

[Open in app](#)

Search



★ Member-only story

Elastic Search — Day 8: Security



Navya Cloudops · Following

8 min read · 5 days ago



Listen



Share



More

Welcome back to our 10-day DevOps Elasticsearch course! Today, we delve into the critical topic of securing your Elasticsearch clusters. As your data grows in importance and volume, ensuring its safety becomes paramount. Let's explore the key facets of Elasticsearch security, including authentication, authorization, and role-based access control (RBAC).



Securing Elasticsearch clusters:

It is of utmost importance to protect the data they contain from unauthorized access, tampering, or breaches. Elasticsearch provides several features and best practices to help you secure your clusters effectively:

1. Authentication and Authorization:

Authentication ensures that only authenticated users can access the Elasticsearch cluster. Common authentication mechanisms include:

- **Native Realm:** This mechanism uses built-in user credentials stored in the Elasticsearch cluster.
- **LDAP/Active Directory:** Integration with existing LDAP or Active Directory infrastructure for centralized user management.
- **OpenID Connect (OIDC):** Authentication via OIDC providers such as Google, Microsoft Azure AD, or Keycloak.

Authorization determines what actions authenticated users can perform within the Elasticsearch cluster. This is typically managed through Role-Based Access Control (RBAC), where you define roles and assign privileges to them. Users are then associated with specific roles. For example, you might have roles like **'admin'**, **'data_analyst'**, or **'monitoring_user'** with varying levels of access privileges.

2. Transport and REST Layer Security:

Securing the transport and REST layers ensures that data exchanged between Elasticsearch nodes and clients is encrypted and authenticated. This prevents eavesdropping and tampering. To implement transport and REST layer security, you can:

- Enable TLS/SSL encryption for both transport and REST layers.
- Generate SSL certificates for each node and configure Elasticsearch to use them.
- Specify settings for certificate validation and client authentication.

3. Encryption:

Encrypting data at rest ensures that even if someone gains physical access to the storage devices, they won't be able to read the data without proper decryption keys. Elasticsearch supports various encryption options, including:

- **File System Encryption:** Encrypting the underlying file system where Elasticsearch data is stored.
- **Index-Level Encryption:** Encrypting individual indices within Elasticsearch using plugins like Search Guard or Elastic's own field- and document-level encryption.

4. Firewall Configuration:

Configure firewalls to control network traffic to and from Elasticsearch nodes. Limit access to only trusted networks and block unauthorized access attempts.

5. Audit Logging:

Enable audit logging to track and monitor actions performed within the Elasticsearch cluster. This helps in identifying security breaches, unauthorized access attempts, or suspicious activities.

6. Regular Updates and Patches:

Keep Elasticsearch and its dependencies up to date with the latest security patches and updates. Vulnerabilities are regularly discovered and patched, so staying current is essential to maintain a secure environment.

Securing Elasticsearch clusters involves implementing a combination of authentication, authorization, encryption, firewall configuration, and monitoring practices.

Authentication and authorization mechanisms:

They are fundamental aspects of securing Elasticsearch clusters, ensuring that only authorized users have access to the system and that they can perform only the actions they are allowed to.

Authentication Mechanisms:

Authentication verifies the identity of users or applications accessing the Elasticsearch cluster. Here are some common authentication mechanisms:

1. Native Realm:

- The native realm is the default authentication mechanism provided by Elasticsearch.
- It relies on built-in user credentials stored within the Elasticsearch cluster.
- Users can be created, modified, and deleted directly within Elasticsearch.
- While simple to set up, it may not be suitable for environments requiring centralized user management or integration with existing authentication systems.

2. LDAP/Active Directory Integration:

- Elasticsearch supports integration with LDAP (Lightweight Directory Access Protocol) and Active Directory.
- This allows organizations to leverage their existing user directories for authentication purposes.
- Users authenticate against the LDAP or Active Directory server, and Elasticsearch trusts the authentication result returned by the directory.

3. OpenID Connect (OIDC):

- OpenID Connect is an authentication layer built on top of OAuth 2.0.
- It allows clients to verify the identity of end-users based on the authentication performed by an authorization server.
- Elasticsearch can act as an OIDC client, delegating authentication to OIDC providers such as Google, Microsoft Azure AD, or Keycloak.

