Husky AI

Owner:
Reviewer:
Contributors: Imported from TM-BOM
Date Generated: Tue Oct 07 2025

Executive Summary

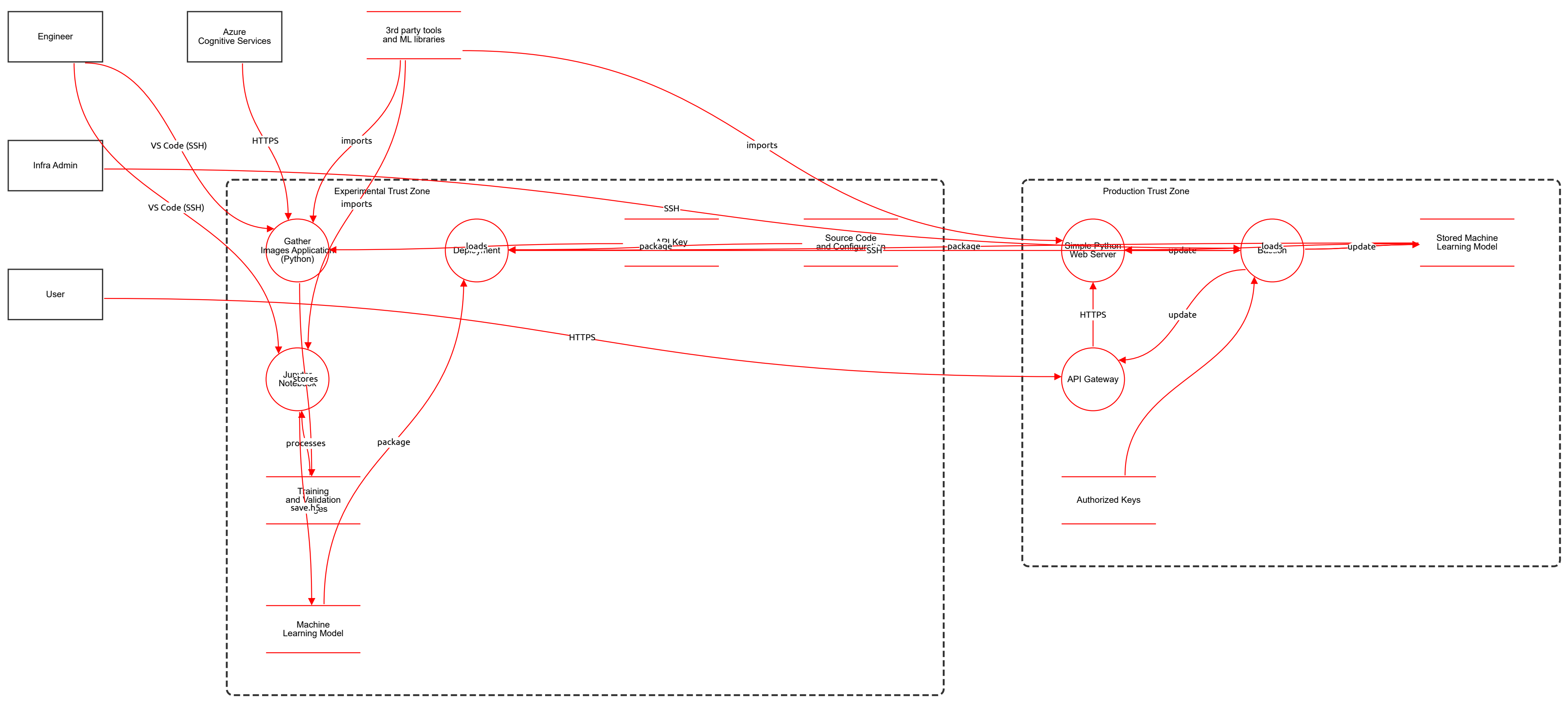
High level system description

A machine learning system to classify Huskies vs dogs. HuskyAI is a machine learning system designed to classify images and distinguish between huskies and non-huskies. It integrates secure data handling practices with a robust convolutional neural network (CNN) for image recognition. Secure Image Retrieval: HuskyAI uses TLS to securely fetch images from Azure Cognitive Services, ensuring encryption during data transmission and validating the server's authenticity to prevent man-in-the-middle attacks. Data Storage and Access Controls: Azure Blob Storage is used to store datasets, with public access fully blocked. Access is controlled using Role-Based Access Control (rbac) and Attribute-Based Access Control (ABAC) to enforce granular, identity-based permissions. Jupyter Notebooks, which host model development and experimentation, are also secured with rbac and ABAC, preventing unauthorized public access. Developer Authentication: Developers access the system through SSH keys protected by passphrases. This adds an additional layer of security, reducing the likelihood of unauthorized access even if keys are exposed. Model and Dataset Dataset Composition: The dataset comprises approximately 1,300 husky images and 3,000 non-husky images sourced via Bing's image search. Data undergoes manual cleansing and is split into training and validation sets to enhance model performance. Model Design: HuskyAI employs a CNN with: Convolutional layers for feature extraction. Max-pooling layers for dimensionality reduction. Dropout layers to prevent overfitting. Dense layers for final classification. The model is trained with the Adam optimizer and a learning rate of 0.0005, optimized for accuracy and computational efficiency. Security Considerations rbac and ABAC controls across storage and development environments ensure sensitive data and configurations are protected. TLS ensures secure communication channels, preventing eavesdropping or data interception during image retrieval. Applications HuskyAI is tailored for accurate image classification and can be adapted for other domains requiring precise visual differentiation, with a focus on maintaining strong security postures. HuskyAI combines state-of-the-art machine learning techniques with stringent security controls, including secure communications, robust access management, and encrypted developer authentication, to deliver a reliable and secure image classification system.

