Cybersecurity Playbook

for Large Language Model applications

Table of Contents

1 Purpose and Scope

GovTech developed the Cybersecurity Playbook for Large Language Model ("LLM") Applications, aiming to provide guidance for agencies and organisations in procuring, developing (incl. training and fine-tuning), deploying (incl. integrating), and using LLM applications securely. The cybersecurity objectives of LLM applications revolve around safeguarding their confidentiality, integrity, and availability.

Given the dynamic nature of the cybersecurity landscape and the early stage of LLM adoption and practice, this playbook is necessarily a tentative draft. It is not obligatory for agencies to adopt. We are committed to regularly reviewing the playbook in consultation with AI practitioners to ensure its continual relevance.

However, agencies and organisations are encouraged to identify threats and implement controls relevant to their LLM applications.

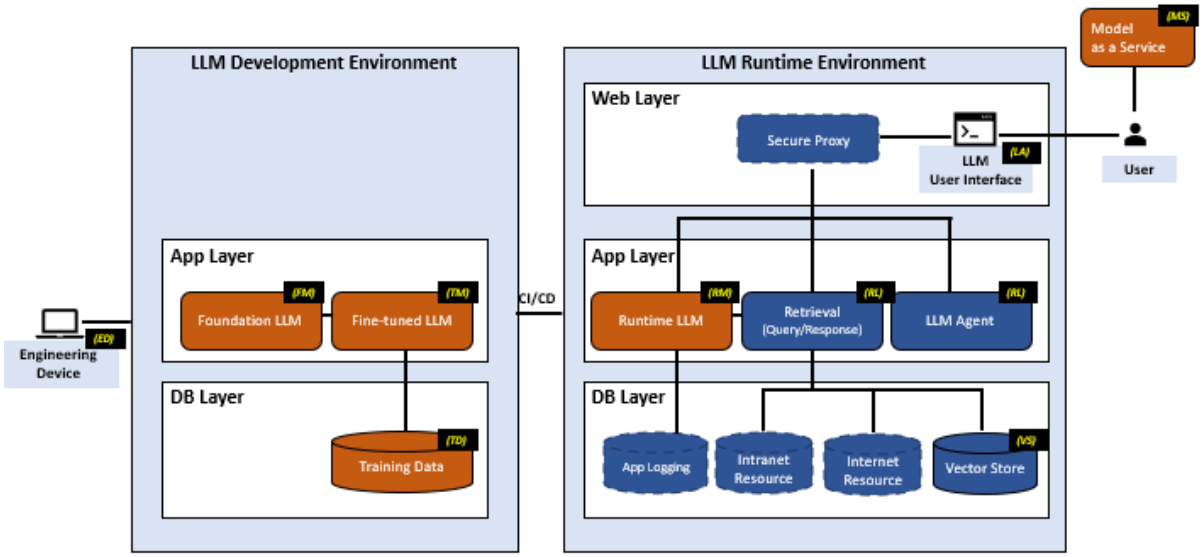
2 Development Approach

In this playbook, we delineate the abstract architecture for a generic LLM Application and incorporate threat modelling to provide a systematic and cost-effective way to identify and assess cybersecurity threats, along with corresponding mitigation controls applicable to such a generic LLM application.

3 Abstract Architecture for LLM Application

In developing our abstract architecture for generic LLM applications below, we integrated the Gartner reference architecture[1] and traditional IT components. This abstract architecture comprises ~~of~~ a development and a runtime environment. Fine-tuning of LLM using training data takes place in the development environment, while deployment and utilisation of LLM take place in the runtime environment. To comprehensively cover typical LLM use cases, we also integrated retrieval augmented generation (RAG) components and LLM agents into the abstract LLM application architecture. Note that the abstract LLM application architecture is a conceptual diagram and not a blueprint.

Figure 1- Abstract Architecture for LLM Applications



The cybersecurity controls specific to LLMs are organised by architecture components in this playbook. The architecture components of LLM applications are labelled in black shaded boxes within bracket, as shown in Figure 1.

Table 1 – Architecture components of LLM Applications.

ED	Engineering device
TD	Training data
FM	Foundation model
TM	Fine-tuned model
RM	Runtime model
VS	Vector store
LA	LLM interface application
RL	Retrieval and LLM agent

[1] Dennis Xu, Kevin Schmidt (2024, Jan 17). Gartner: Generative AI Adoption: Top Security Threats, Risks and Mitigations.