Authorization Mechanisms:

Authorization controls what actions authenticated users can perform within the Elasticsearch cluster. The primary mechanism for authorization in Elasticsearch is Role-Based Access Control (RBAC):

1. Role-Based Access Control (RBAC):

- RBAC allows administrators to define roles and assign privileges to those roles.
- Privileges can be assigned at the cluster level, index level, or even specific actions within the cluster.
- Users are then associated with one or more roles, determining what actions they can perform.
- Example roles might include 'admin', 'data_analyst', 'monitoring_user', etc.

Examples:

Let's consider a scenario where an organization wants to secure its Elasticsearch cluster using authentication and authorization:

- They configure the native realm to allow users to authenticate with Elasticsearch using built-in credentials.

- For centralized user management, they integrate Elasticsearch with their LDAP server, allowing users to log in using their existing LDAP credentials.
- They define roles such as 'admin', 'data_analyst', and 'monitoring_user', each with specific privileges tailored to the user's responsibilities.
- An admin role might have full access to all indices and cluster-level operations.
- A data analyst role might have read access to specific indices containing data for analysis.
- A monitoring user role might have access only to monitoring and diagnostic APIs for cluster health checks.

By combining authentication and authorization mechanisms, organizations can enforce access controls and ensure the security of their Elasticsearch clusters, protecting sensitive data from unauthorized access or modification.

Transport and REST layer security:

These are crucial components of securing data transmission and access to the Elasticsearch cluster. Let's delve into each of these layers:

Transport Layer Security (TLS/SSL):

Transport layer security ensures that communication between Elasticsearch nodes within the cluster is encrypted and authenticated. This prevents unauthorized access and eavesdropping on data transmitted between nodes. Here's how TLS/SSL is implemented in Elasticsearch:

1. Generating SSL Certificates:

- SSL certificates are cryptographic keys used to encrypt and authenticate communication.
- You need to generate SSL certificates for each Elasticsearch node and ensure they are trusted by all nodes in the cluster.

2. Enabling TLS/SSL:

- Once certificates are generated, you enable TLS/SSL in Elasticsearch by configuring the 'elasticsearch.yml' configuration file.

- Set **'xpack.security.transport.ssl.enabled: true'** to enable TLS/SSL for transport layer communication.

3. Certificate Verification:

- Specify the level of certificate verification required by Elasticsearch using the **'xpack.security.transport.ssl.verification_mode'** setting.
- Options include **'certificate'**, **'full'**, **'none'**, etc., depending on the desired level of security.

REST Layer Security:

REST layer security ensures that communication between Elasticsearch clients and the cluster's RESTful API endpoints is encrypted and authenticated. This prevents unauthorized access to cluster resources and data. Here's how to implement REST layer security:

1. Enabling HTTPS:

- Enable HTTPS for the REST layer by configuring the **'elasticsearch.yml'** file.
- Set **'xpack.security.http.ssl.enabled: true'** to enable HTTPS for REST communication.

2. Client Authentication:

- Specify whether client authentication is required for REST communication using the **'xpack.security.http.ssl.client_authentication'** setting.
- Options include **'required'**, **'optional'**, or **'none'**, depending on your security requirements.

3. SSL Certificates for Clients:

- Clients interacting with Elasticsearch via the REST API need to present SSL certificates for authentication if required.
- Configure Elasticsearch to validate client certificates based on your security policies.

By implementing robust transport and REST layer security measures, organizations can ensure the confidentiality, integrity, and authenticity of data transmitted within

the Elasticsearch cluster and accessed via the RESTful API.

Role-based access control (RBAC) in Elasticsearch:

It is a critical feature for controlling access to resources within the cluster. RBAC allows administrators to define roles, assign privileges to those roles, and associate users or groups with specific roles. This fine-grained access control mechanism ensures that users only have access to the resources and operations necessary for their responsibilities.

Here's a more detailed explanation of Role-Based Access Control in Elasticsearch:

1. Role Definition:

Roles define a set of privileges or permissions within the Elasticsearch cluster. These privileges can be scoped at the cluster level, index level, or specific operations within the cluster.

- **Cluster-Level Privileges:** These privileges apply to actions that affect the entire cluster, such as managing indices, executing administrative tasks, or performing cluster-wide monitoring.
- **Index-Level Privileges:** These privileges are specific to individual indices within the Elasticsearch cluster. They control actions like read, write, index management, and document-level operations.

2. Privilege Assignment:

Once roles are defined, privileges are assigned to those roles. Elasticsearch provides a wide range of built-in privileges that can be assigned to roles based on the specific requirements of the users or groups.