Summary

Total Threats	43
Total Mitigated	0
Total Open	43
Open / Critical Severity	0
Open / High Severity	37
Open / Medium Severity	6
Open / Low Severity	0

Husky AI


Engineer
Azure Cognitive Services
3rd party tools and ML libraries
Infra Admin
User
VS Code (SSH)
HTTPS
imports
imports
VS Code (SSH)
Experimental Trust Zone
imports
SSH
Gather Images Application (Python)
Deployment
loads
API Key
package
Source Code and Configuration
SSH
package
Jupyter Notebook
stores
HTTPS
processes
package
Training and Validation Images
save.h5
Machine Learning Model
Production Trust Zone
Simple Python Web Server
update
Bastion
loads
update
Stored Machine Learning Model
HTTPS
update
API Gateway
Authorized Keys

Husky AI

Engineer (Actor)

Description: A Data Engineer responsible for building, training, and deploying machine learning models.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Compromised engineer credentials enabling unauthorized SSH into Experimental Trust Zone	Spoofing	High	Open		The actor Engineer initiates VS Code (SSH) sessions into components inside the Experimental Trust Zone. If the engineer's SSH keys or workstation are compromised, an attacker can impersonate the engineer and gain interactive access to Gather Images Application (Python) and Jupyter Notebook.	- Enforce phishing-resistant MFA (FIDO2/WebAuthn) for SSH via certificates or strong bastion workflows - Use short-lived SSH certificates (CA-signed) with Just-In-Time access and source IP allowlisting - Enforce host key verification, device posture checks, and session recording - Rotate keys regularly and store keys in hardware-backed secure enclaves (TPM/SmartCard)

Infra Admin (Actor)

Description: Administrator responsible for securing and maintaining production infrastructure.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Admin account/key takeover enabling production bastion access	Spoofing	High	Open		The actor Infra Admin connects via SSH to Bastion within the Production Trust Zone. Stolen or phished admin credentials/keys allow attackers to impersonate the administrator and pivot to production systems (Simple Python Web Server, Stored Machine Learning Model).	- Require MFA for SSH (e.g., PAM + FIDO2), with short-lived certs and IP allowlisting - Use privileged access management (PAM) with approval workflows and session recording - Enforce hardware-backed private keys, regular rotation, and least-privilege roles

Azure Cognitive Services (Actor)

Description: External service providing resources for machine learning experimentation.