Government Agencies may refer to existing IM8 security policies for cybersecurity controls applicable to the traditional IT components of the LLM Applications, demarcated with dotted lines in Figure 1. For the same IT components, other organisations may refer NIST 800-53, CIS Benchmarks, ISO27001 and ISO27002. In addition, LLM components which are required to be run on accelerated (high performance) compute are demarcated in orange shaded boxes in Figure 1.

Additional Note on Architecture Components of LLM Applications:

Foundation LLM	*refers to a pre-trained model that can be used as a baseline for deployment, or for further fine-tuning.*
Fine-tuned LLM	*refers to LLM that has undergone training on a specific dataset or task*
Training Data	*refers to the dataset that is required for fine-tuning the LLM*
Secure Proxy	*refers to API Gateway or Application Load Balancer fronted by WAF, Message Queue, File transfer Service, etc.*
Runtime LLM	*refers to LLM that the Agency deploys and retains full control over, which can alleviate privacy concerns, unlike Model-as-a-Service LLM.*
Retrieval (Query/Response)	*responds to the user prompt and improve the accuracy of user prompt by integrating knowledge from LLM, Vector Store, Internet Resource, and Intranet Resource (e.g., knowledge base).*
LLM Agent	*refers to the intermediary application that is used to invoke third party services to perform a specific task. For example, execute a line of code.*
LLM User Interface Application	*refers to the intermediary application through which users communicate with the LLM.*
Internet Resource	*refers to whitelisted internet accessible data sources for access by the Retrieval*
Intranet Resource	*refers to whitelisted Government Enterprise Network (GEN)-accessible data sources for access by the Retrieval*
Model as a Service	*refers to machine learning model that is offered as a cloud service*
Accelerated Compute	*refers to the underlying high-performance computing infrastructure required for running LLMs and Training Data.*

4 Associated Threats for LLM Applications

We mapped the potential threats specific to LLMs, grouped by development, deployment, and utilisation stages of LLM lifecycle and outlined the details of each of the identified threats and its impact to cybersecurity objectives (i.e., confidentiality, integrity, and availability), supported by industry case studies for reference. While not explicitly mentioned here, threats applicable to traditional IT components can still apply.

Table 2 – Identified threats on LLM applications by lifecycle.

Threat ID	Threat	Threat Scenarios	Threat Impact	Industry Case Studies
LLM Lifecycle: **Development** *The development involves training and fine-tuning of the LLM using training data.*				
Dev1	***Foundation / Fine-tuned LLM poisoning*** *(data sets)*	An attacker manipulates LLM behaviour by altering training data, including test data, during development time.	Integrity	*Poison Training Data*
Dev2	***Foundation / Fine-tuned LLM poisoning*** *(engineering elements)*	An attacker manipulates LLM behaviour by altering the code, configuration or LLM parameters during development time.		*Learning to poison LLM during instruction tuning*
Dev3	***Foundation LLM poisoning*** *(transfer learning attack)*	An attacker uploads malicious LLM to third-party repository, which is subsequently downloaded by the victim during development time.		*PoisonGPT* *How to poison LLM supply chain on Hugging Face*
Dev4	***Sensitive data leakage*** *(training data)*	An attacker gains unauthorised access to training data, including test data during development time.	Confidentiality	*Arbitrary Code Execution with Google Colab*
Dev5	***Sensitive data leakage*** *(LLM parameters)*	An attacker gains unauthorised access to LLM parameters during development time.		

Threat ID	Threat	Threat Scenarios	Threat Impact	Industry Case Studies
Dev6	***Sensitive data leakage*** *(source code/ configuration)*	An attacker gains unauthorised access to source code or configuration that is used to pre-process the training data, including test data and train the LLM during development time.		*Uber Breach Exposes the Data of 57 Million Drivers and Users*
LLM Lifecycle: **Deployment** *The deployment involves deploying fine-tune LLM from development environment to runtime environment.*				
Dep1	***Runtime LLM poisoning***	An attacker manipulates the behaviour of the LLM itself or its input and/or output logic by altering the LLM parameters during deployment or utilisation phase.	Confidentiality	*Learning from Tay's introduction*
Dep2	***Runtime LLM theft***	An attacker steals LLM parameters during deployment or utilisation phase. For example, by gaining access to LLM's executables, memory, or other parameters in the deployment environment.		*Stealing part of production LLM*
Dep3	***Indirect runtime LLM behaviour manipulation***	An attacker manipulates the behaviour of the LLM by injecting indirect prompts through the compromised data source used by the LLM application, causing it to perform unintended actions.	Integrity	*Indirect Prompt Injection Threats: Bing Chat Data Pirate*
LLM Lifecycle: **Use** *The use of LLM arise through user interaction with the LLM in the runtime environment.*				
Use1	***LLM behaviour manipulation***	An attacker manipulates the behaviour of the LLM by injecting direct prompts, causing it to perform unintended actions such as gaining unauthorised privilege	Confidentiality, Integrity, Availability	*ChatGPT Plugin Privacy Leak* *Achieving Code Execution in MathGPT via Prompt Injection*

