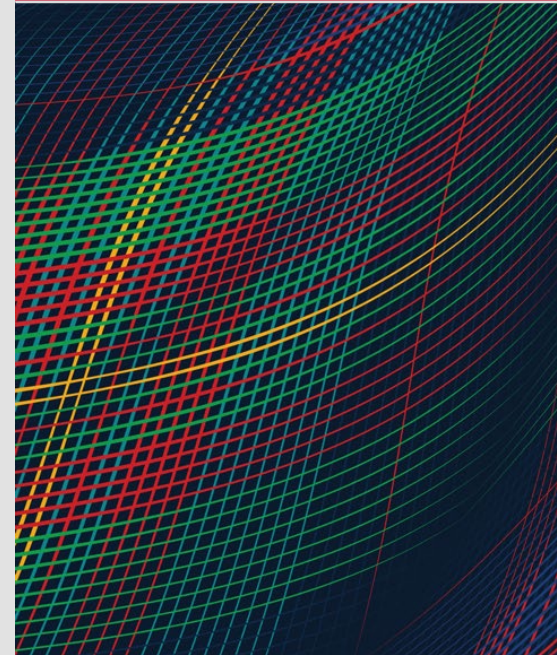


Microservice and API Risks and Mitigations

AUGUST 7, 2024

McKinley Sconiers-Hasan
CERT Associate Solutions Engineer



Document Markings

Copyright 2024 Carnegie Mellon University.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

CERT® and Carnegie Mellon® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.
DM24-0993

Agenda

- Introduction to APIs
- Microservice architectures
- 3 common risks of modern APIs
- Authorization vulnerabilities
- Final recommendations

