Introduction aux systèmes GNU/Linux

Module IntroLinux – S26E02

Philippe Latu / Université de Toulouse
inetdoc.net

Objectif DevOps

De l’idée à la production.

→ Un flux continu de valeur !

Créer un pipeline fluide et continu de la conception à la mise en production, en mettant l’accent sur la livraison rapide de valeur.

Le plan

- Virtualisation et conteneurs
- Logiciel libre, licences et IA
- Développement open source
- DevOps et CI/CD
- Distributions : le cas Debian

Virtualisation et conteneurs

Des machines virtuelles aux conteneurs : partager le matériel, puis le noyau.

Distinguer émulation et virtualisation

Émulation sans accélération

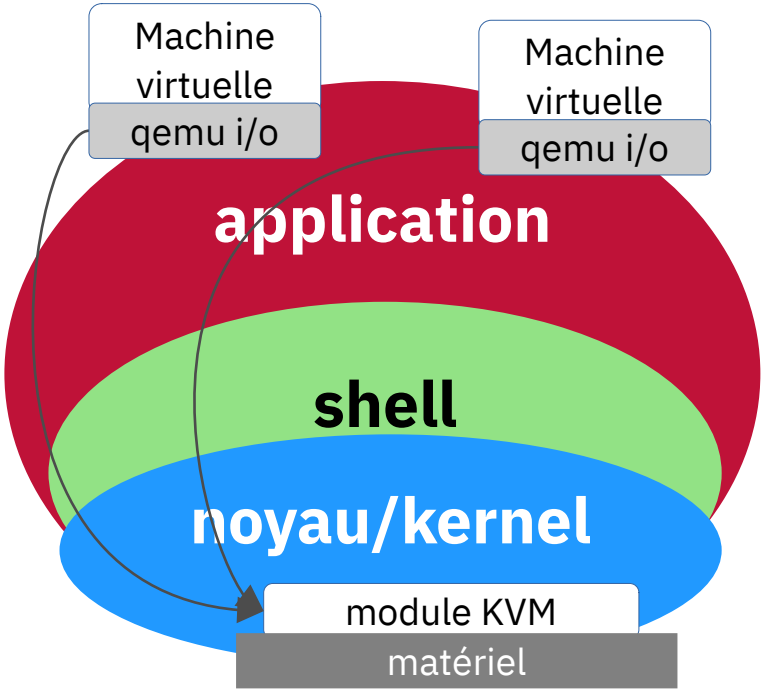
- Matériel simulé par QEMU (TCG)
- Performances limitées
- Utile pour une autre architecture

Virtualisation assistée

- Extensions VT-x / AMD-V, module KVM
- Performances proches du matériel natif

Paravirtualisation des E/S

- Pilotes virtio dans le système invité



Définir un hyperviseur

Type 1

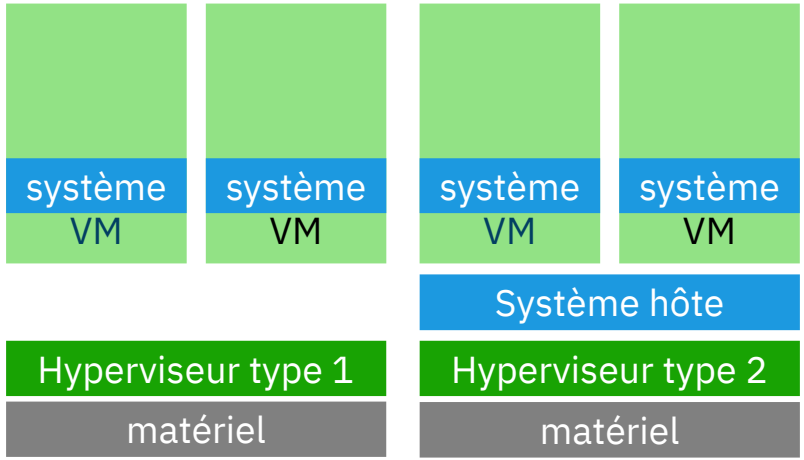
- S’exécute directement sur le matériel
- ESXi, Xen, Hyper-V, KVM

