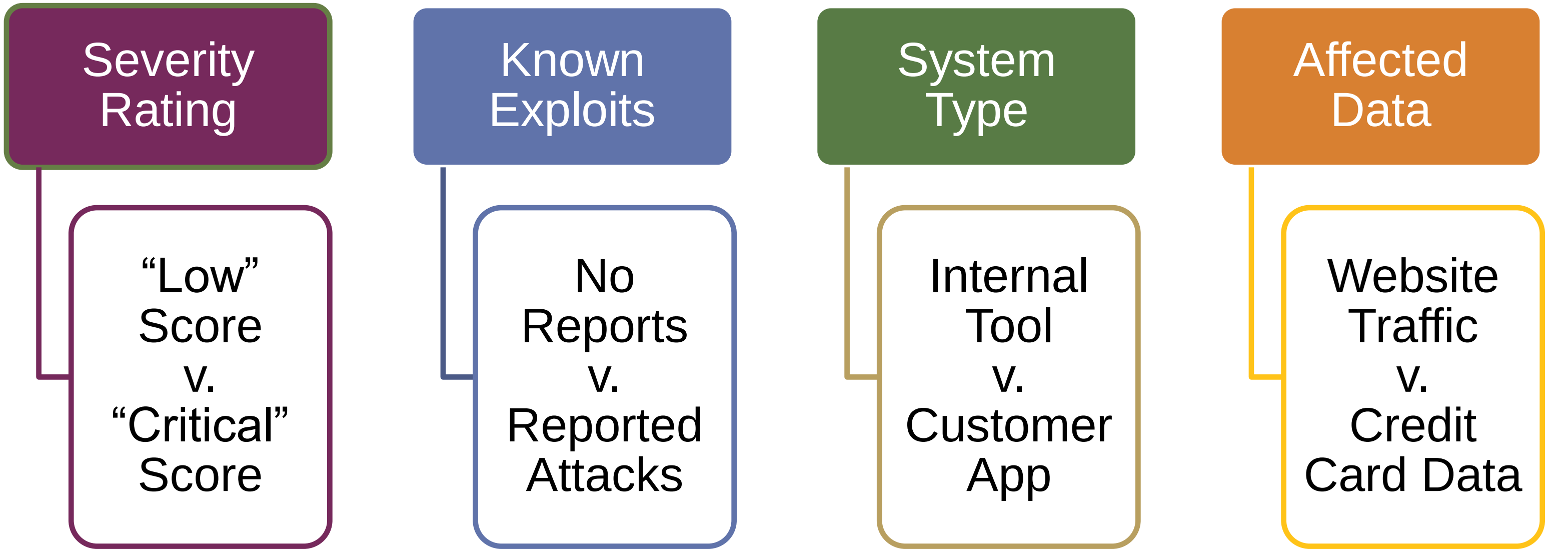
Security Risk

Challenges of Managing Security Risk

3. Must have a framework for prioritizing patches
   - There can be hundreds of updates that need to be installed
   - Some updates can have a significant effect on how other components work (or don’t work)
   - Teams should be careful about which Open Source components to include in the codebase