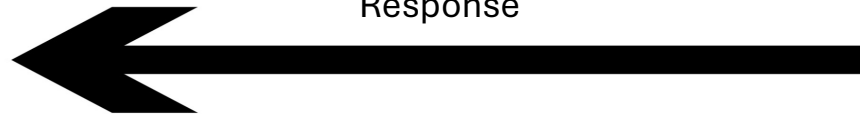
Introduction to APIs

Introduction to APIs

An API is a connector between two applications



Request



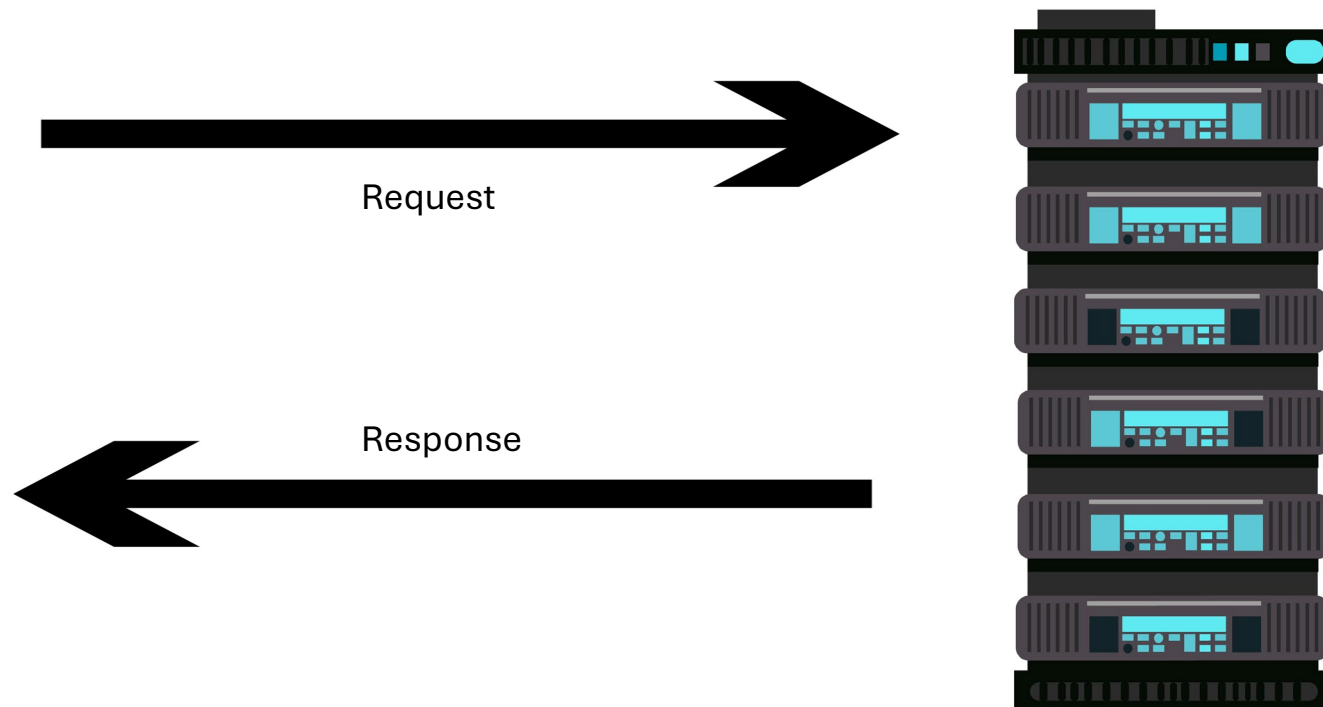
Response



Introduction to APIs