Threat ID	Threat	Threat Scenarios	Threat Impact	Industry Case Studies
		access to restricted parts of a system, altering system configurations or critical data records, and unauthorised purchases or fund transfers where LLM can access to an e-commerce system or financial database.		*Tay Poisoning* *LLMjacking: Stolen Cloud Credentials Used in New AI Attack*
Use2	***Sensitive training data disclosure***	An attacker manipulates the behaviour of the LLM by injecting direct prompts, causing it to leak sensitive data from the private training set.	Confidentiality	*Extracting Training Data from ChatGPT* *Why system prompt leaks are bad*
Use3	***Sensitive in-context learning data disclosure***	An attacker manipulates the behaviour of the LLM by injecting direct prompts, causing it to leak rich context learning data with sensitive data from non-user input.		
Use4	***Unintended sensitive data disclosure***	A user provides sensitive data to an LLM service, unaware that the LLM utilises the provided sensitive data to train itself. Consequently, leading to leakage of this sensitive data to other users of the LLM service.		*Samsung Workers Accidentally Leaked Trade Secrets via ChatGPT*
Use5	***LLM theft***	An attacker collects inputs and outputs of target LLM and uses those combinations to train a replica LLM.	Confidentiality	*GPT-2 Model Replication* *Stanford researchers make a new ChatGPT with less than $600*
Use6	***LLM Service Disruption***	An attacker provides high frequency, voluminous or malicious inputs to the LLM, with aim to cause denial of service.	Availability	*Energy Latency Attacks On Neural Network*

5 Recommended controls and corresponding threats for LLM applications

Table 3 outlines the recommended controls to mitigate corresponding threats, grouped by applicable architecture components. These controls are not intended for compliance purposes. Refer to the controls for architecture components that are relevant for specific LLM application.

Table 3 – Recommended controls to mitigate corresponding threats.

Control ID	Control Description	Threat ID (s)	References
Architecture Component: ***Engineering Device (ED)***			
ED-C1	**Harden Engineering Devices (Endpoints)** Perform engineering activities only through Government Standard Image Build (*GSIB*[2]) or equivalent; or device onboarded to Security Suite for Engineering Endpoint Devices (*SEED*[3]) to mitigate the risks of security vulnerabilities exploitation through insecure engineering endpoints. Non-government organisations could implement similar requirements for hardened endpoints for developers and accompanying identity and access management solutions in order to secure the development environment.	*Dev1 to Dev 3: Foundation/ Fine-tuned LLM poisoning* *Dev4 to Dev 6: Sensitive data leakage*	*IM8*
ED-C2	**Use Trusted LLM Development Tools** Use development tools offered on Government		*Supply Chain Management*

[2] **GSIB**: Government standard image build (GSIB) is a standard operating environment (SOE) device issued by the government. https://docs.developer.tech.gov.sg/docs/techbiz-documentation/glossary.
[3] **SEED**: Security Suite for Engineering Endpoint Devices (SEED) is the Singapore Government's implementation of an Identity and Access Management (IAM) and zero trust platform, ensuring the security of engineering resources against unauthorized access to SGTS services.
https://www.developer.tech.gov.sg/products/categories/cybersecurity/seed/overview.html

Control ID	Control Description	Threat ID (s)	References
	Commercial Centre (*GCC*[4]) as Software-as-a-Service (SaaS) (e.g., *AIBots*[5], LLM Stack[6], *MAESTRO*[7], *SHIP/HATS*[8], etc.) rather than third-party tools to mitigate the risks of security vulnerabilities exploitation through insecure development tools. Non-government organisations could implement similar secure continuous integration / continuous deployment tools.		
Architecture Component: ***Training Data (TD)***			
TD-C1	**Control Access to Training Data and underlying Accelerated Compute** Grant user access to the training data and underlying accelerated compute based on the principle of least privilege to mitigate the risks of unauthorised access to them.	*Dev4 to Dev5: Sensitive data leakage*	*AML.M0019* *IM8*
TD-C2	**Right Classify Training Data and underlying Accelerated Compute** Ensure that the training data stored on accelerated compute are aligned in their classification to mitigate the risks of data exfiltration and to properly assess its impact. For example, accelerated compute classified as	*Dev4 to Dev5: Sensitive data leakage*	*IM8*