Security Risk

Example Prioritization Factors



Severity Rating

"Low" Score v. "Critical" Score

Known Exploits

No Reports v. Reported Attacks

System Type

Internal Tool v. Customer App

Affected Data

Website Traffic v. Credit Card Data

Security Risk

Open Source requires proactive monitoring and management

Review Black Duck reports

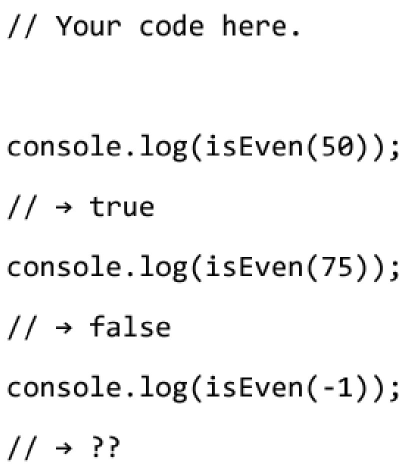
Stay on top of security updates

Install important patches

Track stale components

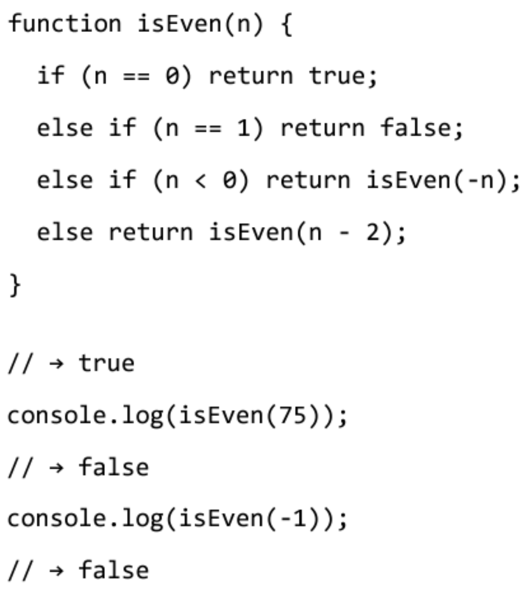
Copyright Infringement Risk for AI Generated Code

Doe v. GitHub, Inc., No. 4:22-cv-06823-JST (N.D. Cal.) (filed 3 Nov 2022)



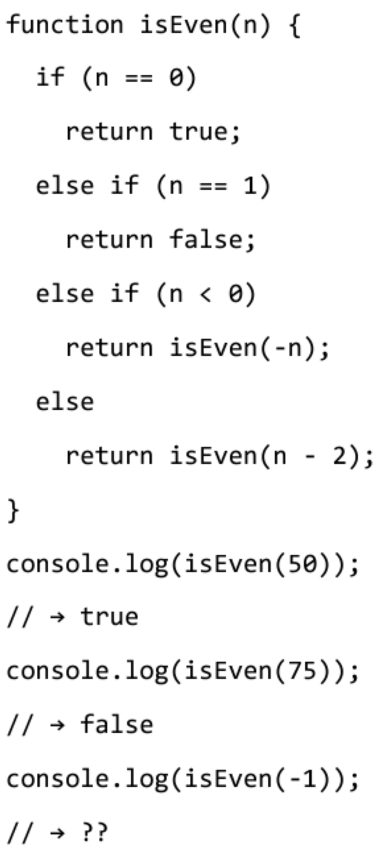
```
// Your code here.

console.log(isEven(50));
// → true
console.log(isEven(75));
// → false
console.log(isEven(-1));
// → ??
```

+



```
function isEven(n) {
  if (n == 0) return true;
  else if (n == 1) return false;
  else if (n < 0) return isEven(-n);
  else return isEven(n - 2);
}
// → true
console.log(isEven(75));
// → false
console.log(isEven(-1));
// → false
```

→



```
function isEven(n) {
  if (n == 0)
    return true;
  else if (n == 1)
    return false;
  else if (n < 0)
    return isEven(-n);
  else
    return isEven(n - 2);
}
console.log(isEven(50));
// → true
console.log(isEven(75));
// → false
console.log(isEven(-1));
// → ??
```