Type 2

- Application d’un système hôte
- VirtualBox, VMware Workstation

Le cas de KVM

- Le noyau Linux devient l’hyperviseur → type 1



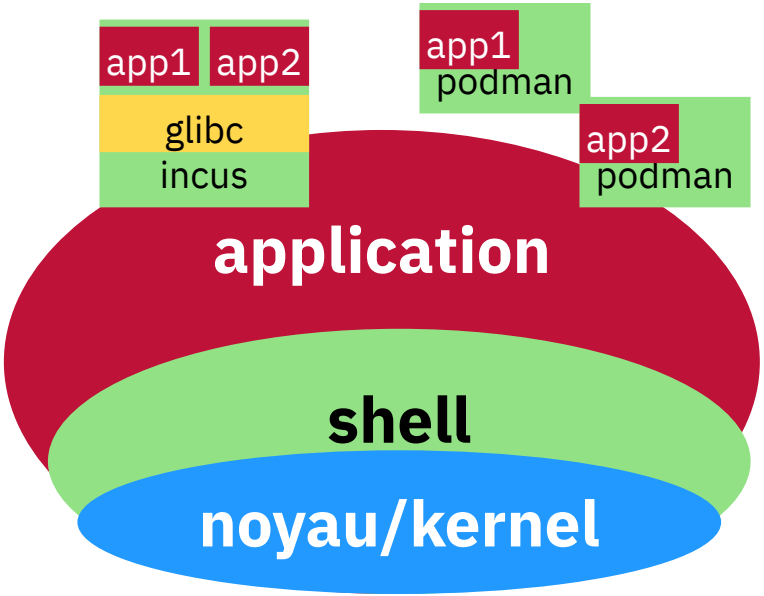
Distinguer conteneurs système et applicatifs

Conteneurs système → Incus

- Système de fichiers complet, sans noyau
- Données dans le conteneur ou montées
- Pile d'applications hébergées
- Administration système complète

Conteneurs applicatifs → Podman, Docker

- Une application par conteneur (OCI)
- Données stockées en dehors (volumes)
- Pile d'applications → pile de conteneurs



Incus : trois types d’instances

Un seul gestionnaire, une seule commande

- Conteneurs système (LXC) : une distribution complète
- Conteneurs applicatifs OCI, depuis Incus 6.3
- Machines virtuelles QEMU/KVM

Lancer une image OCI

- `incus remote add oci-docker https://docker.io --protocol=oci`
- `incus launch oci-docker:hello-world --ephemeral --console`

Un projet communautaire

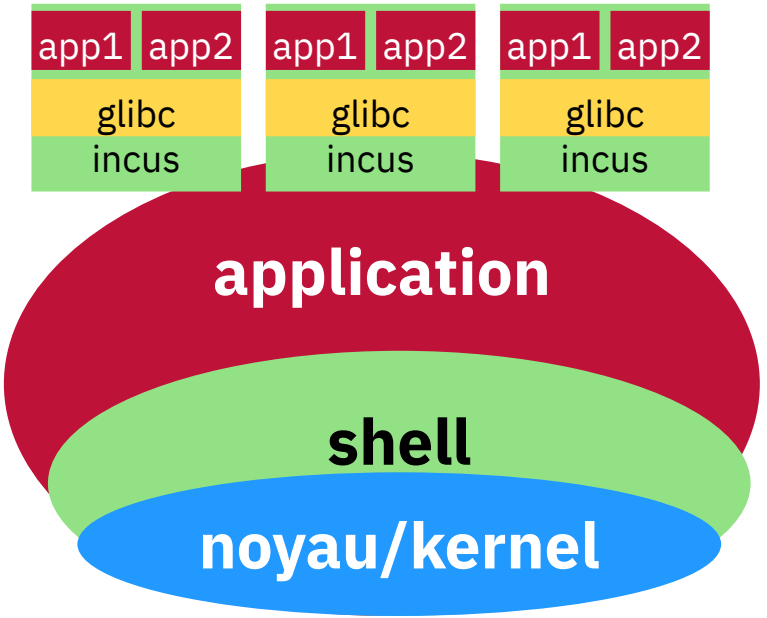
- Linux Containers, sous licence Apache 2.0

Sources : documentation Incus, linuxcontainers.org/incus/docs ; Stéphane Robert, « Running Docker (OCI) containers with Incus »

Introduire le scénario des manipulations

Opération sauver C-3PO

- Instancier des services dans les conteneurs système Incus
- Gérer les conteneurs dans une machine virtuelle
- Identifier les fonctions d'un profil
  - stockage et réseau
- Illustrer le fonctionnement des services « cloud »



Logiciel libre, licences et IA

« Je compare souvent l'open source à la science. La science a adopté cette notion de développer des idées ouvertement et d'améliorer les idées des autres. C'est ce qui a fait de la science ce qu'elle est aujourd'hui et a rendu possibles les incroyables avancées que nous avons connues. » — Linus Torvalds

Définir le logiciel libre et l’open source

Les quatre libertés (FSF)

