Canonical

Canonical Group Limited
United Kingdom

NCSC UK TSA Security Declaration

This document provides Canonical's Security Declaration in response to the Code of Practice related to the published UK Telecommunications Security Act Code of Practice.

It is intended to describe processes and measures implemented by Canonical related to security and engineering best practices to ensure that Canonical supports and complies with the requirements of the Telecommunications Security Act for the Telecommunications Sector within the UK.
This document is also intended to be used for customers and prospective customers for the purpose of the Vendor Security Assessment according to the requirements of the UK Telecommunications Security Act Code of Practice.

DocuSigned by:
Neil French
2FA3519341EF47C...

Neil French
COO & Director of Canonical Group Limited

Canonical

Canonical Group Limited
United Kingdom

#	VSA Criteria	Description	Canonical Response
V.A: Product Lifecycle Management			
1	V.A.1: Product lifecycle process	The vendor clearly identifies the lifecycle for each product. Vendors should have an End of Life Policy which covers details on how long the products will be in general support, extended support, and supported after End of life sales dates.	Canonical publishes new releases of products on a regular cadence, usually every 6 months with the possibility of yearly releases, depending on the product. From time to time, a LTS or 'Long Term Support' releases are published (e.g. every two years in April in the case of Ubuntu). LTS releases are the 'enterprise grade' releases of Canonical products for which support is provided to a defined period of time. End-of-Life dates are defined and communicated in Canonical public channels to customers and are usually defined from the release of the product version. For non-LTS releases, support is provided during the period that the product is active and not end-of-life. Support for non-LTS releases are not covered by the same commitment as LTS releases and could be limited to a set of packages, depending on the product and/or the support offering chosen by customers. For Long Term Support releases, after the LTS period is over, the product is considered end of life and support is discontinued and only avaibale through a possible specific agreement between Customers and Canonical.
2	V.A.2: Software maintenance	Each product is maintained through its published life cycle. This maintenance, as a minimum, covers security fixes for the product.	Canonical supports products from their release until the end-of-life period defined or communicated by Canonical. Support offerings vary depending on the Product release (e.g. LTS versus non-LTS) and the level of support services covered by the agreement between Canonical and its customers. Canonical supports security fixes for all products under the shelf-life or all products. Canonical suppor offerings are available at: https://ubuntu.com/support

Canonical

Canonical Group Limited
United Kingdom

3	V.A.3: Software version control	Each product has a version-controlled code repository which logs every code modification. This audit log will detail: what code has been modified, added, or removed; why the change was made, who made the change; when the change was made; and which version of the code has been built into the released product.	Canonical-managed projects use version control to maintain a full version history of changes made to their codebases
4	V.A.4: Software releases	Each product goes through a rigorous software release cycle including internal testing before a version is released for general availability. Software will not be released if it does not comply with the Secure Engineering requirements detailed below. Each product should have regular external testing carried out on it by an independent third party.	Canonical's Release cycle includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes internal testing before a version is released for general availability.
5	V.A.5: Development processes and feature development	There is one primary release train of the product. Forking of new versions is minimised. Where necessary, customer-specific functionality is provided as optional modules. Any new features are brought into the main product line during the standard development roadmap.	Canonical's Release cycle includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes minimizing forking of versions and adding functionality during the roadmap development only.
6	V.A.6: International release and forking	The vendor maintains a single, global version line for each product. There are a minimal number of other versions (ideally none).	Canonical maintains global version lines for products, where versions are kept to a minimum necessary. If other versions are maintained, this is specified.
7	V.A.7: Use of tools, software and libraries	Third party tools (e.g. code compilers), software components and software libraries that are used within the product are inventoried. Any of the above that are material to the security of the vendor's software are maintained throughout its lifetime	Canonical inventories components and libraries used in products and maintains all relevant components and libraries that are relevant from a Security perspective.
8	V.A.8: Software Documentation	The vendor provides up-to-date and technically accurate documentation alongside new releases of the product. This documentation, as a minimum, shall detail how to securely configure, manage, and update the product.	Canonical maintains product documentation on its website for all top-level products such as OpenStack, Kubernetes, Juju, MAAS and Ubuntu. Documentation is also contributed to the respective upstream projects for products such as Openstack, Kubernetes, Kubeflow, Ceph and OSM.