`https://www.streaming-service.com/get-song?song='Eye of the Tiger'`



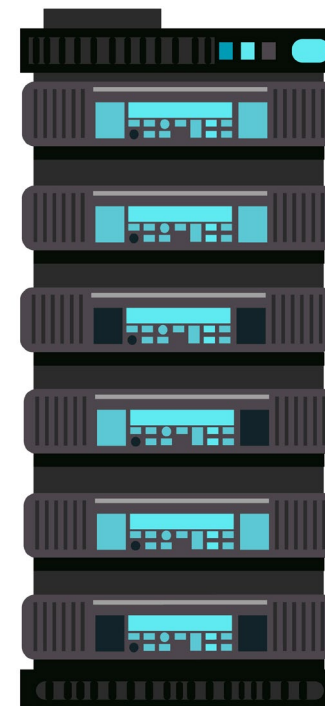
Introduction to APIs



<https://www.streaming-service.com/get-song?song='Eye of the Tiger'>



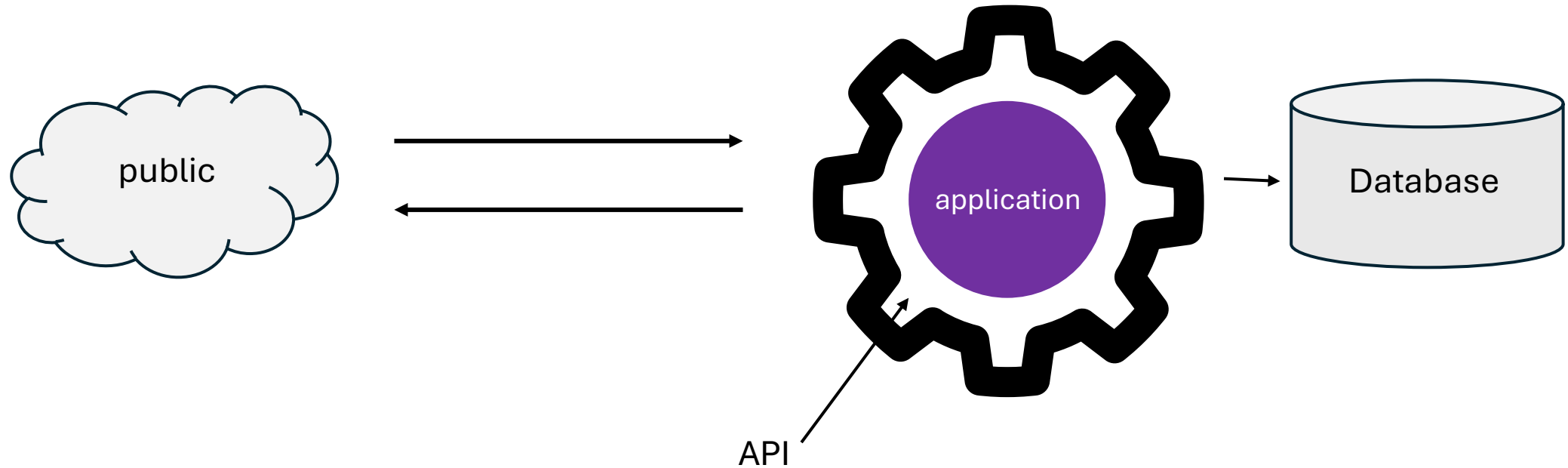
Request



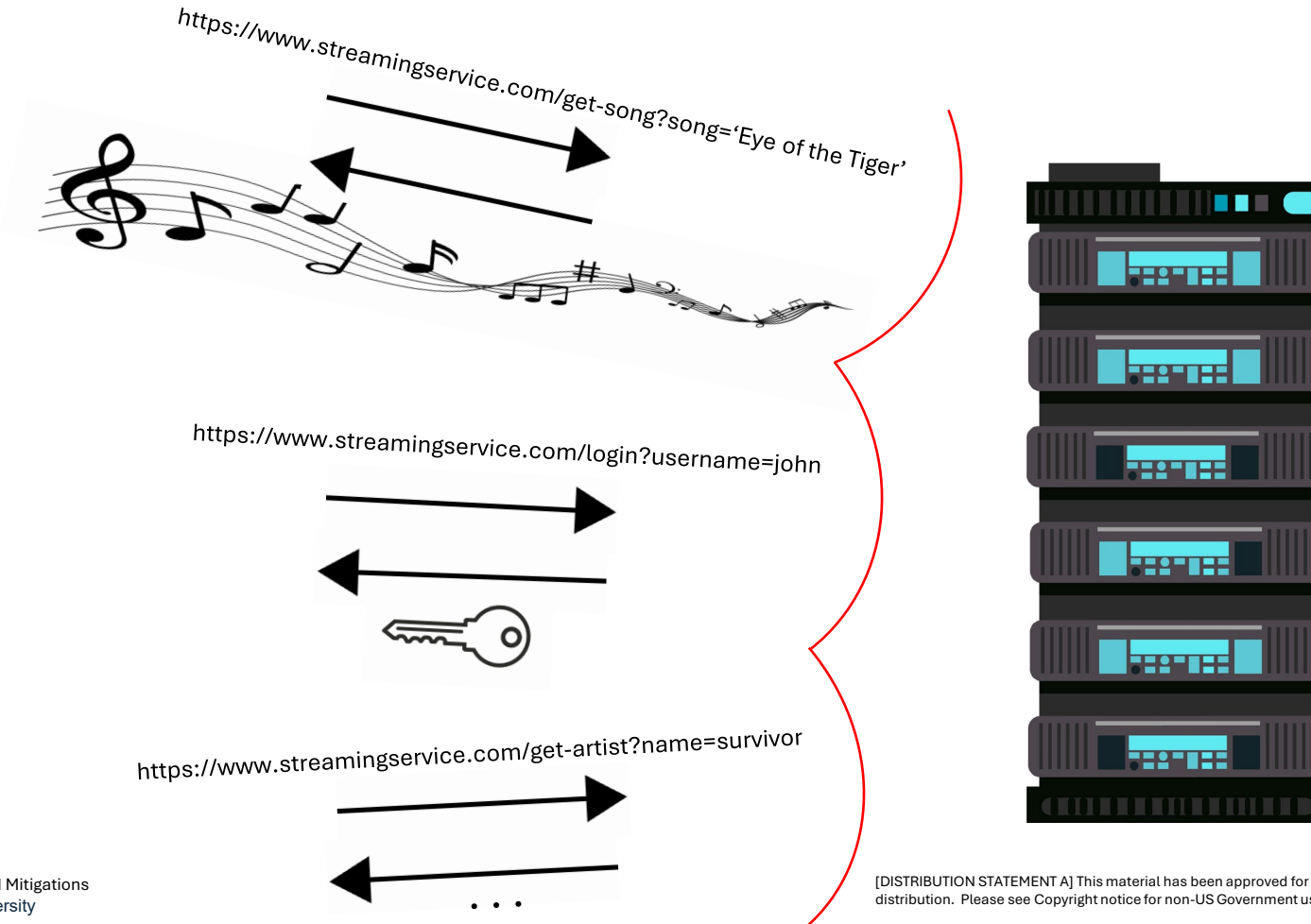
Response



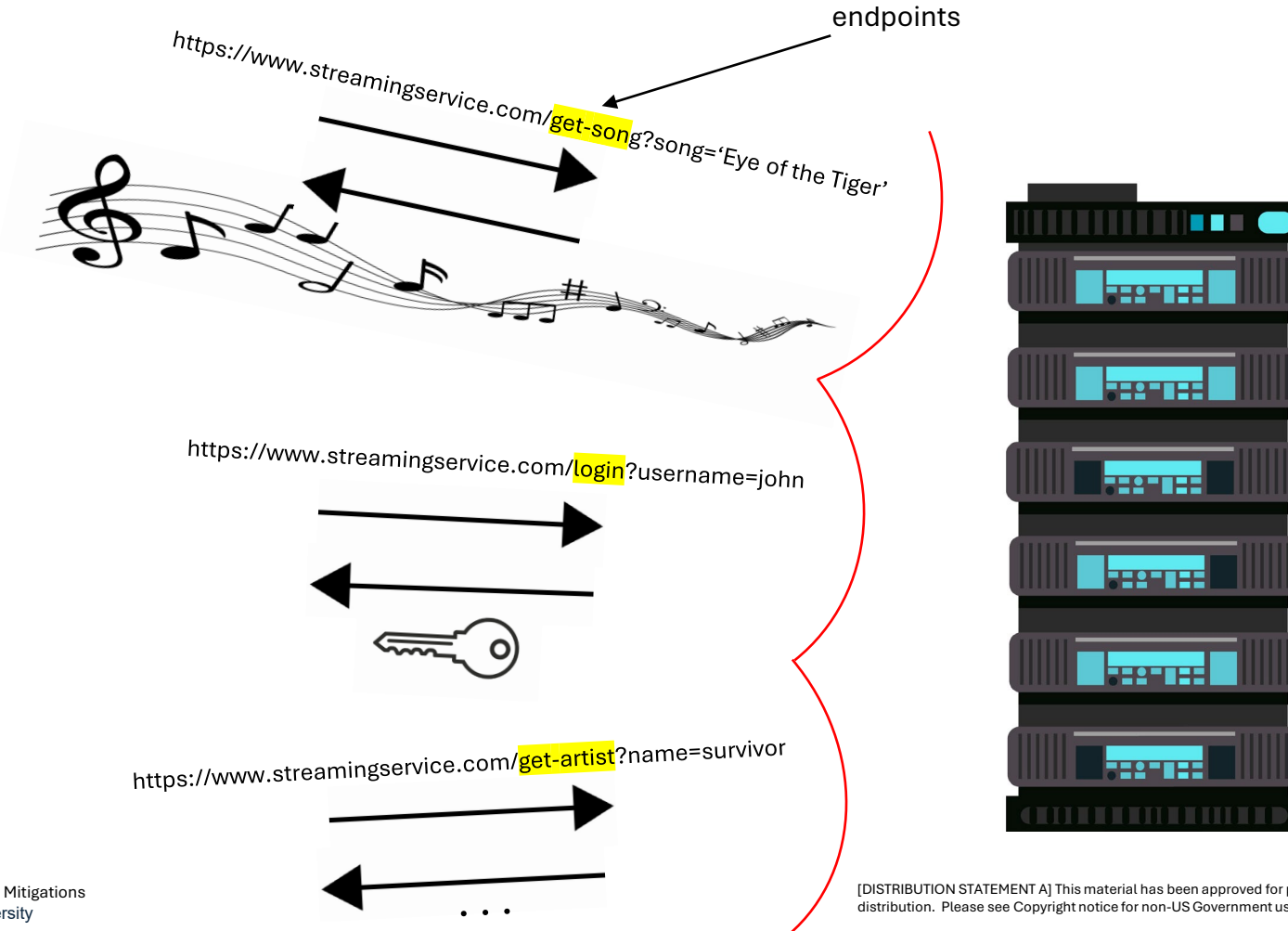
APIs are Important for Security



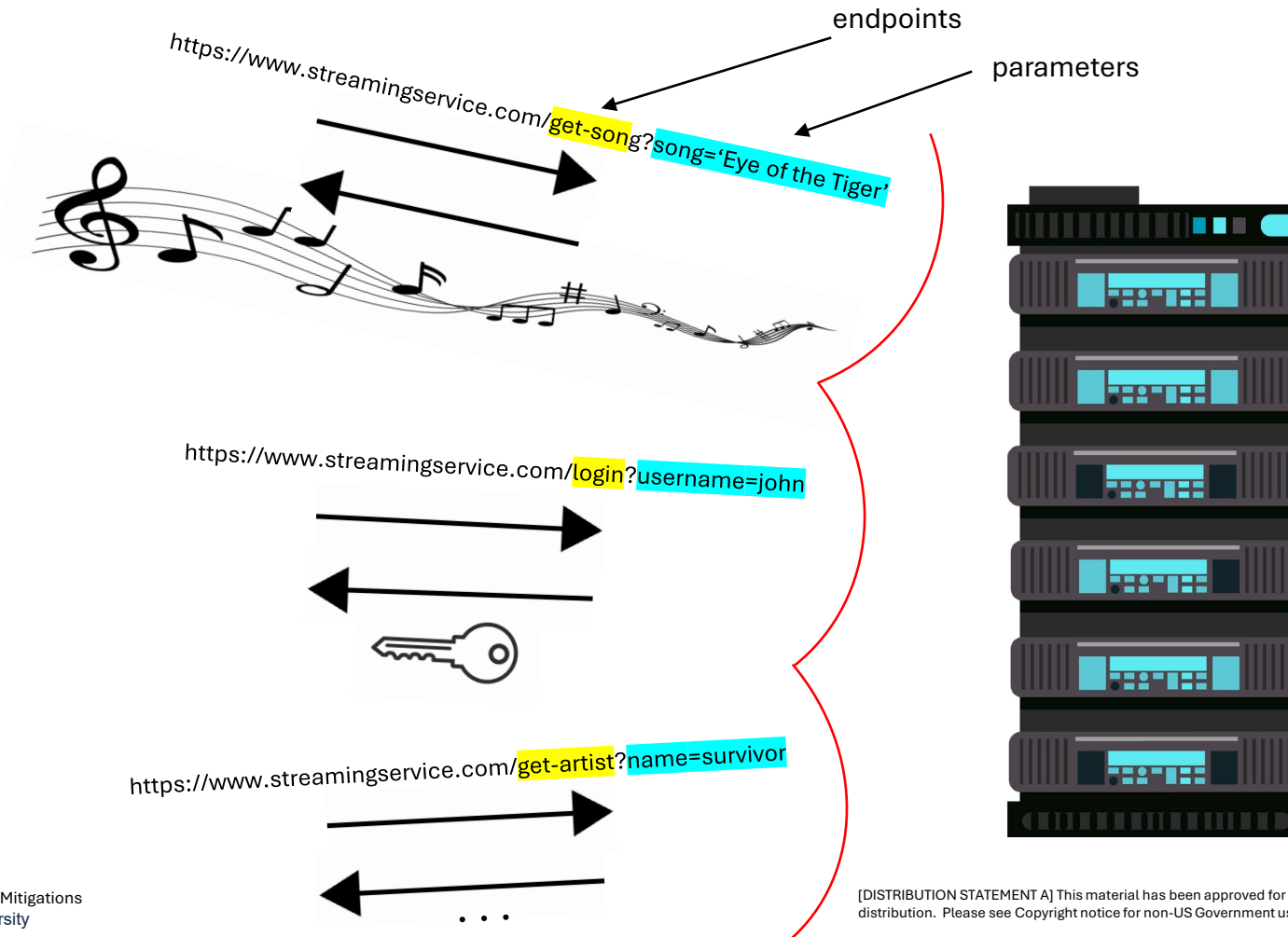