- Exécuter le programme pour tout usage
- Étudier et modifier le code source
- Redistribuer des copies, originales ou modifiées

La définition de l’open source (OSI)

- Dix critères, dont la libre redistribution et la non-discrimination

Deux philosophies, des licences en grande partie communes

- Liberté des utilisateurs ou efficacité du développement

Différencier code source et code exécutable

Tout programme est écrit dans un langage

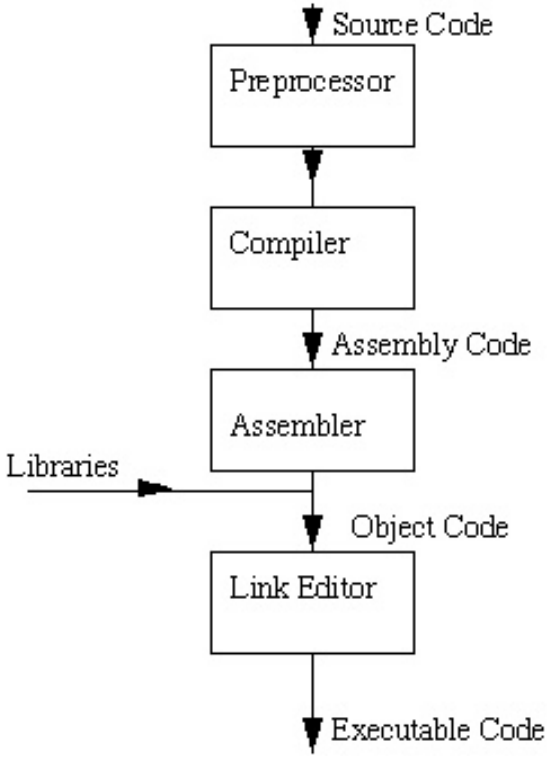
- Noyau Linux : langage C, et Rust
- Le code source n'est pas exécutable tel quel

Chaîne de développement

- Compilation : source → binaire exécutable
- Décompilation : noms et commentaires perdus

Propriétaire ou libre

- Propriétaire : droit d'utiliser un binaire
- Libre : accès au code source et droit de le modifier



Comparer licences permissives et copyleft

Licences permissives (BSD, MIT, Apache)

- Versions propriétaires dérivées autorisées
- Exemples : macOS, PlayStation (code FreeBSD)
- Apache 2.0 : licence explicite sur les brevets

Licence GNU GPL → copyleft

- Copyright ≠ copyleft
- Garantir les libertés des utilisateurs
- Dérivés redistribués sous la même licence

Classer les licences libres

Permissives	Copyleft faible	Copyleft fort
MIT BSD-3-Clause Apache-2.0	LGPL-2.1 MPL-2.0	GPL-2.0-only (noyau) GPL-3.0 AGPL-3.0 (usage réseau) EUPL-1.2 (Union européenne)
→ Intégration possible dans un logiciel propriétaire	→ Copyleft limité à la bibliothèque ou au fichier	→ Dérivés redistribués sous la même licence

Identifiant SPDX normalisé, en tête de chaque fichier source :

```
SPDX-License-Identifier: GPL-2.0-only
```

Décrire les principes du copyleft

- Appliquer un copyright sur le logiciel
- Fixer les conditions de distribution
  - *« Donner à tout utilisateur le droit d'utiliser, de modifier et de redistribuer le programme sans changer les conditions de distribution »*
- Code source et libertés associées non dissociables
  - https://www.gnu.org/licenses/copyleft.fr.html
- Le copyleft repose sur le droit d'auteur
  - Sans auteur, pas de licence à imposer

Licences à l’épreuve du cloud

Des licences « source disponible »

- MongoDB (SSPL, 2018), HashiCorp (BSL, 2023), Redis (SSPL, 2024)
- Objectif : empêcher la revente en service cloud

La réponse des communautés : les forks

- OpenTofu (Terraform), Valkey (Redis, Linux Foundation)

Des retours vers l’open source

- Elastic → AGPL-3.0 (2024), Redis 8 → AGPL-3.0 (2025)
- La licence est un outil de gouvernance : la confiance se perd vite

Source : InfoQ, « Redis Returns to Open Source under AGPL License », mai 2025

Poids ouverts ≠ open source

Un modèle d'IA = code + poids + données

- Open Source AI Definition 1.0 (OSI, 2024) : utiliser, étudier, modifier, partager
- Exige le code complet, les poids et une information suffisante sur les données

