Inside your Samba Security Release

Andrew Bartlett

Catalyst / Samba Team

SAMBA TEAM

catalyst

expert open source solutions

What is a Samba security release?

Where do security reports come from?

Samba Team

Thinking hard or
via direct customers

Public mailing lists (ouch)

Often folks report a crash
and we realise it is a DoS

security@samba.org

Primary contact address
for security issues from
the general public

Fuzzing

Increasingly we find issues
by building a fuzzer

What security reports warrant a Samba security issue?

- Not entirely obvious!
- Privilege escalation
  - including share escape
- Not every crash
  - smbd self-DoS excluded
  - but local crash of winbindd included
- Trust the AD DC (mostly)
  - But don’t trust the server in general
- CVSS 4.5 minimum in general

Fundamental steps in a security release

Discovery

Issue is discovered, and reported to us.

Response

Issue subjected to triage.

Is this important enough?

CVSS Scoring done

Develop a fix

A patch is developed

...continued steps to make a security release

Backport to supported versions

From master to (currently)

4.14, 4.13 and 4.12

Run CI

Both on Gitlab CI **and** on sn-devel

Release!

Tarballs, announcements etc

What really makes a Samba security release?

Hard, hidden work...

And a very specific process:

https://wiki.samba.org/index.php/Samba_Security_Process

“Someone should feel responsible…”

- The Samba Security Process opens so hopefully!
- No overall Samba management so left to developers and their employers
  - Catalyst staff are ‘on the clock’ for all Samba development, so essentially it is up to me
- Some basic patterns:
  - You break it you fix it (regressions should be fixed by those involved)
  - Last one to touch the code owns all the bugs
  - ZeroLogon: All hands to the pump!

We use bugzilla

- The Samba Bugzilla is our store of private security details
- Group based access control
  - Samba Team
  - Vendors
- Ad-hoc additional users via CC
- Ability to redact comments
  - So able to make the record public later
- Suits the security process pretty well actually!

Creating the bug

- Mark the new bug private (under Advanced Fields)
  - Select “Samba Core developers” restriction
- Title it [EMBARGOED][SECURITY]
  - Avoids confusion and makes mail clear
- Fill in as much detail as you have from the reporter

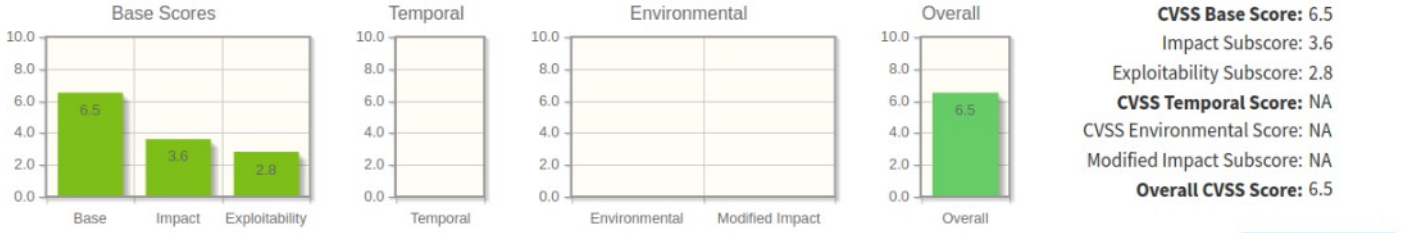
Write the advisory

- Much is not yet known about the bug
  - Like versions with the fix
- Write as much as possible anyway
  - Helps guide research into impact and history
- Confirm the reporter can be named
  - Privacy matters
  - Companies have policies and preferred titles!

Do a CVSS Calculation

Common Vulnerability Scoring System Calculator

This page shows the components of the CVSS score for example and allows you to refine the CVSS base score. Please read the CVSS standards guide to fully understand how to score CVSS vulnerabilities and to interpret CVSS scores. The scores are computed in sequence such that the Base Score is used to calculate the Temporal Score and the Temporal Score is used to calculate the Environmental Score.



CVSS Base Score: 6.5
Impact Subscore: 3.6
Exploitability Subscore: 2.8
CVSS Temporal Score: NA
CVSS Environmental Score: NA
Modified Impact Subscore: NA
Overall CVSS Score: 6.5

Show Equations

CVSS v3.1 Vector
AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*
Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*
Low (AC:L) | High (AC:H)

Privileges Required (PR)*
None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*
None (UI:N) | Required (UI:R)

Scope (S)*
Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*
None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*
None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*
None (A:N) | Low (A:L) | High (A:H)

https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

Get a CVE Number

- Red Hat's security response team is normally pretty fast
  - We don't give them details, just a bug link they can't read
  - Turn around is normally 24 hours
- Put the number on the bug
  - Bug title
  - Bug alias
    - Changes the bug into a CVE- number elsewhere in bugzilla

Write patches (one small part of the process)

Write patches for master

Remember to include tests, not just an “exploit script”

Backport to supported versions

To release candidates, current, maintenance and security-only

CI, much CI

We have a private GitLab instance

Every patch, for every version must pass CI

Karolin will not run your CI for you!