API Structure



API Structure

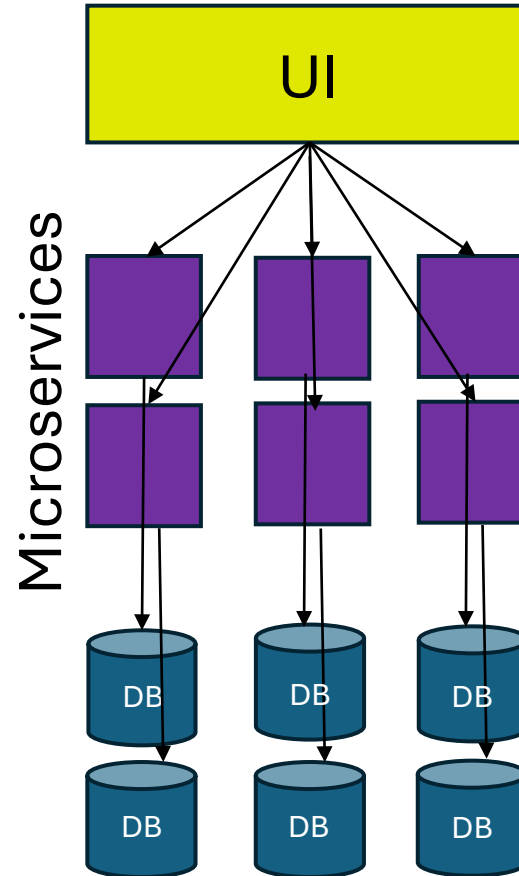
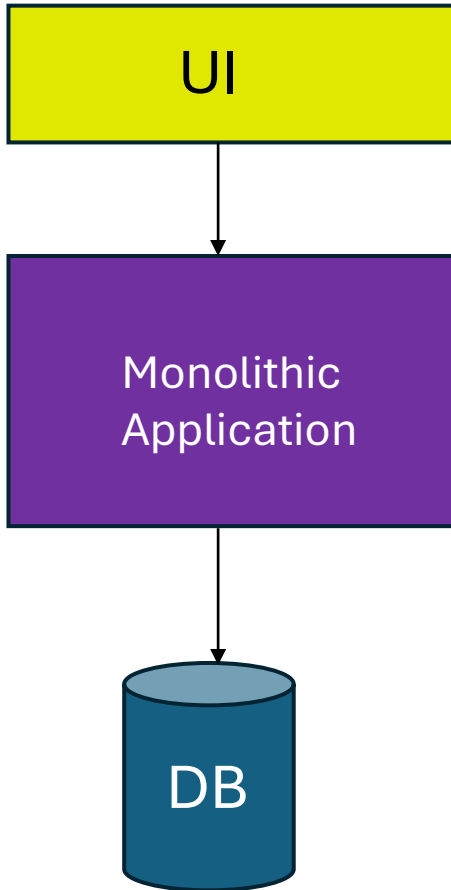


API Structure

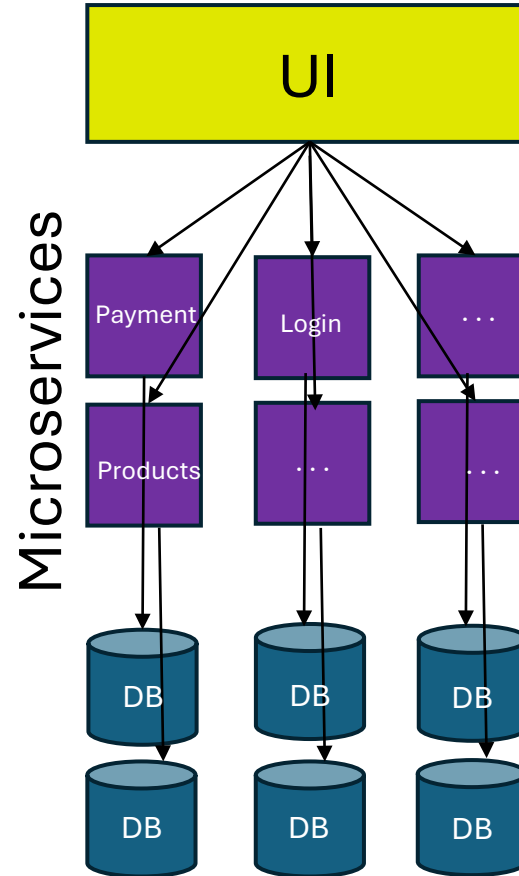
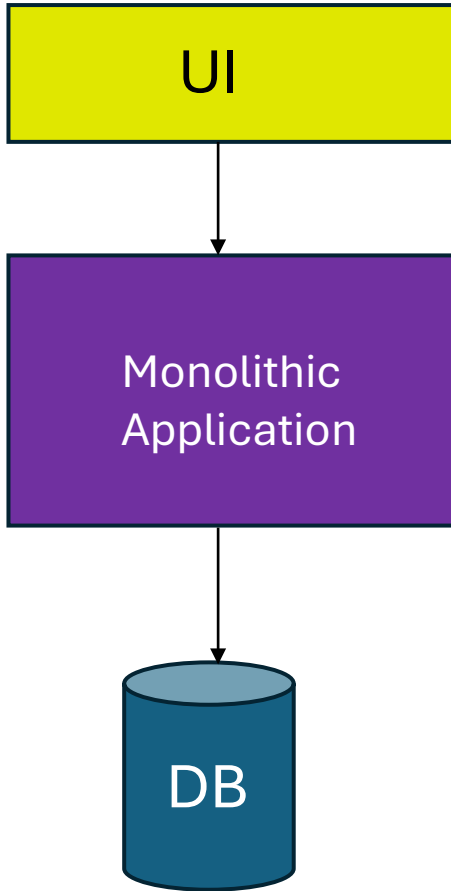