[4] **GCC**: Government on Commercial Cloud (GCC) platform to provide government agencies with a standardised approach to adopting commercial solutions offered by cloud service providers. https://www.tech.gov.sg/products-and-services/for-government-agencies/software-development/government-on-commercial-cloud/.
[5] **AI Bot**: A platform where users can create their own RAG AI chatbots and share with others, quickly and easily. https://aibots.data.tech.gov.sg/.
[6] **LLM Stack**: A platform for quick prototyping and production launch of LLM-powered applications. https://apps.stack.govtext.gov.sg/.
[7] **MAESTRO:** (Machine learning & Artificial intelligence Enterprise-level Secure Tool suite for Reliable Operations) MAESTRO is a centralised WOG data platform offering a comprehensive suite of tools and services, with scalable compute resources. https://www.developer.tech.gov.sg/products/categories/platform/maestro/overview.html.
[8] **SHIP-HATS**: SHIP-HATS is the CI/CD component within the Singapore Government Tech Stack. It integrates DevSecOps best practices into product development cycles for improved productivity. https://www.developer.tech.gov.sg/products/categories/devops/ship-hats/overview.html.

Control ID	Control Description	Threat ID (s)	References
	Restricted should not store training data classified beyond Restricted.		
TD-C3	**Store Sensitive Training Data Securely** Store and encrypt training data that are sensitive[9], in GCC or on-premise based infrastructure to mitigate the risks of data exposure.	*Dev1 to Dev3: Foundation/Fine-tuned LLM poisoning* *Dev4 to Dev5: Sensitive data leakage*	*Segregate Data* *IM8*
TD-C4	**Sanitise Training Data** Sanitise training data classified as Sensitive-Normal and above, using tools such as *Cloak*; or apply other privacy enhancing technologies such as differential privacy[10] or synthetic data generation[11] (using tool like *Mirage*) to mitigate the risks of data exposure and limit its impact.	*Dev4: Sensitive data leakage (Training data)*	*Segregate Data* *AML.M0007*
TD-C5	**Maintain Provenance of Training Data and Validation data** Record metadata of the training data such as author, origin, date, and time of creation, and maintain a record of changes to the metadata, to mitigate the risks associated with inadequate incident handling.	*Dev1: Foundation/ Fine-tuned LLM Poisoning (Data sets)*	*Data Provenance and Lineage*
Architecture Component: ***Foundation LLM (FM)***			
FM-C1	**Verify Integrity of Foundation LLM** Verify the integrity of the foundation LLM's files	*Dev3: Foundation LLM Poisoning*	*NIST 800-53 SI-7*

[9] **Sensitive** refers to information or data that if disclosed, accessed, or handled improperly has potential consequential impact on finance, reputation, operations, privacy, legal matters, etc.
[10] A mathematical framework for ensuring the privacy of individuals in datasets.
[11] The process of creating artificial data that mimics the statistical properties of real-world data.

Control ID	Control Description	Threat ID (s)	References
	using methods like file hashes, digital signatures, or checksums; to mitigate the risks associated with tampered LLMs.		
FM-C2	**Use Benchmarked Foundation LLM** Prefer the use of supplier/vendor-furnished foundation LLM that has undergone academic or industry benchmarking, demonstrating its robustness and security to mitigate the risk of security vulnerabilities exploitation.	*Dev1 to Dev2: Foundation/ LLM Poisoning* *Dev4 to Dev5: Sensitive data leakage* *Dep1 to Dep2: Runtime LLM Poisoning/ Theft*	*Cataloguing LLM Evaluation*
FM-C3	**Maintain Provenance of Foundation LLM** Record metadata of foundation LLM used, such as author, origin, date, and time of creation, and maintain a record of changes to metadata to mitigate the risks associated with inadequate incident handling.	*Dev3: Foundation LLM Poisoning*	*NIST 800-53 AU-6*
Architecture Component: ***Fine-tuned LLM (TM)***			
TM-C1	**Control Access to Fine-tuned LLM and underlying Accelerated Compute** Grant user access to the fine-tuned LLM and underlying accelerated compute based on the principle of least privilege to mitigate the risks of unauthorised access to them.	*Dev5: Sensitive data disclosure (LLM parameters)* *Dep1 to Dep2: Runtime LLM Poisoning/ Theft* *Dep3: Indirect runtime LLM behaviour manipulation* *Use1:*	*AML.M0019*