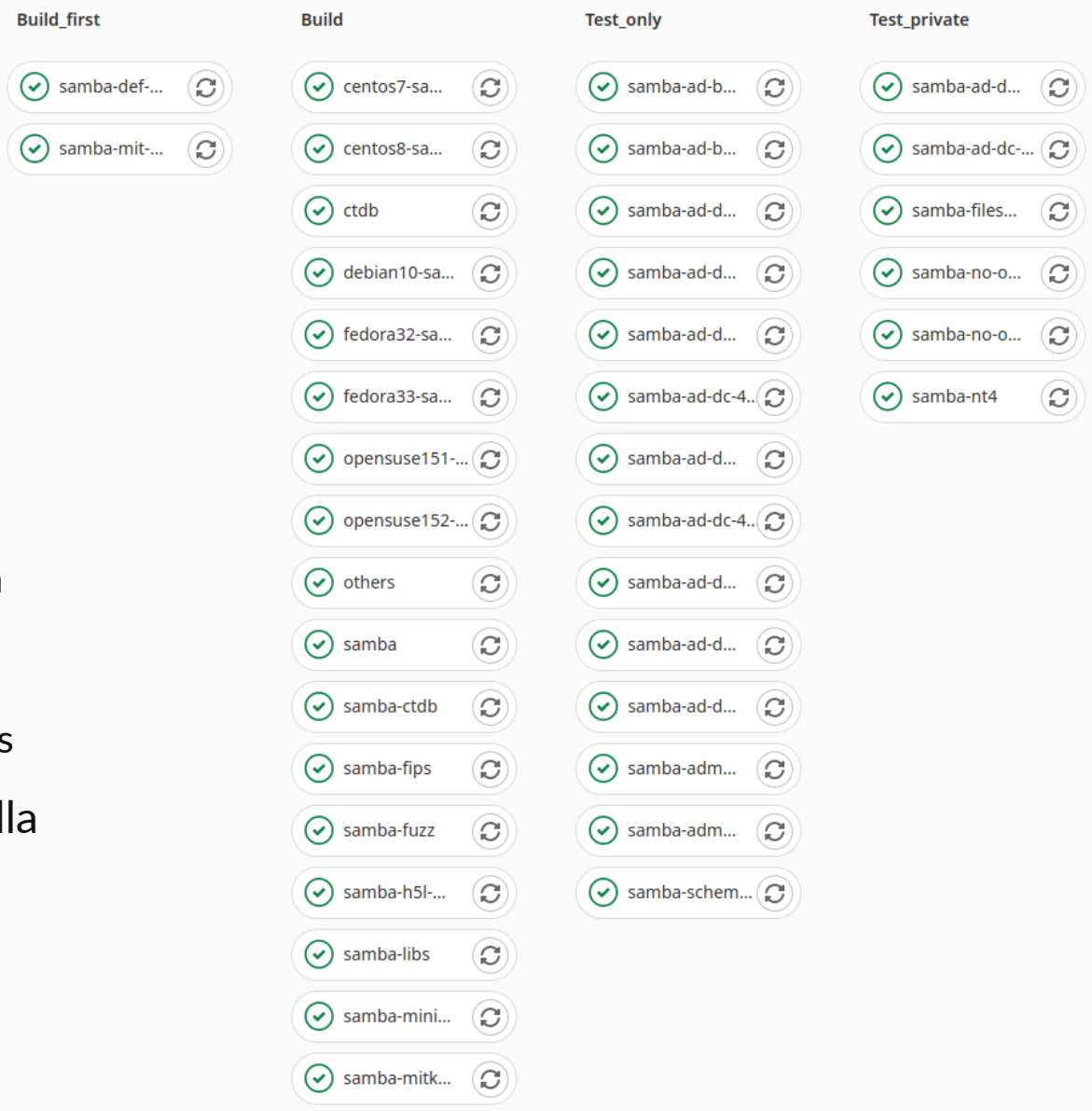
We can't use public GitLab for CI

Not even a private repo on gitlab.com

We have access to another GitLab

Attached to the Samba Team's runners

Tag ci-passed on the patch in Bugzilla



Then more CI

- The “person feeling responsible” runs CI on each individual patch
  - Just like the pre-commit CI on any other merge request
- The Release Manager runs CI using autobuild
  - on sn-devel on the whole release (to ensure no conflicts)
- So Karolin runs the CI for you as well
  - (but this better not fail)

Notifications

Samba Team

Coordinate a release date with Karolin (release manager)

Public mailing lists

7 Days before the release (bare details only)

Advisory

Finish the advisory with final versions and confirmed details

Vendors

10 Days before the release via Bugzilla

Release time!