Number	Title	Type	Severity	Status	Score	Description	Mitigations

User (Actor)

Description: External user interacting with the HuskyAI system via the API Gateway.

Number	Title	Type	Severity	Status	Score	Description	Mitigations

3rd party tools and ML libraries (Store)

Description: External third party tools for the services

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Supply chain compromise of external tools and ML libraries	Tampering	High	Open		3rd party tools and ML libraries are external dependencies used by Experimental and Production components. A compromised upstream package or repository can inject malicious code or backdoors.	- Pin versions; verify integrity via checksums/signatures (e.g., Sigstore) - Use a private, vetted package mirror with allowlists and malware scanning - Generate and verify SBOMs; block implicit dependency resolution from public registries in CI/CD

Gather Images Application (Python) (Process)

Description: This is a Python-based application responsible for gathering images from external sources, specifically Azure Cognitive Services, and storing them in the designated Training and Validation Images storage.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Data set poisoning through ingestion of malicious external images	Tampering	High	Open		Gather Images Application (Python) runs inside the Experimental Trust Zone but ingests images from external sources (e.g., Azure Cognitive Services and third-party tools). Malicious or mislabeled images can poison the Training and Validation Images store, degrading or subverting model behavior.	- Validate and sanitize inputs; use content-type/size checks and malware scanning - Maintain a curated, signed dataset pipeline with human review and anomaly detection - Segregate raw vs. curated datasets and enforce write-once/immutable policies for curated sets

Jupyter Notebook (Process)

Description: A Jupyter Notebook environment that processes the images stored in Training and Validation Images, executes code using external ML libraries, and provides a UI for engineers to interact with and manipulate data, allowing for iterative model development. It can save trained machine learning models to Machine Learning Model storage.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Arbitrary code execution via notebooks leading to environment takeover	Elevation of Privilege	High	Open		Jupyter Notebook in the Experimental Trust Zone executes arbitrary code and imports third-party libraries. A malicious notebook, dependency, or user action could escalate privileges or exfiltrate secrets/data.	- Require strong authentication and per-user authorization with least privilege - Run kernels as non-root with OS sandboxing (AppArmor/seccomp) and network egress policies - Disable or restrict internet access from notebooks where possible; restrict secrets access - Enable detailed auditing of notebook actions and package provenance controls

Deployment (Process)

Description: Handles the deployment of the machine learning model by packaging the model and all necessary source code and configuration stored in Source Code and Configuration. It receives the final model from Jupyter Notebook and prepares it for deployment to the production environment.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Packaging of tampered artifacts for promotion to production	Tampering	High	Open		Deployment in the Experimental Trust Zone collects the model from Machine Learning Model and code from Source Code and Configuration. Unsigned or tampered artifacts can be packaged and later deployed to production, compromising the runtime.	- Enforce artifact signing (models and packages) and verify signatures in CI/CD - Use reproducible builds, SBOMs, and policy-based admission (e.g., SLSA/Sigstore) - Restrict who/what can write to artifact stores; enforce change approvals

Training and Validation Images (Store)

Description: Contains images used for training and validation of machine learning models.
Data set: Training and Validation Images
Contains images used for training and validation of machine learning models.
Record count maximum of 100000 with data sensitivity of biz and access control methods of rbac

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Training data poisoning in blob storage	Tampering	High	Open		Training and Validation Images resides in the Experimental Trust Zone. An attacker with write access (directly or via compromised processes) can inject mislabeled or adversarial images to bias the model.	- Enforce RBAC/ABAC with least privilege and write-once immutability for curated datasets - Require dataset signing and verification; maintain provenance and review workflows - Continuous data quality/anomaly detection and change auditing