Control ID	Control Description	Threat ID (s)	References
		LLM behaviour manipulation	
TM-C2	**Right Classify Fine-tuned LLM and underlying Accelerated Compute** Ensure that the training data, fine-tuned LLM and underlying accelerated compute are aligned in their classification to mitigate the risks of data exfiltration and to properly assess its impact. For example, accelerated compute classified as Restricted should not store training data and fine-tuned LLM classified beyond Restricted.	*Dev4: Sensitive data disclosure (Training data)*	*NIST 800-53 AC-3*
TM-C3	**Save Fine-tuned LLM in Secure Formats** Save fine-tuned LLM in known secure formats (e.g., GGML, GGUF[12], etc.) and avoid the use of insecure formats (e.g., pickle file[13]) to mitigate the risks of data corruption, unauthorized access, and security vulnerabilities exploitation.	*Dev1 to Dev2: Foundation / Fine-tuned LLM poisoning* *Dev4 to Dev5: Sensitive data disclosure* *Dep1 to Dep2: Runtime LLM Poisoning/ Theft*	*Python Serialization Vulnerabilities - Pickle* *AML.M0016* *AML.T0018* *Secure Development Program*
TM-C4	**Employ Adversarial Testing on Fine-tuned LLM** For LLM Applications with medium risk and above, employ adversarial testing utilising tools such as *Garak* and *Giscard*, to mitigate the risk of security vulnerabilities exploitation.	*Dep1 to Dep2: Runtime LLM Poisoning/ Theft* *Use2: Sensitive data disclosure*	*AML.M0003* *Generative Adversarial Reward Modelling*

[12] GGML and GGUF are **GPT-Generated Model Language** and **GPT-Generated Unified Format** by Georgi Gerganov.

[13] **Pickle** is a Python library used for serializing and deserializing Python objects. While it's a convenient tool for storing and retrieving data structures, it's not considered a secure format for storing Large Language Models (LLMs) or any sensitive data.

Control ID	Control Description	Threat ID (s)	References
		Use5 to Use6: LLM Theft/ Service Disruption	
TM-C5	**Use SHIP-HATS or equivalent DevSecOps tools to integration and deployment of LLM** Use *SHIP-HATS* or equivalent DevSecOps tools for integration and deployment of LLMs that covers the following: i. Software Composition Analysis (SCA), Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), Container Image Scanning, etc. in the LLM Application's Continuous Integration (CI) pipeline to mitigate the risk using insecure dependencies and security vulnerabilities exploitation due to inadequate patches and misconfiguration. ii. Code signing in the LLM's Continuous Deployment (CD) pipeline to mitigate the risks of using tampered LLM Applications. iii. Disallow changes on LLM to flow from runtime to development environment to mitigate the risks of unauthorised changes. Non-government organisations could implement equivalent DevSecOps tools to provide the features stated in i., ii. and iii.	*Dep1 to Dep2: Runtime LLM poisoning & theft*	*AML.M0016* *Secure Development Program* *Supply Chain Manage*
Architecture Component: ***Runtime LLM (RM)***			
RM-C1	**Control Access to Runtime LLM** Grant user access to the runtime LLM based on the principle of least privilege to mitigate the risks of unauthorised access to it.	*Dep1 to Dep2: Runtime LLM Poisoning/ Theft*	*LLM Access Control* *AML.M0001* *Weight and Bias*