Canonical

Canonical Group Limited
United Kingdom

V.B: Product Security Management			
9	V.B.1: Security culture	The vendor has a security culture which ensures that security principles are followed.	Security Culture at Canonical is primarily owned by the CISO which implements policies, processses, procedures and controls for Security Awareness, Secure Code Development Vulnerability Management and Incident Management accross all products.
10	V.B.2: Secure Development Lifecycle	The vendor has a Secure Development Lifecycle to embed security into product development. All development teams follow, and can evidence that they follow, the Secure Development Lifecycle processes.	Canonical follows a SDLC process as part of the development and release process established internally. Our process is aligne with NIST SSDF / NIST 800-218.
11	V.B.3: Internal component management	Any shared internal components or libraries are kept up to date and only the latest stable, supported version is used. These components and libraries are not be modified for specific builds and are supported for the lifetime of the product.	All internal components that make Canonical products are managed and maintained as part of the Release Process
12	V.B.4: External component management	Only supported external components are used within a product. The vendor monitors the external component's changelog so that only the latest supported, stable version is used within the product. Additionally, the vendor monitors the external component's security advisories and pulls in any security fixes and integrates them into their product with a security update.	All external components that make Canonical products are supported or managed and maintained internally as part of the Release Process
13	V.B.5: Unsafe Functions	There are no unsafe functions used within the vendor's released code. Unsafe functions are those commonly associated with security vulnerabilities or those considered unsafe by industry best practice.	Use of unsafe functions forms part of the main inclusion request process as part of any security review undertaken. Code security analysis tools are used for this review.
14	V.B.6: Redundant and duplicate code	The vendor's source tree is maintained to a level that there is limited redundant or duplicate code.	Canonical's software development process adheres to best practices that ensure the source tree is optimized with minimal redundant or duplicate code. We employ rigorous code reviews, software compostion analysis, automated testing, and continuous integration practices to maintain code quality and efficiency.

Canonical

Canonical Group Limited
United Kingdom

15	V.B.7: File structure	The vendor's source tree is maintained to a level where code complexity is minimised, and functions perform single, clear, actions.	Canonical's software development process adheres to best practices that ensure the source tree is optimized with minimal redundant or duplicate code. We employ rigorous code reviews, software compostion analysis, automated testing, and continuous integration practices to maintain code quality and efficiency.
16	V.B.8: Debug functionality	There is no engineering debug functionality present within the vendor's released products that could weaken or bypass the product's security mechanisms.	Debug functionality exists in some Canonical products. This is feature is disabled by default in telecom operators' installations.
17	V.B.9: Comments	The source tree has suitable and understandable comments through it, explaining what the code is for and how it performs its actions.	Commenting is encouraged in areas of code complexity in Canonical-managed projects. Levels of commenting may vary and might not always follow the same standards for all products.
V.C: Protected development and build environments			
18	V.C.1: Segregation of development environment	Development environment is segregated from the corporate network and protected from the internet.	Whenever feasible and possible, segregation of the 'development environment' is done.
19	V.C.2: Segregation of build environment	Build environment is segregated from the corporate network and protected from the internet. Very few people can make changes.	Canonical's build environments are hosted on a dedicated infrastructure which is isolated from other parts of Canonical's network infrastructure and are administered by a dedicated team.
20	V.C.3: Build automation	Build environments are simple, and the build process is automated.	Canonical has an automated build process. Debian packages make use of a managed build secure root image; Debian source packages detail dependencies required to build the package, all of which must be sourced from the Ubuntu or Ubuntu Cloud Archive repositories. Build environments are easily reproduced using developer tooling to reproduce build failures. Snaps build against an Ubuntu Core image using the snapcraft build tool; dependencies are detailed in the definition of the snap (snapcraft.yaml). For both distribution methods, developers can easily reproduce the build process outside of the secured build environment used for build and publication of release packages.