Example privileges include:

- **Indices Data Privileges:** read, write, create, delete, manage.
- **Indices Administration Privileges:** create_index, delete_index, aliases.
- **Cluster Privileges:** monitor, manage, manage_security, manage_index_templates, etc.

3. Role Mapping:

After roles are defined and privileges are assigned, users or groups are mapped to these roles. This mapping ensures that users inherit the privileges associated with their assigned roles.

- Users can be assigned one or more roles based on their responsibilities within the organization.
- Groups can also be mapped to roles, allowing for easier management of access control across multiple users with similar roles.

4. Dynamic Roles:

Elasticsearch also supports dynamic roles, where role assignments are based on attributes associated with authenticated users. This allows for more flexible and dynamic access control based on user attributes such as roles, departments, or organizational affiliations.

5. Role Hierarchy:

Elasticsearch allows for the creation of role hierarchies, where roles inherit privileges from parent roles. This simplifies role management by reducing the need to duplicate privileges across multiple roles.

Example Role Definition:

Here's an example of defining a role in Elasticsearch:

```
{
  "cluster": ["monitor"],
  "indices": [
    {
      "names": ["logs-*"],
      "privileges": ["read", "write"]
    }
  ]
}
```

In this example:

- The role has **'monitor'** privileges at the cluster level, allowing users to monitor cluster health.
- The role has **'read'** and **'write'** privileges on indices matching the pattern **'logs-***', allowing users to read and write data to these indices.

Role-Based Access Control in Elasticsearch provides a powerful mechanism for controlling access to cluster resources and operations.

Here are **scenario-based interview questions** for each of the topics mentioned about security in **Elasticsearch Clusters**!

1. A company wants to implement centralized user management for their Elasticsearch cluster. They already have an LDAP server set up. How would you configure Elasticsearch to authenticate users against the LDAP server?
2. Can you outline the steps involved in configuring LDAP integration with Elasticsearch for user authentication? How would you ensure that users authenticate against the LDAP server?
3. An organization is considering implementing OpenID Connect (OIDC) for authentication in their Elasticsearch cluster. What are the key benefits of using OIDC, and how would you set it up?
4. What are the advantages of using OpenID Connect (OIDC) for authentication in Elasticsearch? Can you describe the process of setting up Elasticsearch as an OIDC client?
5. A company wants to ensure that all communication between Elasticsearch nodes is encrypted and authenticated. How would you configure TLS/SSL for transport layer security?
6. Can you explain the steps involved in configuring TLS/SSL for transport layer security in Elasticsearch? What are some best practices for managing SSL certificates in a clustered environment?
7. A security audit revealed that the REST API endpoints in the Elasticsearch cluster are not secured. How would you enable HTTPS and client authentication for REST layer security?
8. Describe the process of enabling HTTPS and client authentication for REST layer security in Elasticsearch. How would you ensure that clients authenticate with valid SSL certificates?
9. An organization needs to grant read-only access to specific indices in their Elasticsearch cluster to a group of data analysts. How would you define roles and assign privileges to meet this requirement?
10. How would you define a role in Elasticsearch to grant read-only access to specific indices? Can you describe the process of mapping users to roles in

Elasticsearch?

11. A new team has been formed within the organization, and they require access to cluster monitoring features only. How would you create a role with the necessary privileges for this team?
12. What steps would you take to create a role in Elasticsearch with privileges limited to cluster monitoring features? How would you ensure that users assigned to this role can access the monitoring tools they need?

Conclusion:

In conclusion, securing your Elasticsearch cluster is a multifaceted task that involves implementing robust authentication, authorization, and encryption mechanisms. By understanding and implementing these security measures, you can ensure the integrity and confidentiality of your data.

Stay tuned for our next session, where we'll explore Elasticsearch Backup and Restore. Until then, happy DevOps-ing!

[Elasticsearch](#)[Security](#)[DevOps](#)[Interview](#)[Learning](#)[Following](#)