Examples from *Eloquent Javascript*

Generated by Copilot

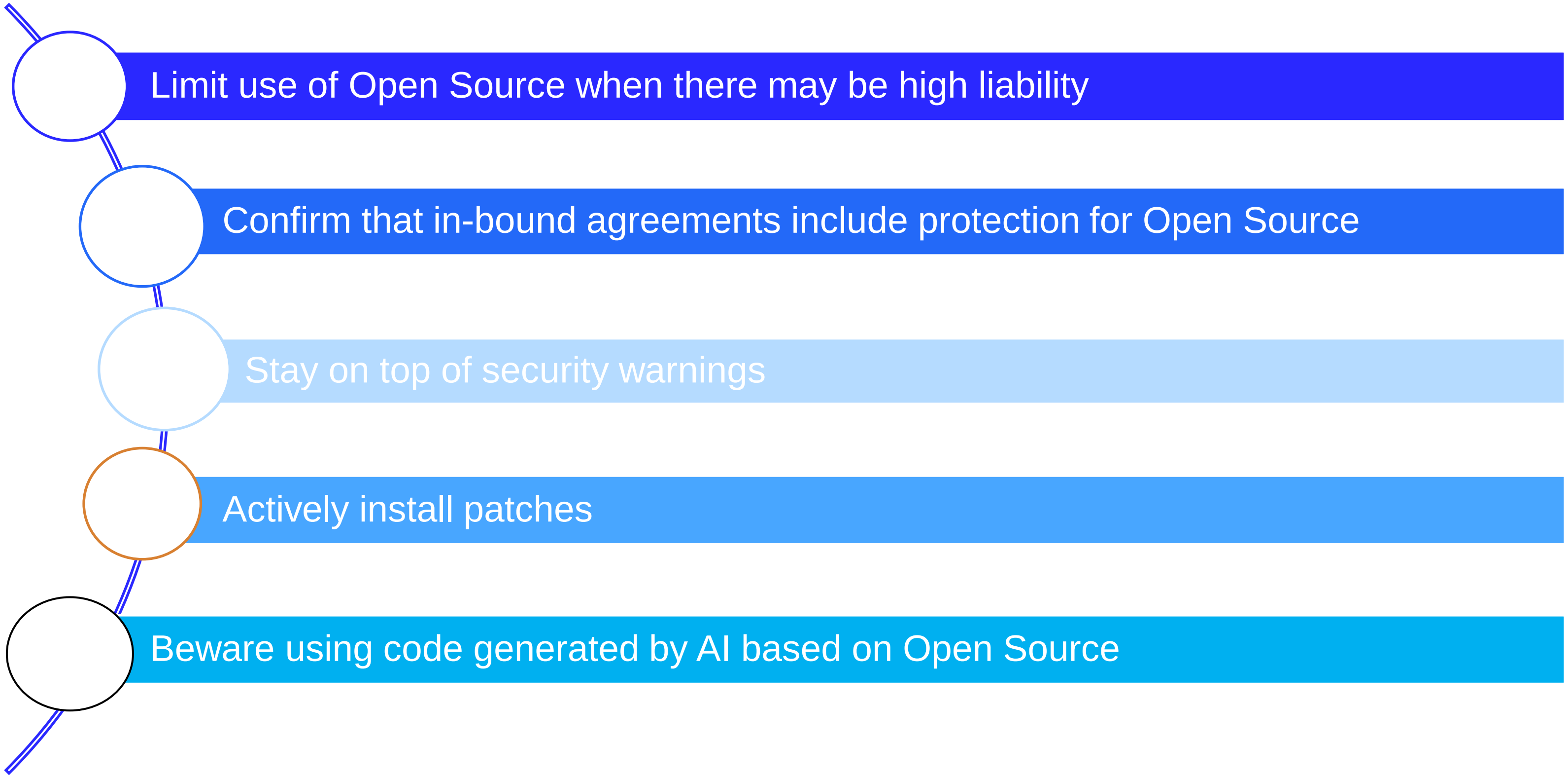
Copyright Infringement Risk for AI Generated Code

- The law here is very unsettled because it doesn't contemplate the scale of the problem
- The problem Generative AI poses
  - Most image generative AI engines are trained on vast data sets
  - These data sets can contain *unlicensed* content
  - If only *licensed* content is used, there may not be a sufficient volume of input to effectively train the generative AI (at least with the current state of the technology)
  - Thus, if effectively trained generative AI is desired, it may be necessary to use unlicensed content for the time being
  - It is not currently clear whether the use of unlicensed content to train a generative AI engine will be considered a fair use
  - This makes the risk of copyright infringement inherent and unavoidable

Copyright Infringement Risk for AI Generated Code

- **So how does a company mitigate its risk?**
  - **Avoid using generative AI to generate code until the landscape clears**
  - **Use generative AI engines from known vendors who will provide indemnification**
  - **For example, consider Microsoft's Customer Copyright Commitment:**
    - Microsoft will **defend** customers of commercial Copilot offerings against third-party IP claims for the use of the services.
    - Microsoft will **indemnify** the amount of the claim, including any resulting adverse judgment or approved settlement.
- **But read the fine print!**

Key Takeaways



- Limit use of Open Source when there may be high liability
- Confirm that in-bound agreements include protection for Open Source
- Stay on top of security warnings
- Actively install patches
- Beware using code generated by AI based on Open Source

Q&A

Thank You

Maria Anderson
Partner
Pronouns: she/her/hers
Maria.Anderson@knobbe.com
206-405-2003 Direct
Knobbe Martens
925 Fourth Ave., Suite 2500
Seattle, WA 98104
www.knobbe.com/maria-anderson