Microservice Architectures

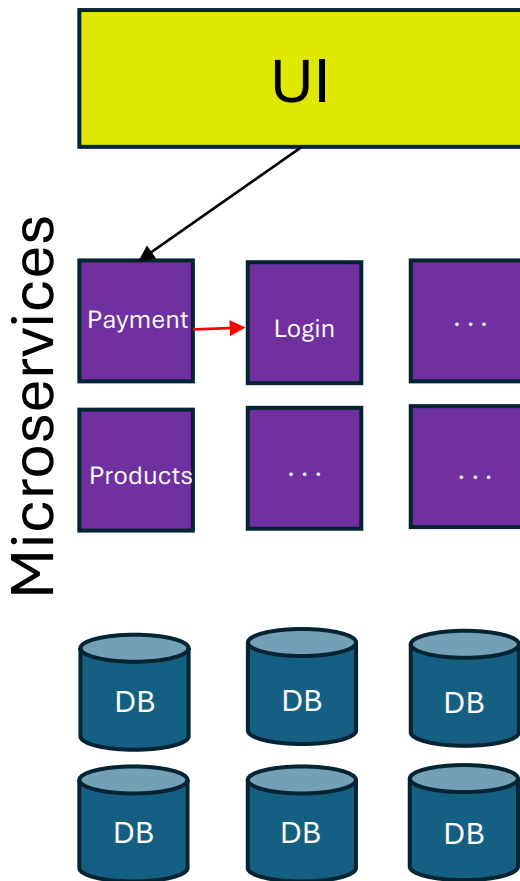
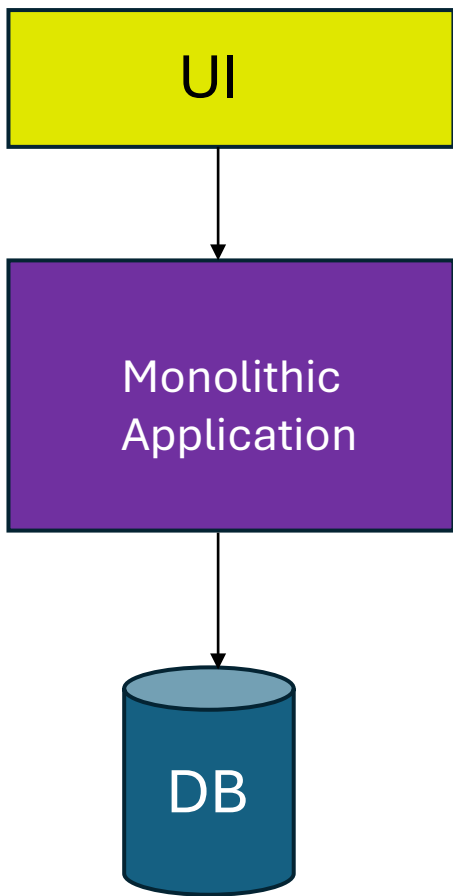
Monolithic vs Microservice Architectures



Monolithic vs Microservice Architectures

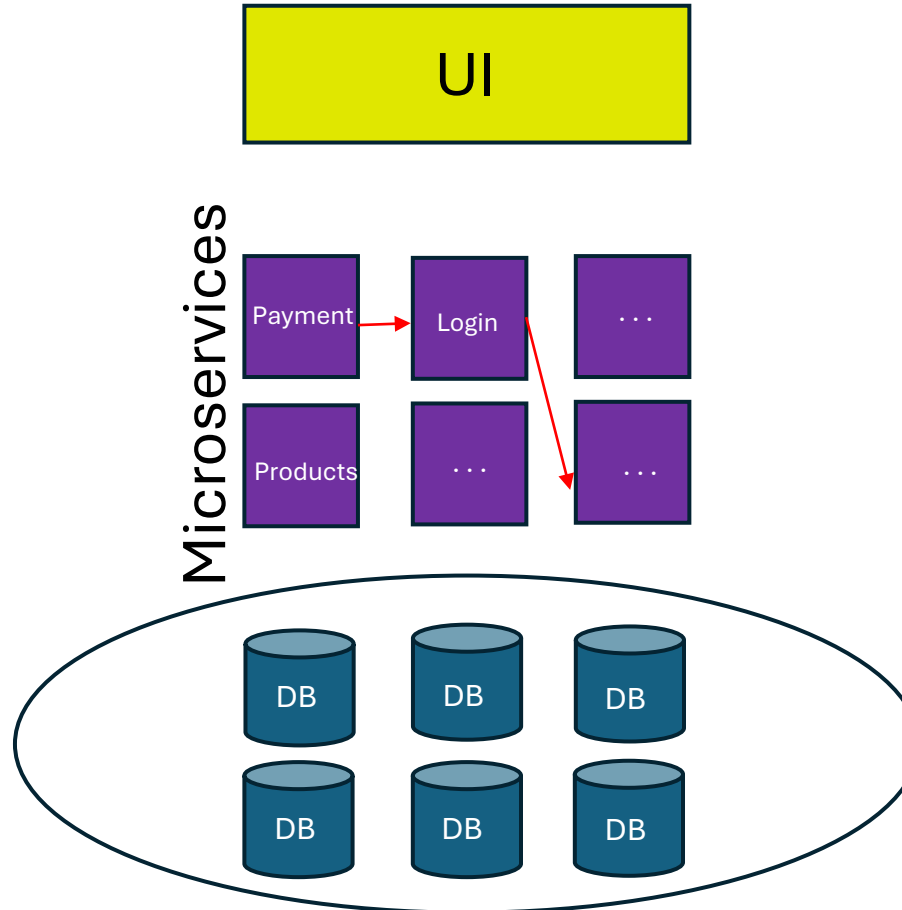


Monolithic vs Microservice Architectures



Risks of Microservice Architectures

Increased Attack Surface



Recommendations

- Add redundancy measures to prevent overload from API use in support structures, such as the API gateway and authentication servers.
- Maintain a thorough inventory of all APIs to prevent shadow and zombie APIs, preferably through auto-generated documentation for uniformity and searchability.
- Configure thorough logging and monitoring efforts for all API activity, including logging all authentication and input failures.^[1]
- Carefully plan API versioning and deployments. Be strategic about adding new APIs so the maintenance and inventory of the existing APIs do not become overwhelming.

