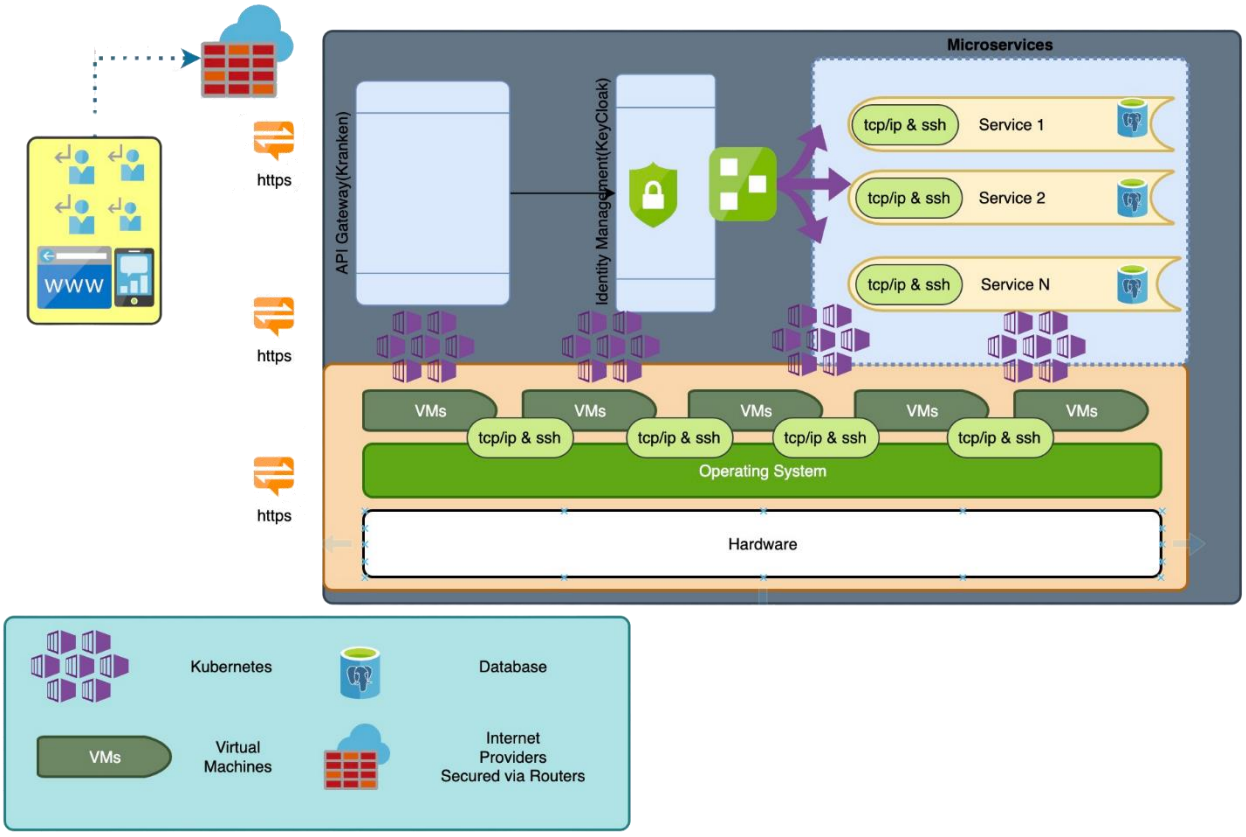
HomeTrumpeter Threat Model & Analysis



Note: Items underlined are assumed to be conducted by HT.

Assets:

1. **Homeowner Data**: Personal details, contact information, property ownership records.
2. **Tenant Data**: Personal details, contact information, rental history.
3. **Property Details**: Location, size, amenities, price, availability status.
4. **Tickets and Tasks Details**: Maintenance requests, complaints, service requests.
5. **Payment Details and History**: Transaction records, bank details, credit card information.

6. **Credentials**: API keys, SSH keys, tokens, and any other credentials used for authentication and authorization.
7. **Sensitive Application Data**: business logic, process data, application settings, environment variables etc.
8. **Infrastructure**: physical hardware, hosted virtual machines and operating systems, Kubernetes cluster, routers, switches, load balancers and firewalls.
9. **Applications**: The web and mobile applications, microservices, API Gateway (Kranken), and Identity Management (KeyCloak).

Threat Agents:

1. **Unauthorized External Users**: Individuals outside the organization trying to gain unauthorized access.
2. **Unauthorized Internal Users**: Employees or associates without the necessary permissions trying to access restricted data.
3. **Authorized Homeowners, Tenants, and Service Providers with Malicious Intent**: Legitimate users who misuse their access rights.
4. **Man-in-the-Middle Attackers**: Attackers who intercept and possibly alter the communication between two parties without their knowledge.
5. **API Attackers**: Individuals targeting the API gateway or microservices directly.
6. **Viruses, Worms, Ransomware, etc.**: Malicious software designed to harm, exploit, or extract data.