Written by Navya Cloudops

514 Followers

More from Navya Cloudops

N Navya Cloudops

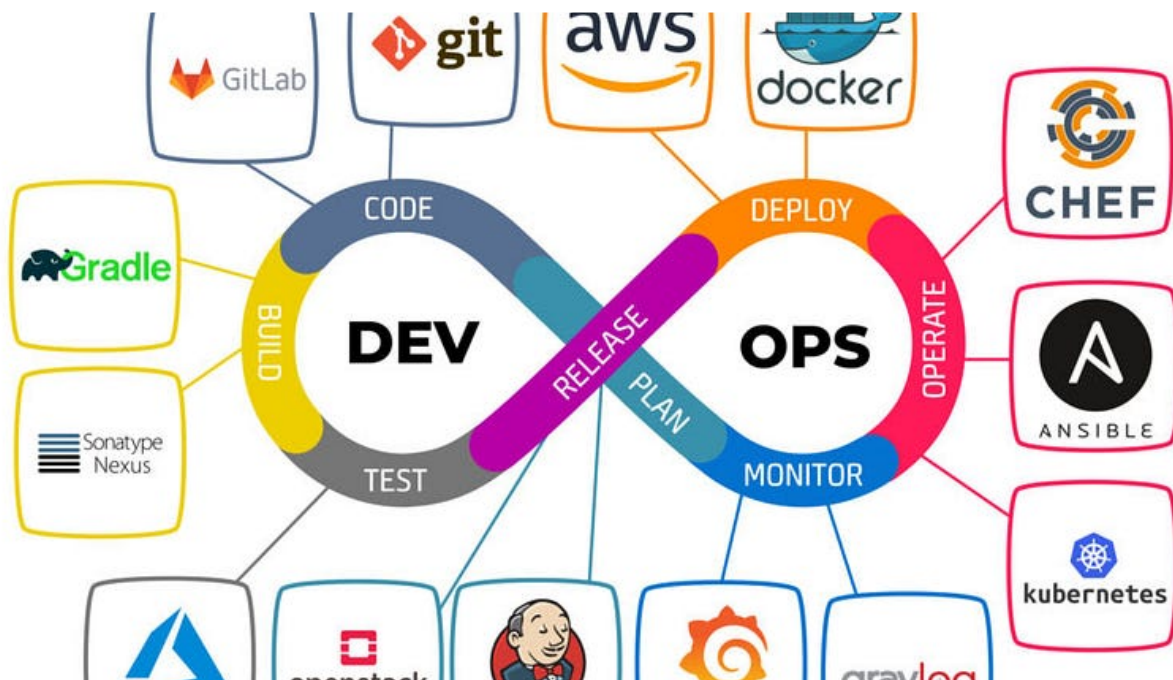
Real-time Interview Questions for 4 years Experience!!

Here are some scenario based interview questions with respect to 4 years experience for a position in CITI Bank.

🌟 · 3 min read · Jan 24, 2024

👏 88 💬

🔖 ...



N Navya Cloudops

DevOps Zero to Hero in 30 days!!

Here's a 30-day DevOps course outline with a detailed topic for each day!

4 min read · Jul 12, 2023



26



1



Navya Cloudops

DevOps Zero to Hero— Day 14: Release Management!!

Welcome to Day 14 of our 30-day course on Software Development Best Practices! In today's session, we'll delve into the critical aspect of...

7 min read · Jul 26, 2023



1





 Navya Cloudops

DevOps Zero to Hero—Day 20: Deployment Strategies

Welcome to Day 20 of our comprehensive 30-day course on Application Deployment! In this segment, we will delve into various deployment...