Cascading Failures

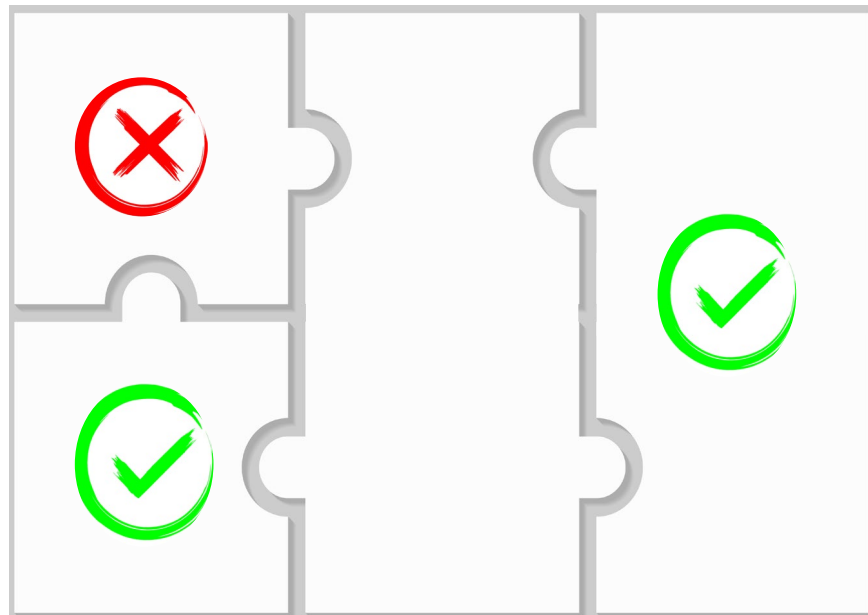


Microservices should be **loosely coupled** so that each of them works independently or mostly independently from other microservices.^[2]

Recommendations

- Design APIs with the goal of loose coupling in mind, including domain coupling, temporal coupling, and implementational coupling.
- Create API documentation that clearly shows which APIs are coupled and the communication flow that happens between coupled microservices. This documentation will be useful for testing and troubleshooting if there are issues with the API after deployment.
- Conduct unit testing, integration testing, and end-to-end testing to confirm the correct functionality of the microservice architecture according to the documented design.
- Strive to prevent single points of failure in the API network by adding redundancy measures in authentication and authorization points and other network structures.^[2]

Third Party Software Integrations



Even the highest quality code can have up to 600 defects per million lines of code while average quality code has around 6,000 defects per million lines of code.^[3]