Control ID	Control Description	Threat ID (s)	References
RM-C2	**Encrypt Runtime LLM Artefacts in Transit and at Rest** Encrypt runtime LLM artefacts in transit (e.g., using https) to mitigate the risks of data exposure and limit its impact.	*Dep1 to Dep2: Runtime LLM Poisoning/ Theft*	*AML.M0012*
RM-C3	**Access only Whitelisted LLM Agent(s)** Ensure that the runtime LLM is only able to access whitelisted LLM Agent(s) to mitigate the risk of manipulation of LLM behaviour.	*Dep1 to Dep2: Runtime LLM Poisoning/ Theft*	*NIST 800-53 AC-6*
Architecture Component: ***Model-as-a-Service (MS)***			
MS-C1	**Disable Interaction History to MaaS** Disable interaction history on MaaS, to mitigate the risk of data exposure on it.	*Use4: Sensitive data disclosure*	*NIST 800-53 R5 AC3, AC6*
Architecture Component: ***Vector Store (VS)***			
VS-C1	**Harden Vector Store** Ensure vector store is hardened to mitigate the risks of security vulnerabilities exploitation due to misconfigurations.	*Dep1 to Dep2: Runtime LLM Theft & behaviour manipulation*	*NIST 800-53 CM-6*
VS-C2	**Control Access to Vector Store** Grant user access to vector store based on the principle of least privilege to mitigate the risk of unauthorised access.		*NIST 800-53 AC-6*
VS-C3	**Encrypt Vector Store** Encrypt all indexed and embedded data stored in vector store to mitigate the risk of data exposure and limit its impact.	*Dep2: Runtime LLM behaviour manipulation*	*NIST 800-53 SC-13*

Control ID	Control Description	Threat ID (s)	References
VS-C4	**Sanitise Inputs to Vector Store** Remove sensitive information from any documents before converting the documents to embeddings for storage in the vector store to mitigate the risk of data exposure and limit its impact.	*Dep2: Runtime LLM behaviour manipulation* *Use3: Sensitive in context learning data disclosure*	*NIST 800-53 PL-2*
Architecture Component: ***LLM User Interface Application (LA)***			
LA-C1	**Apply Rate Limiting on User Inputs using LLM User Interface Application** Apply rate limiting on user inputs via LLM user interface application to mitigate the risk of disruption of service on the LLM Applications.	*Use1 to Use2: Runtime LLM Theft & behaviour manipulation* *Use4 to Use6: Sensitive Data Disclosure*	*NIST 800-53 AC-3*
LA-C2	**Validate User Inputs to LLM User Interface Application** Validate user inputs (e.g., using prompt injection detection systems, web application firewall, etc.) via LLM user interface application to mitigate the risk of prompt injection on LLM and thus manipulating its behaviour.	*Dep3: Indirect runtime LLM behaviour manipulation* *Use1 to Use2: Runtime LLM Theft & behaviour manipulation* *Use4 to Use6: Sensitive Data Disclosure*	*Prompt Input Validation* *Prompt Injections*
LA-C3	**Validate Responses from Runtime LLMs before Displaying**	*Dep3: Indirect runtime*	*Validate ML*

Control ID	Control Description	Threat ID (s)	References
	Validate responses from runtime LLMs and foundation LLM before displaying them to users to mitigate the risk of data exposure. For example, verify that output from LLM does not contain unintended sensitive information.	*LLM behaviour manipulation* *Use1: LLM behaviour manipulation*	*Model*
LA-C4	**Conduct Regular Vulnerability Assessments and Penetration Tests** Conduct vulnerability assessments every quarter and penetration test every year on LLM user interface application to mitigate the risks of security vulnerabilities exploitation.	*Dep1 to Dep2: Runtime LLM poisoning & theft*	*NIST 800-53 CA-2*
Architecture Component: ***Retrieval and LLM Agent (RL)***			
RL-C1	**Access only Whitelisted Internet Resource** Ensure that agency-owned retrieval and LLM agent accesses only whitelisted Internet-accessible resources to mitigate the risk of indirect prompt injection within untrusted Internet resources.	*Dep3: Indirect runtime LLM behaviour manipulation* *Use1 to Use6: LLM behaviour manipulation, sensitive data disclosure, Theft and Service Disruption*	*NIST 800-53 AC-6*
RL-C2	**Access only Whitelisted Intranet Resource** Ensure that the agency-owned retrieval and LLM agent accesses only whitelisted GEN-accessible resources (e.g. knowledge base) to mitigate the risk of indirect prompt injection and sensitive data exposure.	*Dep3: Indirect runtime LLM behaviour manipulation* *Use1 to Use6: LLM behaviour manipulation, sensitive data disclosure, Theft and Service Disruption*	*NIST 800-53 AC-6*

Control ID	Control Description	Threat ID (s)	References
RL-C3	**Validate Content from Internet and Intranet Resources** Ensure that the agency-owned retrieval and LLM agent validates content from Internet and intranet resources before processing them to mitigate the risk of indirect prompt injection and manipulation of LLM behaviour.	*Dep3: Indirect runtime LLM behaviour manipulation* *Use1 to Use6: LLM behaviour manipulation, sensitive data disclosure, Theft and Service Disruption*	*NIST 800-53 AC-17*