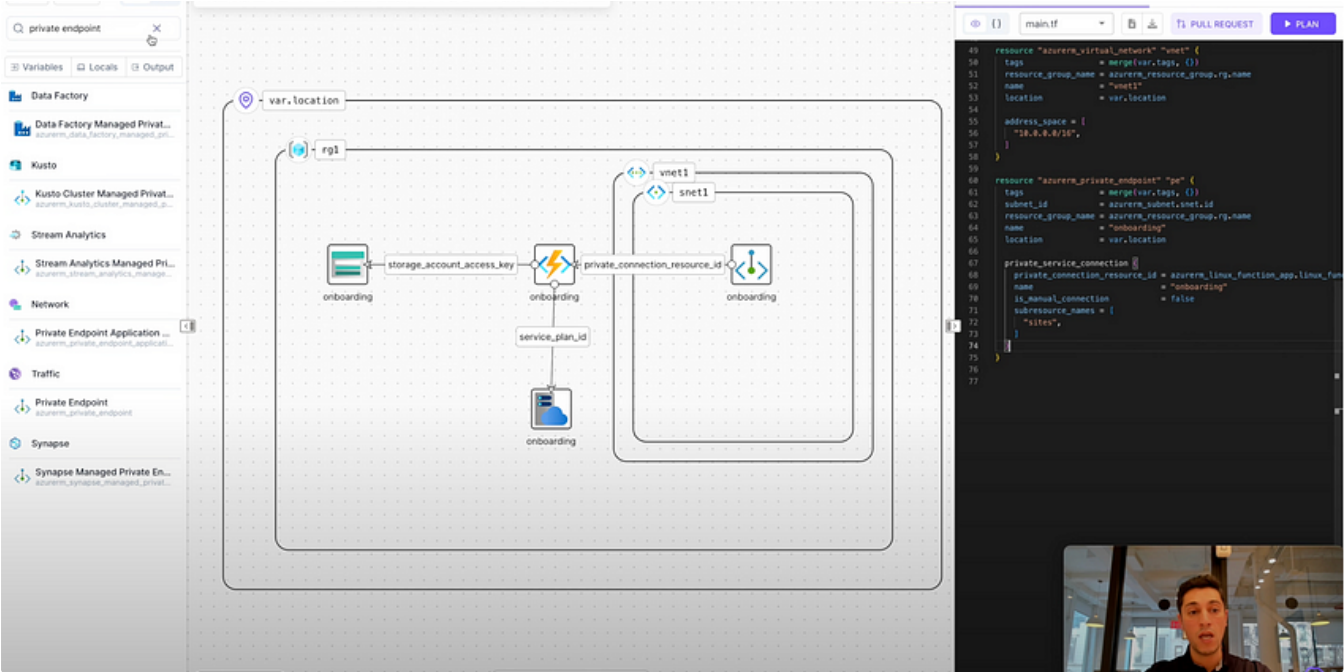
8 min read · Aug 3, 2023


 2 

See all from Navya Cloudops

Recommended from Medium



 Mike Tyson of the Cloud (MToC)

Complete Terraform Tutorial

Your journey in building a cloud infrastructure from scratch and learning Terraform with Brainboard starts here.

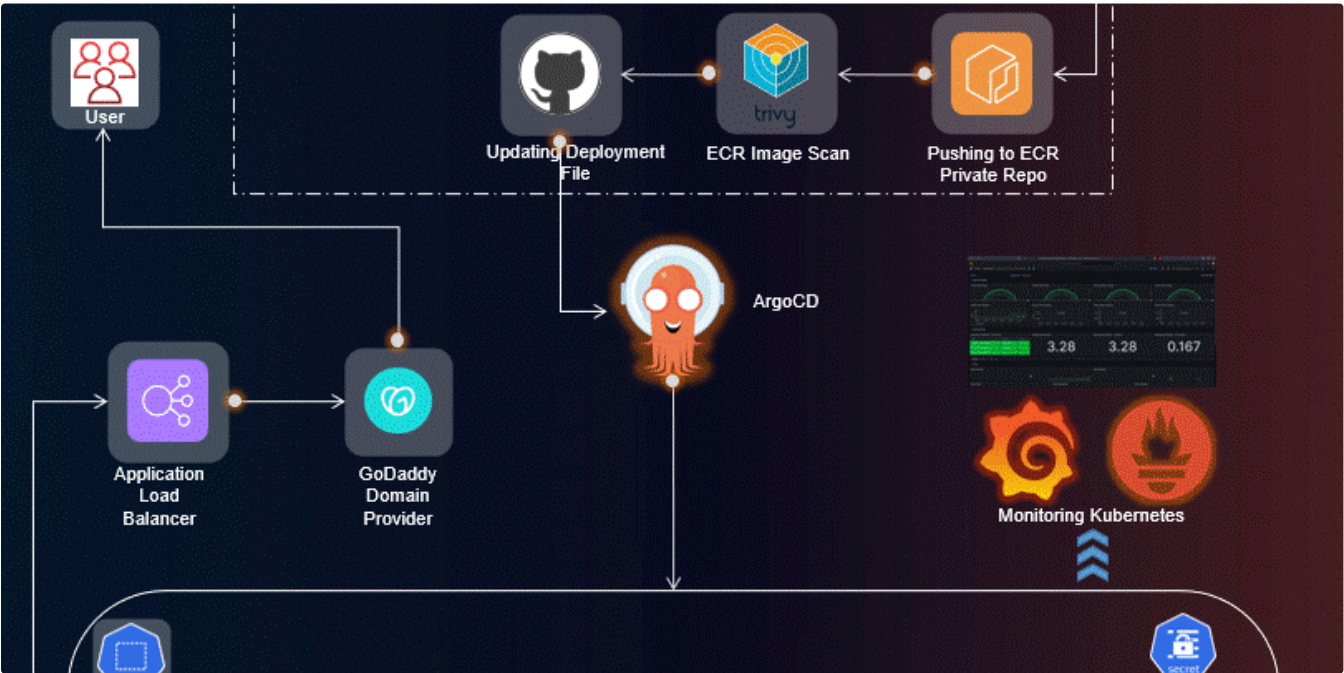
25 min read · Feb 8, 2024


 248









 Aman Pathak in Stackademic

Advanced End-to-End DevSecOps Kubernetes Three-Tier Project using AWS EKS, ArgoCD, Prometheus...