API Key (Store)

Description: Stores API keys for secure access to external services.
Data set: API Keys
Stores API keys for secure access to external services.
Record count maximum of 20 with data sensitivity of cred and access control methods of rbac

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Exposure of API keys enabling unauthorized external access	Information Disclosure	High	Open		API Key in the Experimental Trust Zone stores credentials for external services. Key leakage (misconfig, logs, or weak access controls) could allow attackers to call external APIs or pivot.	- Store keys in a secrets manager (HSM/KMS-backed) with envelope encryption - Scope keys to least privilege; rotate frequently and monitor usage with alerts - Prevent key exposure in logs/env vars; use short-lived tokens where possible

Machine Learning Model (Store)

Description: Contains the machine learning models in serialized format.
Data set: Bastion Logs
Contains trained machine learning models in serialized format for production use.
Record count maximum of 5000 with data sensitivity of biz and access control methods of acl

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Model artifact tampering before deployment	Tampering	High	Open		Machine Learning Model store in the Experimental Trust Zone holds serialized models. A tampered model could be promoted, leading to backdoors or inference failures in production.	- Sign models and verify signatures before packaging/deploying - Maintain immutable artifact versioning with RBAC/ABAC on writes - Scan/validate model files and enforce promotion only from trusted pipelines

Source Code and Configuration (Store)

Description: Stores source code and configuration files for deployment and production setup.
Data set: Source Code and Configuration
Stores source code and configuration files for deployment and production setup.
Record count maximum of 200 with data sensitivity of biz and access control methods of rbac

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unauthorized modification of source code or runtime configuration	Tampering	High	Open		Source Code and Configuration in the Experimental Trust Zone feeds Deployment. Malicious changes can introduce vulnerabilities or secrets leakage into production.	- Enforce branch protection, mandatory code review, and signed commits/tags - Secret scanning and policy-as-code checks in CI - Restrict write access and require change approvals

Simple Python Web Server (Process)

Description: Serves as simple web server

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Inference endpoint resource exhaustion	Denial of Service	High	Open		Simple Python Web Server in the Production Trust Zone processes requests from API Gateway. Attackers can send large or numerous images/requests to exhaust CPU/memory and impact availability.	- Enforce request authentication/authorization, rate limiting, and quotas at API Gateway - Apply payload size limits, timeouts, circuit breakers, and prioritization - Autoscale with admission control and implement cache where feasible

API Gateway (Process)

Description: Serves as the entry point for external users to interact with the production environment via HTTPS. It routes user requests to the Simple Python Web Server and ensures secure communication. The API Gateway enforces request validation and manages APIs exposed to the public while ensuring access control to internal services.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Internet-facing API flooding and volumetric attacks	Denial of Service	High	Open		API Gateway is the ingress point in the Production Trust Zone for external User traffic. Despite HTTPS, it remains susceptible to volumetric floods and application-layer request storms.	- Front with DDoS protection and WAF; apply adaptive rate limiting and IP reputation filtering - Enforce JWT/OAuth validation before forwarding; prioritize authenticated traffic - Use autoscaling with surge protection and SYN/connection limits

Bastion (Process)

Description: A secure access management component for administrative functions. It provides controlled SSH access for the Infrastructure Admin to internal production resources, such as the Stored Machine Learning Model and Simple Python Web Server.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Bastion compromise enabling lateral movement across production	Elevation of Privilege	High	Open		Bastion in the Production Trust Zone has administrative reach to Simple Python Web Server and Stored Machine Learning Model. If compromised, attackers can escalate privileges and tamper with production assets.	- Harden OS/SSH (no root login, strong ciphers), enforce MFA and JIT access - Isolate bastion on a dedicated subnet/security group; monitor with EDR and session recording - Limit reachable targets via firewall rules and use command allowlists