Trois cas à comparer

- **gpt-oss** (OpenAI) : poids sous Apache 2.0, données non publiées
- **Llama** (Meta) : licence communautaire, non open source selon l'OSI
- **OpenMDW-1.1** (Linux Foundation) : soumise à l'OSI en août 2026

Sources : opensource.org/ai ; openai.com ; Linux Foundation, mai 2026 ; OpenSourceForU, août 2026

Les communautés du libre face à l'IA

FSF

- Critères en préparation : le logiciel et les données brutes d'entraînement doivent être libres

Debian

- Résolution de 2025 sur les DFSG et les modèles d'IA, finalement retirée

Constat de septembre 2026

- Aucune des 16 grandes publications « ouvertes » recensées ne fournit sa recette de données

Sources : fsf.org, oct. 2024 ; debian.org/vote ; stateofopensource.ai, v1.1, sept. 2026

Droit d’auteur et code généré

Pas d’auteur humain, pas de protection

- U.S. Copyright Office (janvier 2025) : contribution humaine exigée
- Code produit par l’IA seule → aucun copyleft applicable

Jurisprudence récente

- Doe c. GitHub (9e circuit, 16 sept. 2026) : rejet des demandes DMCA contre Copilot
- GEMA c. OpenAI (Munich, 11 nov. 2025) : mémoriser une œuvre, c’est la reproduire

En pratique

- Le signataire reste responsable de la licence du code proposé

Sources : copyright.gov, janv. 2025 ; Stanford Copyright and Fair Use Center ; Observatoire européen de l’audiovisuel

Encadrer les contributions assistées par IA

Noyau Linux

- Signed-off-by réservé aux humains (DCO), code compatible GPL-2.0-only
- Oct. 2026 : Assisted-by remplacé par une mention dans la lettre de présentation (en cours)

Fedora (octobre 2025)

- Usage autorisé, responsabilité totale, déclaration des parts générées

Debian (vote d'août 2026)

- « Responsible Use of Generative AI » l'emporte, l'interdiction est rejetée

Sources : docs.kernel.org/process/coding-assistants.html ; heise, nov. 2026 ; docs.fedoraproject.org ; debian.org/vote/2026/vote_002

Mesurer les effets de l’IA sur les projets

Saturation

- curl arrête son bug bounty le 31 janvier 2026 (rapports « slop »)
- Robots d’indexation sur les forges → défis Anubis (GNOME…)

Nouvelles attaques

- Slopsquatting : 5,2 % à 21,7 % de paquets suggérés qui n’existent pas

Nouveaux outils

- Linux 7.3 : l’outil de revue Sashiko signe 19,4 % des Reported-by

Sources : daniel.haxx.se, janv. 2026 ; Semiautonomous Systems ; OX Security (USENIX Security 2025) ; LWN.net, oct. 2026

Situer le cadre européen

Cyber Resilience Act

- Fabricants : signalement des vulnérabilités exploitées depuis le 11 sept. 2026
- Intendants open source : à partir du 11 déc. 2027, via la plateforme de l'ENISA

AI Act

- Règles des modèles à usage général applicables depuis le 2 août 2026
- Modèles sous licence libre : exemption partielle de documentation
- Systèmes à haut risque repoussés au 2 déc. 2027 et au 2 août 2028

Stratégie open source de l'UE (3 juin 2026)

- L'open source comme levier de souveraineté technologique

Sources : Commission européenne ; ENISA ; AI Act Service Desk ; Parlement européen (omnibus numérique, JO du 24 juillet 2026)

Observer les licences sur un système Debian

Fichier copyright de chaque paquet (format DEP-5)

- `cat /usr/share/doc/bash/copyright`

Analyser un arbre de sources

- `licensecheck -r .`
- `reuse lint`

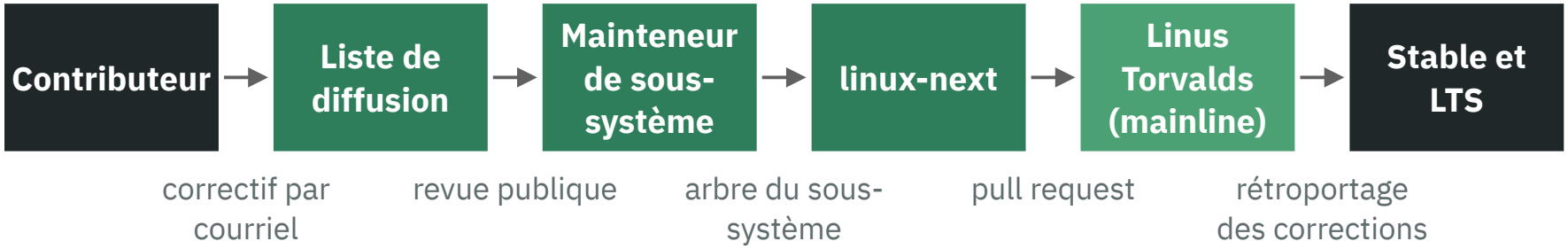
Modèles d’IA

- Lire le champ license de la fiche du modèle sur Hugging Face
- Comparer avec la définition de l’OSI