Canonical

Canonical Group Limited
United Kingdom

21	V.C.4: Role-based access	Only individuals with a need have access to the internal code base, and access is controlled and limited based on role	RBAC is enforced in multiple ways for Canonical's products; Upstream projects typically use VCS repositories with core developers having commit privileges; other projects may also not allow direct commits, requiring pull requests or reviews which are peer reviewed before approval subsequently committed automatically into the VCS repository. Access is controlled based on need-to-know principle and restricted to appropriate teams. For Canonical-managed projects and product only Canonical employees have access to commit/merge code.
22	V.C.5: Code review	All code is independently reviewed prior to acceptance. Feedback processes exist.	Canonical's SDLC process enforces peer-reviews prior to acceptance. Feedback is provided in the pull-request/merge proposal and code is re-submitted for approval, if needed.
23	V.C.6: Repeatable builds	All builds of released software must be repeatable at a future date	All builds have verbose logs, detailing all of the components used in a build. This includes versions, hashs, and source of files
V.D: Exploit mitigations			
24	V.D.1: Heap Protections	The vendor makes use of modern heap protection mitigations to help prevent heap-based memory corruption attacks against the product.	Canonical's products code are compiled/interpreted using modern heap mitigations. Implementation varies from Product to Product based on compliler and interpreters used."
25	V.D.2: Stack Protections	The vendor only ships executable code that has been compiled using modern stack mitigations.	Canonical's products code are compiled/interpreted using modern stack mitigations. Implementation varies from Product to Product based on compliler and interpreters used.
26	V.D.3: Data Execution Prevention	The vendor supports hardware-enforced data execution prevention for example DEP.	Data Execution Prevention is used on Canonical products. Implementation varies from Product to Product based on compliler and interpreters used.
27	V.D.4: Address Space Layout Randomisation	The vendor only ships executable code that has been compiled using modern ASLR techniques.	Address Space Layout Randomisation is used on Canonical products. Implementation varies from Product to Product based on compliler and interpreters used.
28	V.D.5: Memory mapping protections	The vendor's product will have no memory pages mapped by default as both "Writable" and "Executable". This excludes areas of the code required to do Just-In-Time code compilation.	The memory protections in place while Canonical product is in use is dependent on the hardware and environment it is deployed in.

Canonical

Canonical Group Limited
United Kingdom

29	V.D.6: Least Privilege code	The vendor follows a “least privilege” methodology when developing and executing code within their products. The vendor ensures that their product only runs at or requests the minimum privilege level required for it to fulfil its advertised purpose. If higher privilege levels are ever required, then the product only elevates privilege for the specific task.	Canonical products follow a “least privilege” methodology when developing and executing code within their products.
30	V.D.7: Secure execution environment	The vendor has a product road map detailing when and how they plan to implement secure execution environments to enable execution of sensitive workloads on untrusted hardware.	Canonical products already delivers secure execution environments. Also, the telco edge setup with MAAS, LXD and MicroK8s running on Ubuntu Core can provide customers with confinement based on Snaps to work with untrusted workloads.
V.E: Secure Updates and Software Signing			
31	V.E.1: Software and firmware signing	Vendor’s software and firmware is digitally-signed.	Canonical's products are digitally-signed. For Ubuntu, packages in Ubuntu and the Ubuntu Cloud Archive are not directly signed. Instead the archive maintains indexes of packages and checksums in signed files which can be used by Ubuntu installations to verify the integrity of the packages downloaded for install or update.
32	V.E.2: Signature verification	Software signatures are verified before binaries are executed.	Canonical applies software signature verification before executing binaries. By default any signature verification will result in packages not being installed.
33	V.E.3: Secure update	Updates are delivered via a secure channel that is mutually authenticated.	Updates to Canonical products are delivered via secure channels.
34	V.E.4 Downgrade protection	Built-in detection capabilities alert whenever software is downgraded during an install process.	Any attempt to downgrade a package to an older version results in a warning and a prompt to the operator.
V.F: Hardware roots of trust and secure boot			
35	V.F.1: Hardware root-of-trust	The equipment contains a hardware root-of-trust for identity and storage.	Not applicable as Canonical does not provide any hardware products.
36	V.F.2: Secure Boot	Each product will support a secure boot process, initiated by the hardware root-of-trust (V.F.1) to bring the equipment into a known-good state on restart.	Not applicable as Canonical does not provide any hardware products. Whenever needed, Canonical's products can help customers implement this requirements
37	V.F.3: Securing JTAG	Each compute element on product will will have debug interfaces (such as JTAG and UART) access disabled	Not applicable as Canonical does not provide any hardware products.

Canonical

Canonical Group Limited
United Kingdom

V.G: Security Testing			
38	V.G.1: Automated testing	Once developed, extensive security tests are automatically run against all versions of applicable products.	Canonical's Canonical's Release process includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes automated and non-automated Security Tests.
39	V.G.2: Testing rigour	Developers cannot modify the build environment to hide or disregard build issues, or issues detected by automated tests. Failing builds are automatically rejected. Therefore, code does not create any compiler errors or security related warnings during build.	Canonical's Canonical's Release process includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes automatic rejection of failed builds.
40	V.G.3: Security Testing	Security functionality is tested to demonstrate correct operation.	Canonical's Canonical's Release process includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes testing of functionality, including security fucntionality, to assure correct operation.
41	V.G.4: Negative testing	Extensive negative testing is performed against every product release, including a wide range of potential failure cases, inappropriate message sequencing and malformed messages.	Canonical's Canonical's Release process includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes negative testing.
42	V.G.5: Fuzzing	Fuzzing is performed against the product, especially focusing on interfaces which cross security boundaries. The approach is sophisticated enough to ensure that a high proportion of code is tested.	Canonical does not perform fuzz testing.
43	V.G.6: External testing	External security research teams perform testing against a selection of major product releases. Some of this testing is un-scoped.	Canonical's Canonical's Release process includes a process adherent to the NIST SSDF framework / NIST 800-218. This includes periodic testing and assessments of Product security by external third parties.
44	V.G.7: Dynamic application security testing (DAST)	The vendor has a DAST solution integrated into the vendor's test process	Software Composition Analysis and Static Application Security Testing are mandatory as part of the Release process, while Dynamic Application Security Testing is optional for Product teams to include in their testing.
V.H: Secure management and configuration			