Authorized Keys (Store)

Description: Contains SSH keys used for securing administrative access.
Data set: Authorized Keys
Contains SSH keys used for securing administrative access.
Record count maximum of 100 with data sensitivity of cred and access control methods of rbac

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Disclosure of SSH authorized keys enabling unauthorized production access	Information Disclosure	High	Open		Authorized Keys in the Production Trust Zone hold credentials used by Bastion. Compromise of this store enables attackers to authenticate to production systems.	- Store keys in an HSM/KMS-backed secrets manager with strict RBAC and audit - Use short-lived cert-based SSH instead of static keys; rotate and revoke rapidly - Encrypt backups and prevent key exposure in logs/config

Stored Machine Learning Model (Store)

Description: Contains storage for machine learning models in serialized format.
Data set: Stored Machine Learning Models
Contains trained machine learning models in serialized format for production use.
Record count maximum of 10 with data sensitivity of biz and access control methods of rbac

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Production model tampering due to lack of encryption/integrity	Tampering	High	Open		Stored Machine Learning Model in the Production Trust Zone is marked as not encrypted. Attackers with access could alter models to introduce hidden behaviors or denial of service at inference time.	- Enable encryption at rest and enforce integrity checks (signatures) on load - Restrict write access to CI/CD only; use immutability and versioning - Monitor access patterns and verify checksums before serving

HTTPS (Data Flow)

Description: Transfer data from Azure Cognitive Services to Gather Images Application in Python.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Impersonation of external API endpoint despite TLS	Spoofing	Medium	Open		The flow HTTPS crosses from external Azure Cognitive Services into the Experimental Trust Zone targeting Gather Images Application (Python). Even with TLS enabled, lack of certificate pinning or token validation could allow endpoint spoofing through misconfiguration or compromised trust stores.	- Enforce TLS 1.2+ with certificate pinning and strict CA validation - Use OAuth2/AKV-signed requests or mTLS for server identity - Validate response domains/hosts and rotate credentials frequently
	Data poisoning via external image retrieval	Tampering	High	Open		The flow from Azure Cognitive Services to Gather Images Application (Python) ingests images across a trust boundary into the Experimental Trust Zone. Malicious or mislabeled content may poison downstream training data.	- Content validation (MIME/size), malware scanning, and allowlisted sources - Human-in-the-loop sampling and anomaly detection on incoming images - Maintain a quarantine/curation step before adding to training datasets

imports (Data Flow)

Description: Transfer data from Third Party tools and ML libraries to Gather Images Application in Python.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Malicious dependency injection during imports to Gather Images	Tampering	High	Open		The flow imports crosses from external 3rd party tools and ML libraries into Gather Images Application (Python) inside the Experimental Trust Zone. A compromised package mirror or dependency confusion can deliver tampered code.	- Use private package repositories with allowlists and signature verification (Sigstore) - Pin exact versions and verify checksums/SBOMs; disable implicit latest resolves - Run builds in isolated networks with no direct internet access
	Code execution via poisoned library	Elevation of Privilege	High	Open		Imported libraries can execute code within Gather Images Application (Python), enabling attackers to gain execution inside the Experimental Trust Zone.	- Run with least-privileged service accounts; sandbox process (AppArmor/seccomp) - Restrict filesystem and secret access; monitor egress and behavior - Verify package provenance and enforce policy controls in CI/CD

imports (Data Flow)