Développement open source

« Avec l’open source, vous avez le droit de contrôler votre propre destin » — Linus Torvalds

Suivre le parcours d’un correctif



Un cycle de 9 à 10 semaines

Fenêtre de fusion 2 semaines	**rc1 → rc7 ou rc8** une version candidate par semaine, corrections uniquement	**Version 7.x**

Les correctifs circulent par courriel (git send-email, b4, archives lore.kernel.org).

Seuls les mainteneurs envoient des pull requests à Linus, pendant la fenêtre de fusion.

Une version n’est jamais retardée pour une fonctionnalité : elle attend le cycle suivant.

Décrire le développement du noyau Linux

Collaboratif et ouvert

- L’un des plus grands projets de logiciel libre
- Linux 7.3 : 3 021 développeurs, dont 807 nouveaux
- 232 employeurs identifiés, 18 765 commits en 9 semaines

Cycle de publication calendaire

- Nouvelle version toutes les 9 à 10 semaines
- Les versions ne sont pas retardées pour une fonctionnalité
- Développement et intégration en parallèle (linux-next)

Source : Jonathan Corbet, « Development statistics for the 7.3 kernel », LWN.net, 19 octobre 2026

Automatiser la revue et les tests

Linux 7.3 en chiffres

- Reviewed-by sur 50,9 % des commits, en hausse
- Près de la moitié des rapports de bogues viennent d’outils
- Sashiko : 19,4 % des Reported-by, syzbot : 15,3 %

Points de vigilance

- Des contributeurs modifient du code qu’ils ne comprennent pas
- L’effet sur la qualité se mesurera sur plusieurs années

Des outils assumés

- Linus Torvalds publie du code développé avec Claude Code

Sources : LWN.net, 19 octobre 2026 ; heise, novembre 2026

Décrire les fondations open source

Définir et défendre

- **OSI** : définitions des licences et de l'IA open source
- **FSF** : licences GNU et philosophie du logiciel libre

Héberger et financer les projets

- **Linux Foundation** : noyau, CNCF, OpenSSF
- **Agentic AI Foundation** (2025) : MCP, AGENTS.md, A2A
- Apache, Eclipse (Europe), Mozilla, The Document Foundation

Sources : Linux Foundation, 9 décembre 2025 ; aaif.io

DevOps et CI/CD

Automatiser, intégrer, livrer : la culture système au service du flux continu.

Décrire les évolutions des métiers

Le credo DevOps

- Toujours plus d’automatisation
- Toujours plus d’applications intégrées ou dépendantes
- Tout est programmable : l’infrastructure se décrit en code

Fusion du développement et de l’administration système

- Vidéo → DevOps vs SRE
- Assistants IA : produire vite, relire et tester toujours

Définir les concepts CI/CD

Intégration continue – *continuous integration*

- Déposer souvent le code dans Git ; chaque modification déclenche une construction et des tests