Canonical

Canonical Group Limited
United Kingdom

45	V.H.1: Product hardening	The product can be easily hardened into a secure configuration. Documentation exists to help customers perform this hardening process. Alerts are created should the device be taken out of the hardened State.	Canonical maintains product documentation on its website for all top-level products such as OpenStack, Kubernetes, Juju, MAAS and Ubuntu. Documentation is also contributed to the respective upstream projects for products such as Openstack, Kubernetes, Kubeflow, Ceph and OSM.
46	V.H.2: Protocol Standard-isation	The product can be configured to only use standardised protocols.	No proprietary protocols are used as part of Canonical operated open source projects.
47	V.H.3: Management plane security	By default, the product is configured to only use modern, secure protocols on the management plane.	Canonical follows best practices with regards to TLS protocols and cipher suites.
48	V.H.4: Management access	Access to the management plane is user-based and supports Asymmetrickey-based (e.g. X.509 certificates or SSH keys).	Canonical's products related to management plane (MAAS and OpenStack) follow user-based access and can be LDAP integrated.
49	V.H.5: No unencrypted protocols	Secure protocols are used whenever possible (e.g. SSH and HTTPS). If an unencrypted protocol is enabled, and a secure alternative exists, the product warns the administrator, and provides the option to create a security alert.	Secure protocols are used whenever possible in Canonical solutions.
50	V.H.6: No un-documented administrative mechanisms	The product does not have any undocumented administrator accounts. Examples include, but are not limited to, hard coded passwords, access key pairs (SSH keys) or otherwise administrative access tokens.	Canonical products have no undocumented administrative mechanisms which can be easily confirmed based on the open source nature of products.
51	V.H.7: No un-documented administrative features	The product does not have any undocumented administration features	Canonical products have no undocumented administrative mechanisms which can be easily confirmed based on the open source nature of products.
52	V.H.8: No default credentials	No default passwords are left on the device after the initial set up. For clarity, this also means there are no administrative accounts coded into the vendor's software.	Canonical's products either are shipped with no default credentials or provide fucntionality to disable credentials are created as part of an initial setup/deployment.
53	V.H.9: Best Practice Guidance	The vendor is explicit about the threats to the equipment that they have sought to mitigate, and those they have not. The vendor provides detailed configuration and notes on how the equipment can be protected in networks.	Canonical maintains product documentation on its website for all top-level products such as OpenStack, Kubernetes, Juju, MAAS and Ubuntu. Documentation is also contributed to the respective upstream projects for products such as Openstack, Kubernetes, Kubeflow, Ceph and OSM.

Canonical

Canonical Group Limited
United Kingdom

V.J: Vulnerability and Issue Management			
54	V.J.1: Issue tracking and remediation	The vendor has a process for issue remediation. This ensures the vulnerability is resolved in all impacted products. Vulnerabilities are patched within appropriate timeframes.	Canonical has a process for issue remediation which is public and transparent. Security notices are released for Products.
55	V.J.2: Issue comprehension.	For issues, the vendor identifies the root cause analysis of the issue and is able to detail the origin of the vulnerability.	Canonical's process to Vulnerability Management prioritizes issues based on their criticality. For issues categorized as High or Critical, Canonical strives to identify the root cause and to provide detailed origin of the vulnerability, whenever feasible and possible. For Medium, Low or Negligible issues, that process is optional.
56	V.J.3: Vulnerability reporting	The vendor provides a publicly advertised route for disclosure of security issues that links into their vulnerability management process.	The Vulnerability reporting process is public and available at Canonical's website
57	V.J.4: Issue transparency	The vendor is transparent about their patching of security issues.	Canonical's process is transparent about patching of security issues, which is handled by the Vulnerability Management process. Canonical's Security and Engineering teams manage the triaging, patching and updating of latest public vulnerabilities from various sources and update information on Product Documentation and/or websites.
58	V.J.5 Product Security Incident Management Team (PSIRT)	The vendor has set up the PSIRT structures within its organisation	Canonical has a PSIRT team structured and organized.