Description: Transfer data from Third Party tools and ML libraries to Jupyter Notebook.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Poisoned ML library imported into Jupyter Notebook	Tampering	High	Open		The flow imports crosses from external 3rd party tools and ML libraries into Jupyter Notebook within the Experimental Trust Zone. A malicious dependency can alter computations or exfiltrate data.	- Vendor critical dependencies; enforce signature verification and checksums - Use a private, vetted mirror and dependency allowlists - Lock environments with hashed lockfiles and reproducible builds
	RCE via compromised notebook dependency	Elevation of Privilege	High	Open		Imported packages execute within Jupyter Notebook, potentially granting attackers code execution inside the Experimental Trust Zone.	- Non-root users, kernel isolation, and network egress restrictions - Execution policy enforcement and package provenance controls - Continuous monitoring and alerting on abnormal notebook activity

VS Code (SSH) (Data Flow)

Description: Transfer data from Engineer to Gather Images Application in Python.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Stolen SSH keys enabling direct access to Gather Images	Spoofing	High	Open		The VS Code (SSH) flow crosses from the external actor Engineer into the Experimental Trust Zone to Gather Images Application (Python). If SSH keys are stolen, attackers can impersonate the engineer and modify code/data.	- Use SSH certificates with short lifetimes, MFA, IP allowlists, and session logging - Enforce host key pinning and device posture checks - Prefer access via bastion with JIT approvals instead of direct host access

VS Code (SSH) (Data Flow)

Description: Transfer code and ML models from Engineer locally to Jupyter Notebook.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Engineer account impersonation for Notebook access	Spoofing	High	Open		The VS Code (SSH) flow crosses from an external Engineer into the Experimental Trust Zone to Jupyter Notebook. A compromised identity/key enables unauthorized notebook access and arbitrary code execution.	- MFA-backed SSH certificates and JIT access via bastion - Strong audit logging, command restrictions, and IP allowlisting - Rotate credentials and revoke compromised keys immediately

stores (Data Flow)

Description: Transfer images from Gather Images Application to Training and Validation Images.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unencrypted internal transfer of training images	Information Disclosure	Medium	Open		The flow stores within the Experimental Trust Zone from Gather Images Application (Python) to Training and Validation Images is marked as not encrypted. Traffic sniffing on the internal network could expose dataset contents.	- Enforce TLS 1.2+ for all internal service-to-store communications - Use private endpoints and network ACLs; segregate data planes - Disable plaintext protocols and validate certificates
	On-path manipulation of images in transit	Tampering	Medium	Open		Unencrypted flow inside the Experimental Trust Zone allows an attacker with internal foothold to modify images before storage, poisoning the dataset.	- Use TLS with integrity checks and authenticated clients (mTLS) - Enable write-once/immutable storage for curated datasets and verify checksums on upload

loads (Data Flow)

Description: API Key Storage to Gather Images Application in Python.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	API key leakage over unencrypted internal flow	Information Disclosure	High	Open		The flow loads from API Key to Gather Images Application (Python) inside the Experimental Trust Zone is not encrypted. Keys traversing the network can be intercepted and reused.	- Deliver secrets via a local sidecar/agent with TLS and short-lived tokens - Use mTLS, TLS 1.2+, and secret retrieval with audience-bound tokens - Avoid transmitting secrets over the network where possible (mount via secure vault)

processes (Data Flow)

Description: Load from Training and Validation Images to Jupyter Notebook.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Training data tampering in transit to Notebook	Tampering	Medium	Open		The flow processes from Training and Validation Images to Jupyter Notebook within the Experimental Trust Zone is unencrypted, enabling manipulation of batches in flight and subtle poisoning.	- Enforce TLS/mTLS for datastore reads - Validate dataset integrity with checksums/signatures at load time - Limit network exposure with private endpoints and ACLs

package (Data Flow)

Description: Transfer data from Machine Learning Model to Deployment.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Model artifact alteration en route to Deployment	Tampering	High	Open		The flow package from Machine Learning Model to Deployment within the Experimental Trust Zone is not encrypted. An internal attacker could alter the model in transit, impacting the deployed package.	- Enable TLS/mTLS and verify model signatures/checksums on receipt - Pull artifacts from a trusted registry over authenticated channels - Restrict network paths and enforce allowlisted callers

save.h5 (Data Flow)