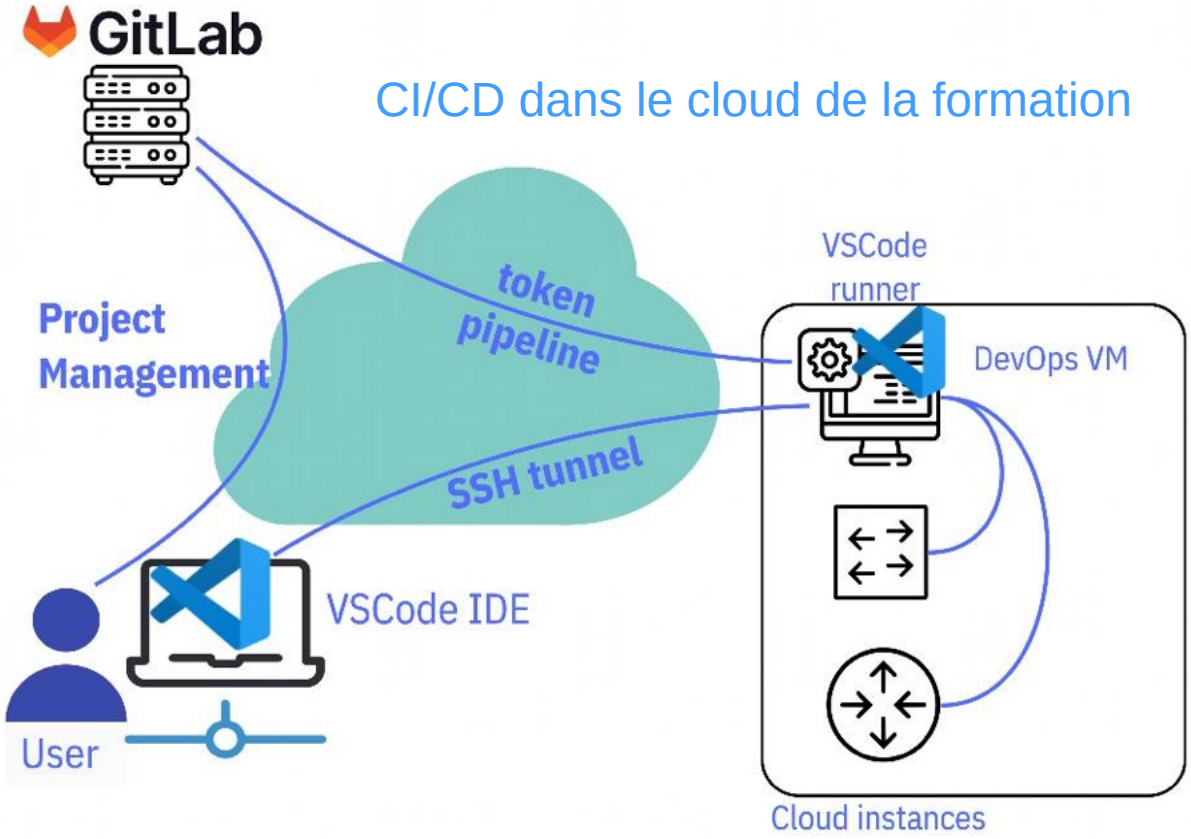
Livraison continue – *continuous delivery*

- Produire un artefact toujours déployable ; mise en production sur décision

Déploiement continu – *continuous deployment*

- Chaque modification validée part en production automatiquement

Utiliser la CI/CD



Sécuriser la chaîne d’approvisionnement

Savoir ce que l’on déploie

- SBOM : inventaire des composants (formats SPDX, CycloneDX)

Prouver d’où vient un binaire

- Signature des artefacts et des dépôts APT (OpenPGP)
- Constructions reproductibles : même source → même binaire

Vérifier ce que l’IA propose

- Contrôler l’existence et l’origine des dépendances suggérées
- CRA : la gestion des vulnérabilités devient une obligation légale

Distributions : le cas Debian

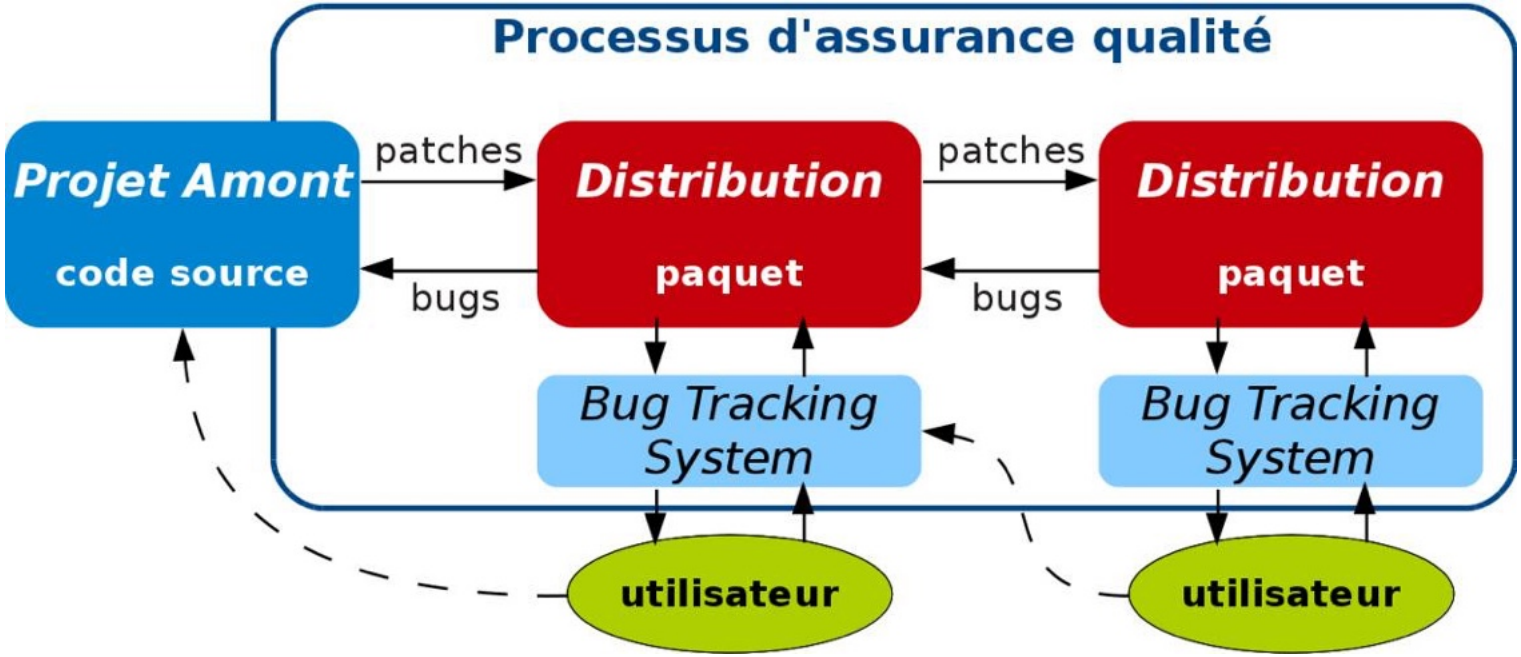
Une distribution réunit un noyau, des paquets et un processus qualité.