Weaknesses/Vulnerabilities/Attack Surfaces:

1. **Network**: Inadequate network security could allow for lateral movement within the system.
2. **Containers**: If not properly secured, containers can be an entry point for attackers. Vulnerabilities in container images or misconfigurations can be exploited.
3. **Kubernetes Orchestration**: Misconfigurations in Kubernetes can expose sensitive information or control interfaces.

4. **API Gateway (Kranken)**: If not properly secured, it can be a target for DDoS attacks, injection attacks, etc.
5. **Identity Management (KeyCloak)**: Weaknesses in authentication or session management can lead to unauthorized access.
6. **Communication Channels**: Unencrypted or weakly encrypted communication can be intercepted and read by attackers.
7. **Web and Mobile Applications**: Vulnerabilities in the application code can be exploited to gain unauthorized access or execute malicious actions.
8. **Database**: If not properly secured, sensitive data can be extracted or tampered with.

Controls:

1. **Network Security**: Firewalls, intrusion detection/prevention systems (IDS/IPS), and network segmentation.
2. **Container Security**: Use trusted base images, regularly scan for vulnerabilities, and employ runtime security.
3. **Kubernetes Security**: Limit access to the Kubernetes API, use network policies, and regularly audit configurations.
4. **API Security**: Implement rate limiting, use strong authentication and authorization mechanisms, and regularly audit and test for vulnerabilities.
5. **Identity Management**: Use multi-factor authentication, regularly review access rights, least privilege principles and use strong encryption for data at rest and in transit.
6. **Encryption**: Use strong encryption protocols for data in transit and at rest.
7. **Regular Patching**: Regularly update all software components to patch known vulnerabilities.
8. **Monitoring and Logging**: Continuously monitor the system for suspicious activities and maintain logs for forensic purposes.
9. **User Training**: Regular security awareness training for all users to recognize phishing attempts and other social engineering attacks.

Essential Assets:

1. **Homeowner Data**: Personal details, contact information, property ownership records.
2. **Tenant Data**: Personal details, contact information, rental history.
3. **Payment Details and History**: Transaction records, bank details, credit card information.
4. **API Tokens**: Used for authentication and authorization for API access.

Abuse Use Cases:

Use Case #	Asset	Abuse Use Case	Countermeasures
1	Web Application Database	Unauthorized Access to Homeowner and Tenant Data	Implement strong authentication mechanisms, input validation, parameterized queries, sanitize user inputs, use a Web Application Firewall (WAF), conduct regular security audits, ensure strong database authentication, encrypt sensitive data in transit and at rest, regularly patch the web application.
2	Communication Channel	Man-in-the-Middle Attack on Communication Channels	Use strong encryption protocols for data in transit, implement HTTPS for all communications.
3	API Gateway	API Token Theft via API Vulnerabilities	Implement API rate limiting, use input validation to prevent injection attacks, regularly update and patch the API.

4	Container Infrastructure	Malicious Activities in Containers	Use trusted base images, regularly scan for vulnerabilities, employ runtime security for containers.
5	Kubernetes Configuration	Misconfigurations in Kubernetes	Ensure Kubernetes configurations are secure, regularly update Kubernetes, monitor for vulnerabilities.
6	API Gateway	DDoS Attack on API Gateway	Implement rate limiting, use services like Cloudflare to mitigate DDoS attacks.
7	Identity Management System	Unauthorized Session Access via Identity Management Weaknesses	Use multi-factor authentication, implement session management best practices, auto logout after inactivity.
8	Database	Database Tampering	Encrypt sensitive data in the database, implement strong authentication mechanisms for database access, regularly monitor and audit database activities.

Risk	Monitoring	Assessment	Mitigation
Technical Challenges	Monitor development progress and track integrated system performance	Review technical task status and assess task completion against schedule	Schedule extra time for troubleshooting, engage HomeTrumpeter's technical team, conduct thorough testing
Scope Creep	Monitor project progress and review product backlog	Assess requirement changes and review stakeholder feature requests	Implement formal change request process, prioritize new requests for future sprints
Resource Constraints	Monitor team workloads and track external dependencies	Evaluate resource impact on timelines	Manage commitments, reallocate tasks as needed, communicate with stakeholders about external delays
Security Vulnerabilities	Conduct security assessments and review security measures	Identify and evaluate security vulnerabilities	Perform regular security testing and code reviews, use automated security checks in CI/CD, penetration testing, implement strong authentication measures
External Dependencies	Track availability and performance of external services	Assess impact of external service disruptions or changes	Develop contingency plans, implement fallback mechanisms, maintain communication with third-party providers

Scope Uncertainty	Seek clarification from HomeTrumpeter and stakeholders	Evaluate gaps in stakeholder requirements	Maintain open communication with HomeTrumpeter, hold regular requirement clarification meetings, document all requirements
Time Management	Monitor sprint progress and review project timeline	Assess task completion within timeframes	Adjust sprint planning, allocate additional time as needed, refine sprint planning process