Not actually a relaxing day however!

Time To Party: License: CC0 Public Domain Linnaea Mallette via https://www.publicdomainpictures.net/en/view-image.php?image=277619

What does a release involve?

- A number of new Samba versions
  - Patched Stable Samba versions: Never including bug fixes or new features
- Upload tarball to https://download.samba.org/pub/samba/
- Push patches to stable branches (bypassing autobuild)
- Publish Announcments
  - Pre-announcments (one per “drop”)
  - Advisory (per issue)
  - WHATSNEW updates
  - Announcement e-mails to mailing lists
- Website and wiki updates

How much work is this anyway?

“Someone should feel responsible...”

...comes with a cost

How much time is involved?

CVE-2021-20251 (undisclosed):

Suspended at 285 hours

CVE-2020-27840 Unauthenticated remote heap corruption via bad DNs

52 hours

CVE-2021-20277 out of bounds read in ldb_handler_fold

32 hours

Release Management (Karolin / SerNet)

2 – 8 hours per release

ZeroLogon: Zero Notice!

0

CVE-2020-1472

Not notified to Samba in advance

SerNet / Red Hat

Worked tag-team follow-the-sun

catalyst

Catalyst Upstream contribution

> 100 hours

Release Management (Karolin / SerNet)

No notice is no fun!

Two years of Security updates

CVE	Vendor	Type of flaw	CVE	Vendor	Type of flaw
CVE-2018-16860	Red Hat	Historical	CVE-2020-10700	Catalyst	Regression
CVE-2019-10197	Google / SerNet	Historical	CVE-2020-10704	Catalyst	Historical
CVE-2019-10218	Google	Historical	CVE-2020-10730	Catalyst	Regression
CVE-2019-12435	Catalyst	Historical	CVE-2020-10745	Catalyst	Historical
CVE-2019-12436	Catalyst	Regression	CVE-2020-10760	Catalyst	Regression
CVE-2019-14833	Catalyst / SerNet	Regression	CVE-2020-14303	Catalyst	Historical
CVE-2019-14847	Catalyst	Historical	CVE-2020-14318	Google	Historical
CVE-2019-14861	Catalyst	Historical	CVE-2020-14323	SerNet	Historical
CVE-2019-14870	Red Hat	Historical	CVE-2020-14383	Catalyst (from external patch)	Historical
CVE-2019-14902	Catalyst	Historical	CVE-2020-1472	SerNet / Catalyst / Google / Red Hat	Protocol
CVE-2019-14907	Catalyst	Historical	CVE-2020-27840	Catalyst	Historical
CVE-2019-19344	Catalyst	Regression	CVE-2021-20254	SerNet	Historical
			CVE-2021-20277	Catalyst	Historical

Overall scale of effort: Last two years

19 non-regression CVEs

(this is at a very expansive definition of “regression”)

50 hours per CVE easily

950 hours

a full time week per month, all year

Catalyst “Regressions”	6	
Catalyst “Historical / Protocol”	12	63.16%
SerNet	5	26.32%
Google	4	21.05%
Red Hat	3	15.79%

Collaborative efforts recorded under all companies
Based on credits in the security advisory

Workload is increasing:

Catalyst recorded **100+ hours per month** over the past 8 months

Includes fuzzing research and unreleased issues

The heavy weight of historical bugs

- Who should fix really old code from pre-history?
  - (or at least before Catalyst started doing Samba in 2013)
- Currently roughly
  - AD DC:
    - Catalyst
  - smbd etc:
    - SerNet
    - Google (Jeremy)
    - Red Hat

Admirable, but not sustainable!

Why is this so hard?

What are the impacts?

Peer review: Do we issue too many CVEs?