Description: Transfer final model from Jupyter Notebook to Machine Learning Model.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unsigned model written to storage	Tampering	Medium	Open		The flow save.h5 from Jupyter Notebook to Machine Learning Model in the Experimental Trust Zone is unencrypted and lacks integrity guarantees, allowing potential tampering of the stored model.	- Require artifact signing at write time and verify on read - Use TLS/mTLS for storage writes and immutable versioning - Restrict write permissions to CI/service identities only

package (Data Flow)

Description: Transfer from Machine Learning Model Blob to Deployment Service.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Cross-zone model exfiltration (Production to Experimental) without encryption	Information Disclosure	High	Open		The flow package crosses from Stored Machine Learning Model in the Production Trust Zone to Deployment in the Experimental Trust Zone and is marked as not encrypted, risking leakage of production model artifacts across trust boundaries.	- Prohibit cross-zone artifact pulls; instead promote via controlled CI/CD within production - Enforce TLS/mTLS and data minimization if movement is necessary - Implement approval gates and audit trails for any cross-zone transfers
	Cross-zone artifact tampering during transfer	Tampering	High	Open		The same cross-boundary flow allows tampering with production models when moving into a less trusted Experimental Trust Zone, risking backdoored artifacts being reintroduced.	- Only allow artifact promotion from trusted pipelines with signature verification (model signing) - Use immutable artifact registries and block writes from lower-trust zones - Network ACLs to deny Experimental → Production write paths

package (Data Flow)

Description: Transfer data from Source Code and Configuration to Deployment.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Code/config tampering in transit to Deployment	Tampering	High	Open		The flow package from Source Code and Configuration to Deployment inside the Experimental Trust Zone is not encrypted and may lack integrity checks, enabling unnoticed modification of code or configuration.	- Enforce TLS/mTLS and verify signed commits/tags and checksums - Retrieve from a trusted VCS over authenticated channels; require merge approvals - Immutable, versioned configuration with policy enforcement

HTTPS (Data Flow)

Description: Transfer from User to API Gateway.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Public API request floods degrading availability	Denial of Service	High	Open		The flow HTTPS crosses from external User into the Production Trust Zone at API Gateway. Even with TLS, unauthenticated or low-cost requests can overwhelm the gateway or downstream services.	- WAF + DDoS protection, CAPTCHA/Challenge for anonymous traffic, per-IP/user rate limits - Early authentication and request validation; block oversized payloads - Autoscale with circuit breakers and backpressure

update (Data Flow)

Description: Transfer data from Bastion to API Gateway.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unencrypted admin update channel from Bastion to API Gateway	Tampering	High	Open		The flow update runs inside the Production Trust Zone but is marked as not encrypted. An attacker with internal foothold could modify admin updates or inject commands to API Gateway.	- Enforce TLS/mTLS for all administrative flows and restrict to bastion-managed identities - Require change approvals and immutable audit logs for admin updates - Limit admin API surface and use signed change bundles

HTTPS (Data Flow)

Description: Transfer data from API Gateway to Simple Python Web Server.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Misconfigured HTTPS between API Gateway and Web Server (no encryption)	Information Disclosure	High	Open		The flow HTTPS from API Gateway to Simple Python Web Server within the Production Trust Zone is marked as not encrypted, indicating misconfiguration or plaintext traffic. Sensitive requests/responses could be exposed or altered on-path.	- Enforce TLS 1.2+ end-to-end with certificate validation or mTLS - Private networking and strict ACLs between gateway and backend - HSTS and TLS policy checks in deployment pipelines

update (Data Flow)

Description: Transfer data from Bastion to Simple Python Web Server.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unencrypted admin update path enabling code/config tampering	Tampering	High	Open		The flow update from Bastion to Simple Python Web Server inside the Production Trust Zone is unencrypted, allowing on-path modification of deployments or commands.	- Require TLS/mTLS and signed, attested deployments from CI/CD - Limit direct admin channels; prefer pull-based, verified deployments - Maintain immutable logs and approvals for changes

loads (Data Flow)