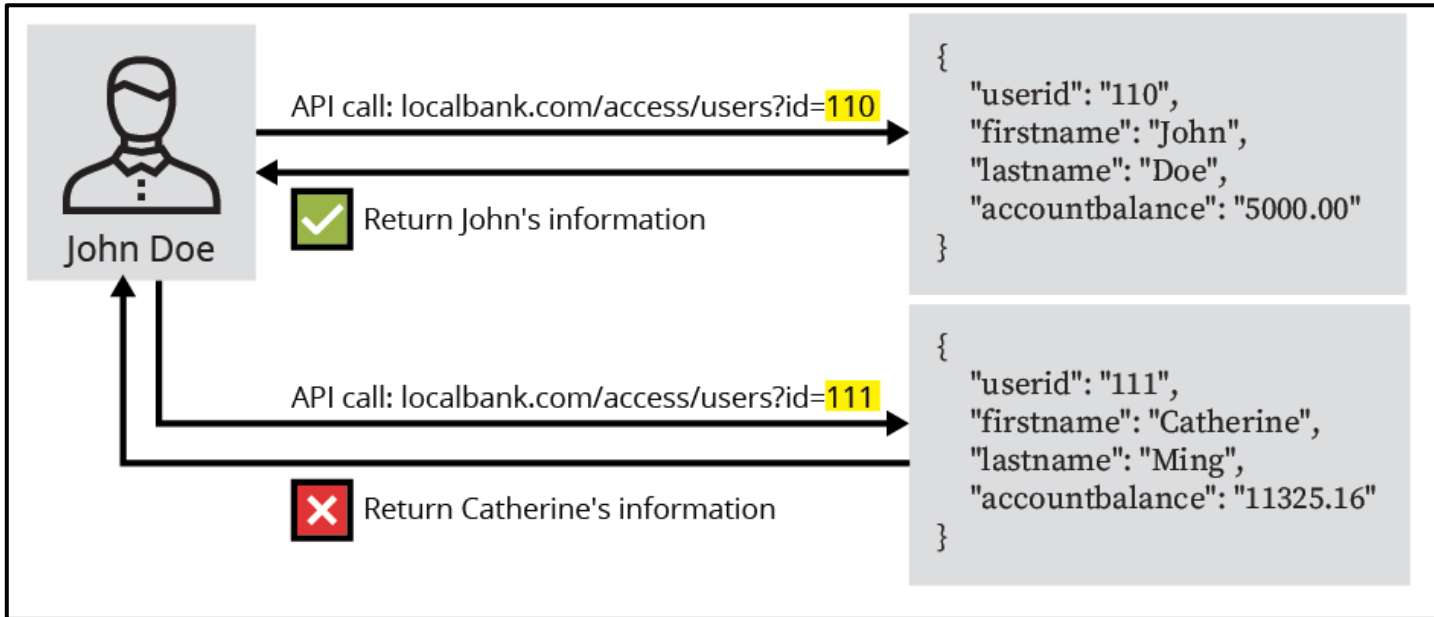
Recommendations

- Perform risk assessments on all third-party software that might be used in a new API to prevent introducing additional API vulnerabilities.
- Stay informed about the latest vulnerabilities in all third-party software that is integrated into currently deployed APIs.
- Use automated vulnerability scanners on third-party software if possible.
- Consider implementing a Software Bill of Materials (SBOM) framework to help secure the supply chain and mitigate the risks of using third-party software.^[3]

Authorization Vulnerabilities

Authorization problems are the top issue in APIs today

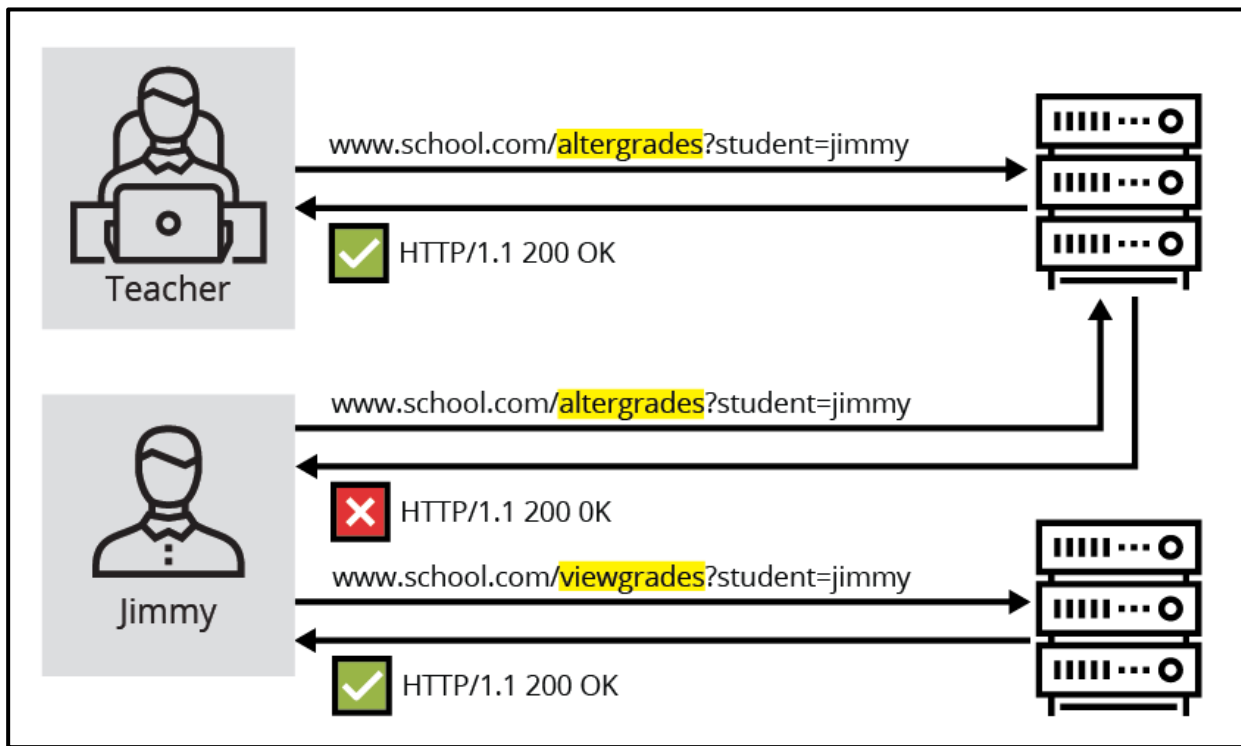
Broken Object Level Authorization (BOLA)



Broken Object Property Level Authorization (BOPLA)



Broken Function Level Authorization (BFLA)



Recommendations

- Keep a thorough inventory of all network APIs.
- Implement static and dynamic testing of all APIs.
- Check authorization vulnerabilities within your API designs and implementations.



References

- [1]<https://www.proquest.com/openview/34b6b229f88e1cfb944a42b8dfebe934/1?pq-origsite=gscholar&cbl=5314840>, Date Accessed: July 26, 2024
- [2]<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-204.pdf>, Date Accessed: July 23, 2024
- [3]https://insights.sei.cmu.edu/documents/5364/Leveraging_SBOM_for_Risk_Reduction.pdf, Date Accessed: July 28, 2024
- [4]<https://owasp.org/API-Security/editions/2023/en/0x11-t10/>, Date Accessed: July 26, 2024
- [5]<https://insights.sei.cmu.edu/documents/5908/api-vulnerabilities-and-risks-2024sr004-1.pdf>, Date Accessed: July 26, 2024

Q&A