Debian issued a **no-dsa** for may of our CVEs:

Opting not to ship a stable update

17 of 25 issues (those in **bold)**

Including ZeroLogon

So perhaps there are other factors at play

Still deeply depressing to put effort into a CVE and not have it shipped to users with urgency

CVE-2018-16860	CVE-2020-10700
CVE-2019-10197	**CVE-2020-10704**
CVE-2019-10218	**CVE-2020-10730**
CVE-2019-12435	**CVE-2020-10745**
CVE-2019-12436	**CVE-2020-10760**
CVE-2019-14833	**CVE-2020-14303**
CVE-2019-14847	**CVE-2020-14318**
CVE-2019-14861	**CVE-2020-14323**
CVE-2019-14870	**CVE-2020-14383**
CVE-2019-14902	**CVE-2020-1472**
CVE-2019-14907	**CVE-2020-27840**
CVE-2019-19344	**CVE-2021-20254**
	CVE-2021-20277

Denial of Service issues in AD DC design

- Many single-process or pre-forked tasks
  - Pre-forked tasks chosen over fork()ing due to DoS and performance issues!
- Instant 6.5 CVSS if you can crash one on demand!
  - Because of the (by design) restart-backoff
  - We stand by the design, but hate paying the CVEs

DNS is a weak point

- The DCE/RPC DNS Management “dnsserver” in Windows is locked to Administrators only
  - Samba relies on the LDAP ACLs instead
  - Samba has issued number of security releases due to bugs in this code
    - (They would not be worth a security release if only privileged users cout exploit them)
- The LDB partitions for DNS can be written as a normal user
  - Means our DNS record parsing code is also an attack surface
  - I’m keen to diverge from MS behaviour here

Unaddressed issues

- A number of lower-priority issues are embargoed without a fix
  - Like the 280 hour CVE-2021-20251 I mentioned earlier
- Marking an issue as [SECURITY] prevents a partial fix from being developed
  - Because to fix anything requires fixing everything
  - Works against Samba's pattern of incremental development
  - Users can't know about the issue to implement their own workaround

Embargo can stop an issue being fixed at all

- Embargo prevents discussion with potential funders!
- Very hard to sell “support Samba security” as it is
  - Much harder when we **look** like we are addressing everything just fine

We need to make some changes

Reducing the feature set wont help (quickly)

A quite long-term play

- Takes two years to stop having to fix the bugs

Only ever deprecated one feature mid-release

- MIT Kerberos KDC

We should still try to actively reduce our feature set

Stop shipping a distinct LDB

- No longer any good reason why LDB should be a separate tarball
  - And therefore a separate release etc
- Need for LDB release makes more complex both:
  - Samba security releases
  - Distribution security releases (as the versions must be aligned)
- Should be installed just like libsmbclient, as a Samba public library

Raise the bar for an embargo

- Denial of Service issues should only under embargo if:
  - Being actively worked on or
  - within 90 Days of the report
- Password policy weakness should not qualify
  - Issues that Samba can be made to to allow a poor password or
  - Issues where an “OK” password would mitigate the issue

Fund Fuzzing

- Users of Samba who value security should
  - Fund a fuzzing campaign against Samba
  - Include in that funding enough for the finder to fix the issues raised
- Compared with just funding “security work” fuzzing
  - Always has a work product (an outcome)
  - Has a good chance of finding at least a minor issue
  - Google oss-fuzz will keep it running after the end of the project

Fund Hardening

- Users of Samba who value security should:
  - Fund changes like locking down our DNS partition
  - Assist with the upgrade to a modern Heimdal
  - Help us get rid of questionable cryptography like LM and LMv2
  - Fund a key roll-over scheme for the AD DC and krbtgt accounts
- Or for a bigger ask
  - Fund moving some significant part of Samba to (eg) Rust
  - Lock down replication more strictly than Windows
    - (some kind of 2FA for DCs)

Allow pitching security issues to trusted clients?

Perhaps a bridge too far...

Being able to ask clients for funding for a specific issue

(rather than funding work on unspecified issues)

Beware the perverse incentives however!

Perhaps just to existing “vendors”?

Samba Commercial support, or Samba donations?

- Samba Commercial support and development funds Samba security
  - Not donations in general: these pay for CI testing and travel
- Impractical to fund day-to-day: Donations were around 30,000 USD per year
  - “Historical” CVEs over the last two year could cost ~95,000 USD per year
    - (rough calculations using samba.plus shop SerNet rates)

- Should the Samba Team fund “catching a whale”?
  - This is the opposite, impractical to fund commercially!
  - Significant **people-space** challenges when open source projects pay developers however.

Support your commercial support vendor

Vendors

Employ experienced Samba developers to help with the next 0-day

Samba commercial support

Ask your vendor for a support package including upstream Samba security.

Major world government?

Please fund Samba security specifically!

Buy a package subscription

If security matters, getting the fix on-time every-time matters also!

Thanks and Questions!

abartlet@catalyst.net.nz

Catalyst: Samba Development and Support

https://catalyst.net.net/services/samba

catalyst

Thanks to Bruce Bartlett for the lovely photos of the NSW Far South Coast