Description: Transfer sensitive data from Stored Machine Learning Model to Simple Python Web Server.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Model tampering during load to runtime	Tampering	High	Open		The flow loads from Stored Machine Learning Model to Simple Python Web Server within the Production Trust Zone is not encrypted. Altered models could be served, enabling backdoors or denial of service.	- Enable TLS/mTLS with integrity checks; verify model signatures before load - Restrict write access to the model store and enable immutability/versioning - Monitor model hash at startup and periodically

SSH (Data Flow)

Description: Transfer sensitive data from Deployment Service to Bastion

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	CI/CD to bastion bridge enabling privilege escalation into Production	Elevation of Privilege	High	Open		The SSH flow crosses from Deployment in the Experimental Trust Zone to Bastion in the Production Trust Zone. If the deployment service is compromised, attackers can pivot into production via this pathway.	- Remove direct Experimental → Production SSH; perform deployments from a controlled runner within Production or via control-plane APIs - Enforce JIT approvals, mTLS/SSH certs, and IP allowlisting - Strong segmentation and one-way artifact promotion with signature verification

update (Data Flow)

Description: Transfer sensitive data from Bastion to Stored Machine Learning Model.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Administrative updates can tamper with production models	Tampering	High	Open		The flow update from Bastion to Stored Machine Learning Model within the Production Trust Zone is unencrypted and allows direct modification of model artifacts without integrity guarantees.	- Enforce TLS/mTLS and require signed, immutable model promotions via CI/CD only - Limit bastion write access; use approval workflows and audit logs - Verify model signatures/hashes on write and before load

SSH (Data Flow)

Description: Transfer data from Infrastructure Admin to Bastion.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Brute force or credential theft enabling SSH access to Bastion	Spoofing	High	Open		The SSH flow crosses from external Infra Admin to Bastion in the Production Trust Zone. Attackers can attempt credential stuffing or use stolen keys to impersonate the admin.	- Enforce MFA-backed SSH certificates, fail2ban/connection limits, and IP allowlisting - Disable password auth; use hardware-backed keys and short-lived certs - Continuous monitoring with alerting and session recording

update (Data Flow)

Description: Transfer sensitive data from Bastion to Stored Machine Learning Model.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Unencrypted admin path allows silent model manipulation	Tampering	High	Open		The flow update from Bastion to Stored Machine Learning Model in the Production Trust Zone is not encrypted, permitting on-path alteration and lack of integrity on production models.	- Require TLS/mTLS for admin/model update channels - Only allow model updates via signed CI/CD artifacts and immutable registries - Enforce approvals and audit logging for all model changes

(Data Flow)

Description: Transfer sensitive data from Authorized Keys Storage to Bastion.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Improper key distribution to Bastion	Tampering	Medium	Open		The flow from Authorized Keys to Bastion in the Production Trust Zone, while encrypted, can still propagate outdated or unauthorized keys if distribution is not controlled and audited.	- Enforce signed key manifests and approval workflows; limit who can add keys - Automate key rotation and revocation with audit trails - Validate keys against a central CA or directory before acceptance

imports (Data Flow)

Description: Transfer data from Third Party tools and ML libraries to Simple Python Web Server.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
	Malicious runtime dependency in production web server	Tampering	High	Open		The flow imports crosses from external 3rd party tools and ML libraries into Simple Python Web Server within the Production Trust Zone. A compromised dependency can lead to code execution or data exfiltration in production.	- Use a private, signed package mirror and lockfiles; verify signatures/checksums (Sigstore) - Disallow direct internet package installs in production; deploy vetted, vendored wheels - Monitor for known vulnerabilities (SCA) and enforce policy-as-code