Project Introduction:

23 min read · Jan 18, 2024



1.5K



15



Lists



Self-Improvement 101

20 stories · 1364 saves



How to Find a Mentor

11 stories · 422 saves



Good Product Thinking

11 stories · 470 saves



The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 307 saves



Dolan Miu

Why have 100% Test Coverage

100% test coverage is somewhat of a taboo phrase in software. It's "unachievable" with "diminishing returns" they would say. Non-developers...

4 min read · 3 days ago



53



Chameera Dulanga in Bits and Pieces

Best-Practices for API Authorization

4 Best Practices for API Authorization

9 min read · Feb 6, 2024

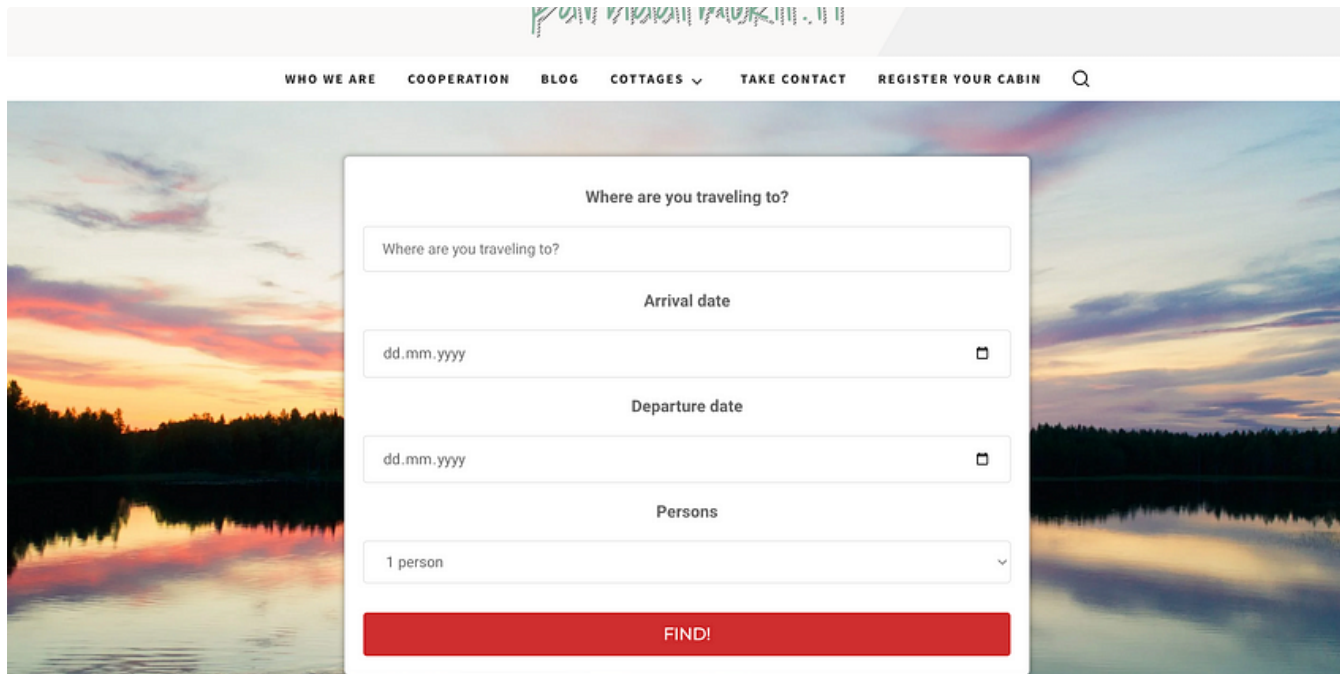


1.1K



4





 Artturi Jalli

I Built an App in 6 Hours that Makes \$1,500/Mo

Copy my strategy!

🌟 · 3 min read · Jan 23, 2024

 9.2K  121



 Akhilesh Mishra

Devops zero to hero #3—Everything you need to know about Dockers

A Complete Guide to start using Docker in your devops workflow

14 min read · Jan 23, 2024

 308







See more recommendations