Définir les fonctions d’une distribution

- Distribution → canal de diffusion du logiciel libre
- Association de 3 éléments
  - Noyau
  - Un ou plusieurs shells
  - Un ensemble d’applications
- Applications distribuées sous forme de paquets
  - Code binaire exécutable
  - Configuration type
  - Métadonnées : dépendances, scripts de maintenance
- Gestionnaire de paquets
  - Pierre angulaire de la vie d’une distribution

Décrire filiations et processus qualité

- Distribution → filiations et processus qualité
- Pipeline : amont → distributions → utilisateurs



Choisir une distribution

- Facilité d’adaptation
  - Obtenir une configuration type par contexte
  - Bénéficier de l’assurance qualité
- Continuité lors des mises à jour
  - Cohérence des évolutions et des corrections
  - Mise à niveau d’une version stable à la suivante
- Cycle de vie connu à l’avance
  - Durée du support de sécurité, LTS
  - Continuité de service, haute disponibilité

Pourquoi Debian GNU/Linux ?

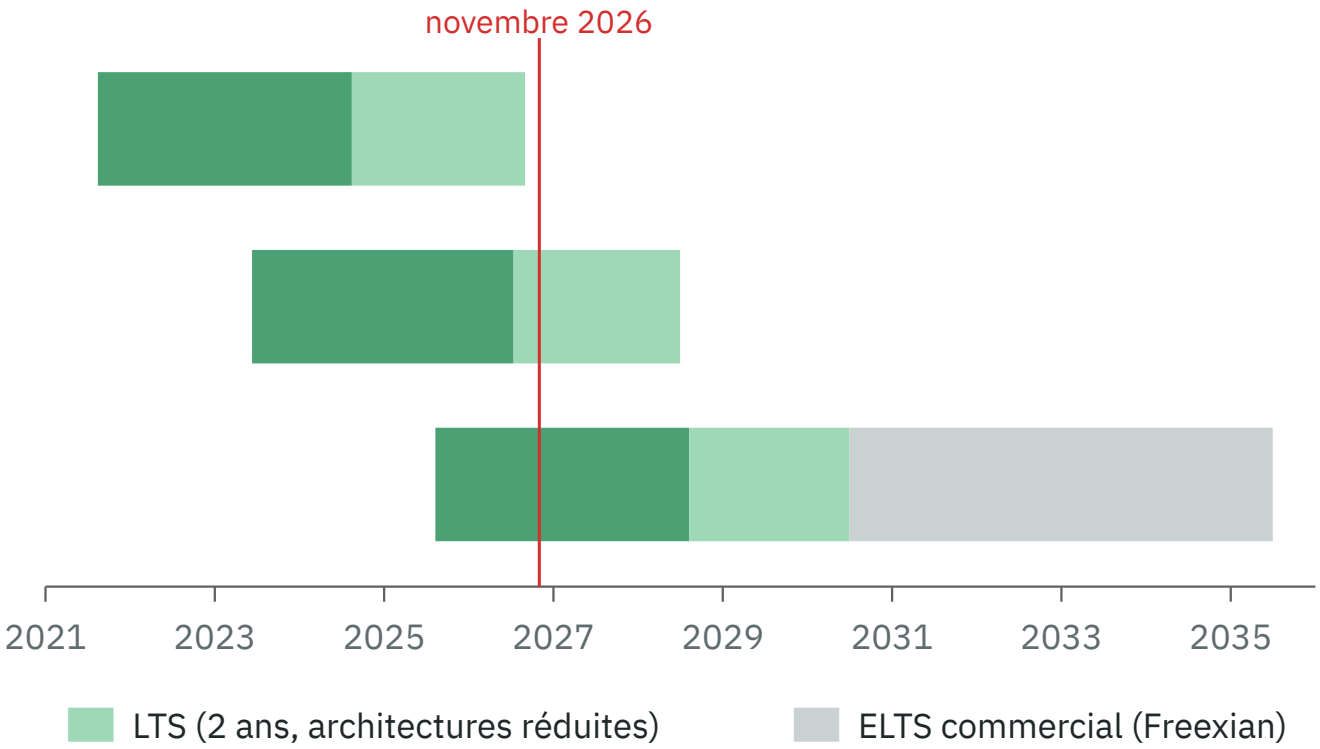
- Principes du logiciel libre selon Debian (DFSG)
  - Règles qui garantissent qu'un logiciel est libre
  - https://www.debian.org/social_contract.fr.html
- Charte Debian (Debian Policy Manual)
  - Procédure qualité du projet
  - https://www.debian.org/doc/debian-policy/
- Gestionnaire de paquets APT
  - Synthèse des qualités de la distribution
  - Mise à niveau entre versions stables sans réinstallation
  - Configuration séparée de l'application
  - Publication automatisée des mises à jour

