Traitement des incidents de sécurité dans un projet libre

Matthieu Herrb

Campus du Libre, 6 novembre 2021

https://homepages.laas.fr/matthieu/talks/campus-du-libre-21.pdf

Licence

Agenda

Agenda

À propos de l'auteur

- Ingénieur de Recherche au LAAS du CNRS
- Contributeur de X.Org et d'OpenBSD depuis plus de 25 ans
- Maintaineur de X.Org sur OpenBSD (project Xenocara)
- Membre de l'équipe qui gère les incidents de sécurité de X.Org depuis 2006

Publication des correctifs de sécurité pour un Logiciel Libre

La transparence du logiciel libre peut poser des problèmes
en cas de publication trop rapide d'un correctif pour un bug de sécurité :

- fournit suffisament d'infos pour les pirates pour exploiter la vulnérabilité...
- ...sans laisser le temps aux utilisateurs honnêtes de corriger pour se protéger !

Responsible Vulnerability disclosure

Vulnerabilities in widely used software can have a huge bad impact on existing installations.

General process, globally agreed on by many Free and Open Source projects.

- "White hats" security researchers report the vulnerabilities they find privately, work with the maintainers on a fix and decide on a disclosure date.
- Gives time for binary package maintainers to prepare updates, ready to be installed.
- So this is a limitation to full transparency, to mitigate the impact (avoid "zero days").
- But we should keep the embargo on vulnerabilities as short as possible.
- Some organizations (not X.Org !) have bug bounties programs to encourage this process.

Full disclosure

Vulnerabilities are disclosed as soon as they are found by researchers

Patches are not yet available

Users get exposed to the vulnerability.

Note that this can happen, even with a responsible disclosure policy in place :

- because of “black hats” who don’t follow the policy
- because of mistakes from developers who don’t see the security implications of their bug fixes (or don’t know about the policy)

Agenda

Acteurs et vocabulaire

- MITRE et CVE-IDs
- CVE Numbering authorities (CNA)
- Scores de vulnérabilités CVSS pour évaluer « objectivement » l'impact d'une faille.
- La liste de diffusion OSS-security et le wiki associé.
- La liste de diffusion distros
- Le CERT US et le CERT FR.
- L'ANSSI
- ...

Le marché des vulnérabilités

Quelques sociétés spécialisées en sécurité vendent à leurs clients des informations “en avance” sur les vulnérabilités et les correctifs.

- achètent les bug reports auprès des chercheurs,
- préparent un rapport (éventuellement un correctif),
- le diffusent à leurs clients,
- après quelques semaines :
  - contactent l’auteur initial,
  - acceptent la publication.

Constat...

Souvent le nom, le logo et le site web de la vulnérabilité ont mobilisé plus d’énergie que le correctif et sa diffusion

En pratique

- Créer un point de contact facilement identifiable pour signaler les pbs de sécurité de manière confidentielle.
  (par ex. : `security@project.org`)
- Se faire connaitre auprès de la communauté (wiki oss-sec,...) pour être tenu au courant des failles qui peuvent affecter son code.
- Préparer une procédure de gestion des incidents de sécurité (signalement ou découverte de faille)
- Suivre la procédure.

Procédure standard

1. Développer le correctif et informer la personne qui a signalé la faille.
2. Obtenir un numéro de CVE (Common Vulnerabilities and Exposures).
3. Si le projet est packagé (par des distributions Linux par ex), prévenir leurs contacts de sécurité de la faille, du correctif et décider ensemble d'une date de publication.
4. Tester et valider le correctif en collaboration avec les packageurs !
5. Le jour donné, publication du correctif et des paquets mis à jour et diffusion de l'information (listes de diffusion, réseau sociaux,...)

Choix de la date de publication

- pas une veille de WE ou (pire) de pont
- pas le lundi matin (souvent occupé par d'autres choses)
- le mardi ou le jeudi sont des bons jours.
- tenir compte des fuseaux horaires. 14 :00 GMT est généralement considéré comme le moins pire des compromis.
- éviter le 3e mardi du mois (patch tuesday de Microsoft) déja assez occupé en général.

Ce qui est important

- ne pas se précipiter pour publier un bug-report ou corriger un bug si il peut avoir un impact de sécurité.
- discuter du pb avec des personnes de confiance uniquement.
- ne pas ignorer ou laisser traîner inutilement un problème.
- ne pas minimiser les conséquences possibles d'un bug
- reconnaître ses erreurs et en informer les autres rapidement.
- ne pas culpabiliser inutilement. "Shit happens".

Sécurité des structures

Assurer la confidentialité, l'authenticité et l'intégrité des échanges.
Repose en général sur PGP.

- diffuser sa clé publique :
  - pour permettre l'envoi de messages chiffrés au point de contact.
  - pour signer les avis et les correctifs
- Avoir sous la main les clés publiques des partenaires
- Ne jamais faire suivre en clair un message reçu chiffré (sauf autorisation expresse)
- considérer toutes les informations comme sensibles et les protéger comme telles (chiffrement sur disque, pas sur un poste en libre-service ou une clé usb qui circule, ...

Limites

- Difficile de garder la confidentialité - d'identifier les sources de fuites
- Grand nombre d'acteurs
- Retarder trop la publication d'un correctif officiel laisse les utilisateurs vulnérables en cas de re-découverte ou de fuite.
- Les bugs découverts via un exploit en circulation (Zero-day) doivent être corrigés au plus vite.

CERTs traditionnels

Relation difficiles....