Suivre le cycle de vie de Debian

Debian 11 bullseye
fin de vie le 31/08/2026

Debian 12 bookworm
en LTS jusqu'au 30/06/2028

Debian 13 trixie
stable, 13.7 le 12/09/2026



Sources : debian.org/releases ; debian.org/News 2026 ; freexian.com. Debian 14 forky est en testing.

Utiliser Debian 13 « trixie »

APT 3.0

- Nouveau solveur de dépendances, affichage en colonnes
- Sources au format deb822 : /etc/apt/sources.list.d/debian.sources
- `apt modernize-sources`

Stabilité ou nouveauté

- trixie : noyau 6.12 LTS
- testing (forky) : suit les versions 7.x du noyau

Décrire les points forts de Debian dans le cloud

- Stabilité et fiabilité reconnues
- Large base de paquets logiciels
- Développement communautaire transparent
- Support à long terme des versions stables
- Images cloud officielles (cloud.debian.org)

Exemples d’hébergeurs : IONOS, Hetzner, OVHcloud, Scaleway

Décrire les atouts de Debian en sécurité

- Rejeter la « sécurité par l'obscurité »
- Publier rapidement les correctifs (annonces DSA et DLA)
- Support : 3 ans complets + 2 ans de LTS
  - Architectures réduites en LTS, ELTS commercial au-delà
- Mises à jour automatiques : unattended-upgrades
  - Paquet à installer et à configurer
- Politique de sécurité stricte
  - Installation minimale par défaut
  - Séparation des privilèges, cloisonnement des processus
  - Suivi public des vulnérabilités (security-tracker)

Bilan du cours 2

Retenir les points clés des cinq parties.

Retenir : virtualisation et conteneurs

- Émulation, virtualisation assistée, paravirtualisation (virtio)
- Hyperviseurs de type 1 et 2 : KVM est de type 1
- Conteneurs système (Incus) et applicatifs (Podman, Docker)
- Incus gère conteneurs système, images OCI et machines virtuelles

Retenir : logiciel libre, licences et IA

- Libre = accès au code source et quatre libertés
- Permissive ou copyleft : choisir ce que l’on veut garantir
- « Poids ouverts » ne veut pas dire « open source »
- Le contributeur humain reste responsable, quel que soit l’outil
- CRA et AI Act : la conformité entre dans le métier

Retenir : développement, DevOps et Debian

- Noyau : cycle calendaire, mainteneurs, revue de plus en plus automatisée
- Fondations : gouvernance neutre des projets, IA comprise
- CI/CD : intégrer, livrer, déployer en continu
- Chaîne d’approvisionnement : SBOM, signatures, constructions reproductibles
- Debian 13 : stabilité, cycle de vie connu, sécurité