- grosses machines administratives manquant de réactivité
- manque de confiance mutuelle
- imposent souvent des contraintes de confidentialités déraisonnables.
- peuvent néanmoins être utiles pour aider à coordonner la gestion de failles importantes.

Agenda

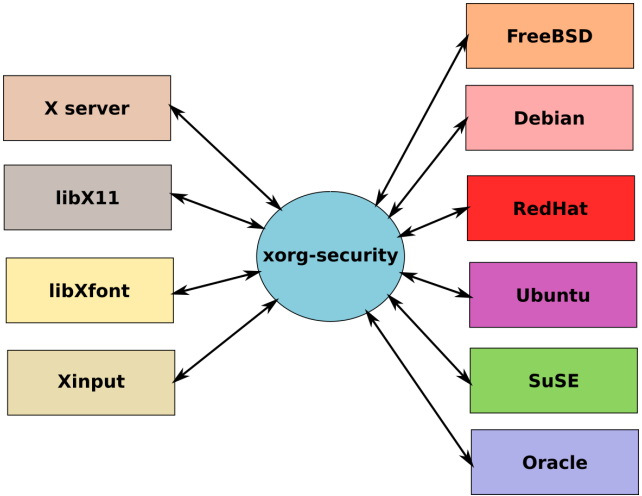
Exemple : X.Org

- Projet qui développe le système de multi-fenêtrage X (The X Window System) et Wayland.
- Animé par la fondation X.Org
- Membres de la fondation : toute personne impliquée dans le projet (développeur, testeur, soutien,...) gratuit. Critères définis par les status.
- Sponsors de la fondation : entreprises qui donnent de l'argent.
- Budget : environ 40k$ annuels : 2 conférences, achat de matériel, *vacation of code*.
- Infrastructures hébergées à l'Université de Portland.

xorg-security

- adresse de contact `xorg-security@lists.x.org`
- regroupe des développeurs experts en sécurité de X et les distributeurs
- liens avec les listes `distros` et `linux-distros`
- bugs privés dans gitlab : visibles des seuls utilisateurs autorisés

Interactions



An example of the actual process

2019-03-06 jolibug@gmail.com sends an email to `xorg-security@lists.x.org` talking about a new vulnerability discovered in libXfoo version 3.33.

2019-03-07 some-dev@example.com, member of the list, confirms the existence of the problem and analyzes that it was introduced by a change in version 2.45. They answer to jolibug to acknowledge their email and asks them if a CVE-Id has already been allocated or a disclosure date has been decided. They also propose a patch.

2019-03-08 jolibug answers that no date
has been set up and no id was allocated.
they're willing to postpone the disclosure
of the bug up to a reasonable delay.

2019-03-08 some-dev asks Redhat for a CVE-Id

2019-03-09 someone from Red-Hat allocates CVE-2019-666 for this.

2019-03-09 knowing that libXfoo is used by a number of Linux and BSD distros, some-dev writes to the distros mailing list, and proposing 2019-03-22 (14 days later) as release date.

2019-03-09 another-dev@BlueCap.com answers to the distros mail signaling the proposed date is a friday, which is a bad day for releasing security advisory and suggests to move it forward to 2019-03-19.

2019-03-09 some-dev agrees and informs jolibug and xorg-security.

2019-03-11 eric@nobugs.com says in an email to distros and xorg-security that some-dev's patch isn't fully solving the issue : a similar issue exists in another function in libXfoo.

2019-03-12 new version of the patch sent to xorg-security, jolibug and distros.

2019-03-14 some-dev sends xorg-security and jolibug an draft of the security advisory, what should be the final patch.

2019-03-15 some-dev prepares a new release of libXfoo including the patch and checks that it's good for release.

2019-03-19 14 :00 GMT

- libXfoo 3.34 is released
- the security advisory is sent to `xorg-announce@lists.x.org` and `oss-security@lists.openwall.com`
- the X.Org wiki security page is updated
- jolibug send out their advisory to various security mailing lists and social networks.

shortly after : The bug is now public !:

- Various distros distribute updated binary packages.
- MITRE makes the CVE-ID description public,
- Various CERTS, as well as LWN and Phoronix publish their own news on the bug,

A Quick Taxonomy of the reported vulnerabilities

In the early days trivial buffer overflows, argument sanitizing bugs,..
almost always direct local root access.

nowadays Mostly protocol handling bugs

- both client side and server side
- insufficient or incorrect validation of the (complex) protocol messages $\rightarrow$ unauthorized memory accesses
- some file format decoding bugs (mostly fonts)

Mitigations :

- less privileges for the X server
- less privileged X clients
- XCB automates protocol encoding/decoding, but not done server side yet.

Agenda

Conclusion

- Un traitement responsable des bugs de sécurité est indispensable.
- Tant pis pour la “full disclosure”.
- Ne pas rester isolé
- Éviter de publier du code passoire. Se faire aider avant la publication.
- Prévoir l’infrastructure pour le traitements des incidents dès la mise à disposition initiale du projet

Questions ?

Resources

- the Security Checklist on the X.Org wiki (to be revised)
- Security process for Open Source Projects, Alex Gaynor, 2013
- The CERT Guide to Coordinated Vulnerability Disclosure
- The OSS-security mailing list and wiki
- The distros mailing-list
- The CVE request form at MITRE
- Google Project Zero
- Zero Day Initiative
- Pour signaler une faille de sécurité en France
- Le site de l'Observatoire de la Sécurité des Systèmes d'Information et des Réseaux