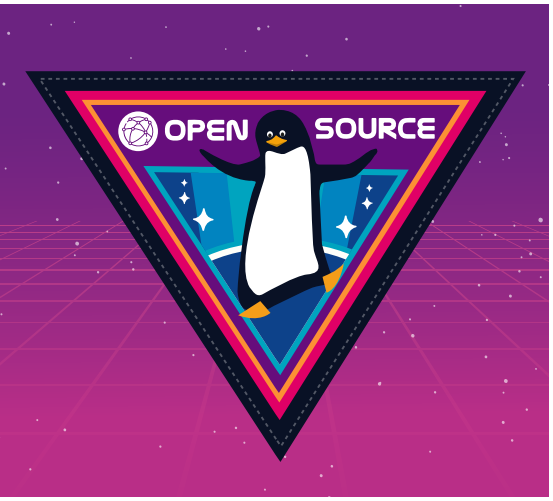

Verteilte Entwicklung im Samba-Projekt

In seiner über dreißigjährigen Geschichte stand das Samba-Team immer wieder vor der Herausforderung, kommerzielle – auch konträre – und Open-Source-Interessen in einem Projekt zu integrieren. Doch mit welchen Methoden funktioniert das überhaupt?

Von Volker Lendecke

Das Open-Source-Projekt Samba gilt als die Standardsuite von Programmen für die Interoperabilität zwischen Windows und Linux/Unix. Das ist ein riesiger Markt potenzieller Clients, die sowohl kommerziell als auch für die Entwickler persönlich interessant sind. Daraus ergibt sich aber auch, dass viele Kontributoren an Samba beteiligt sind, die sich untereinander auf die Ausrichtung des Projekts verständigen müssen.

Eine kurze Anekdote verdeutlicht, was das Projekt bis heute zusammenhält: 1991 hatte Andrew Tridgell das Problem, dass er an seinem Rechner unter MS-DOS Dateien gleichzeitig mit einem Server unter DES Pathworks und einer Sun-Workstation austauschen wollte. Sowohl der Pathworks- als auch der PC-NFS-Client brachten ihre eigenen, nicht kompatiblen TCP/IP-Stacks mit. Damit war nur entweder Pathworks oder PC-NFS möglich. So schrieb er einen Serverprozess, der den Ultrix-Pathworks-Server auf der Sun-Maschine emuliert. Dass der Pathworks-Server vom OS/2 LAN Manager abstammte, der später auch in Windows für den Austausch von Dateien sorgen sollte, war Tridgell nicht klar. Das führte aber dazu, dass Samba, das damals noch nicht so hieß, von Anfang an kompatibel zu sehr vielen Clients war.

Es ist Andrew Tridgell zu verdanken, dass er seinen „Server 1.0“ für eine Community geöffnet hat. Mit dem Samba-Team hat er eine Struktur geschaffen, die es anderen Interessierten ermöglicht, gleichberechtigt am Projekt mitzuwirken. Andrew hat sich mittlerweile aus der aktiven Entwicklung zurückgezogen, aber das Samba-Team funktioniert weiter. Meine persönliche Geschichte mit Samba begann, als ich den NetWare-3-Server im elterlichen Betrieb durch etwas ersetzen konnte, das ich selbst kompiliert habe.

Wie das Samba-Team heute arbeitet

Typische Arbeiten an Samba sind neben dem Entwickeln neuer Features vor allem das Beheben von Bugs. Dazu dienen neben dem Communitykanal für Samba, dem für alle offenen bugzilla.samba.org, auch die Samba-Mailinglisten. Diese Kanäle basieren komplett auf freiwilligen Beiträgen, was mehr oder weniger gut funktioniert. Sehr offensichtliche und einfach zu behebende Fehler sind meistens schnell gelöst. Wird es aber komplizierter, bekommen Community-Bugs oft nicht die Aufmerksamkeit, die Anwender sich wünschen.

Dann kommt Geld ins Spiel: Betrachtet man bei GitLab die Beitragenden zu Samba, zeigt sich, dass Angestellte der Firmen Catalyst, IBM (Red Hat), SerNet und SUSE (in alphabetischer Reihenfolge) den größten Teil der Patches zum Projekt beisteuern. Diese Firmen haben alle ein kommerzielles Interesse an Samba und stehen teilweise im Wettbewerb miteinander. Google, Arbeitgeber des prominenten Samba-Kontributors Jeremy Allison, ist ein Sonderfall: Das kommerzielle Interesse einer IBM an Samba ist durch ihr Produkt Red Hat Enterprise Linux viel direkter zu begründen als die Vorteile, die Google als Unternehmen von seinem Engagement hat.

Die vier zuvor genannten Firmen bieten Unterstützung bei Problemen mit Samba jedoch gegen Geld an. Auch andere Firmen können sich auf www.samba.org unter dem Menüpunkt „Support Samba“ eintragen lassen. Meldet ein Kunde bei einer der dort eingetragen Firmen ein Problem mit Samba, setzt sich ein Entwickler an die Arbeit. Da Samba sehr komplex und vielfältig ist, ist es für einen einzelnen Entwickler jedoch unmöglich, alles komplett zu überblicken. Daraus ergibt sich eine zentrale Frage, die dieser Artikel näher beleuchtet: die Kommunikation innerhalb des Teams. Wen fragt man als Entwickler, wenn man mit einem Problem nicht weiterkommt, und auf welchem Weg?

iX-TRACT

- Im Samba-Projekt mussten die Entwickler schon immer persönliche und kommerzielle Interessen unter einen Hut bringen.
- Ihre Maxime für Veränderungen lautet: Funktionierender Code steht über allem.
- Für die Kommunikation untereinander nutzen die Teammitglieder alle verfügbaren Kanäle und Medien.

30 Jahre gewachsene Komplexität

Die Komplexität von Samba ergibt sich daraus, dass es seit den frühen Anfängen als reiner SMB-Server in den letzten 30 Jahren zu einer Familie von Komponenten mit eigenen Implementierungen fast aller in der Active-Directory-Welt wichtigen Protokolle geworden ist: Angefangen bei DNS, LDAP und Kerberos über

SMB1/2/3 bis hin zu MS-RPC mit seinen Dutzenden Unterprotokollen bedient Samba Clients mit eigenen Servern. Nicht jeder dieser Server kann es bezüglich Leistung oder Skalierbarkeit mit Alternativen auch nur ansatzweise aufnehmen.

Ein gut getunter OpenLDAP-Server tanzt beispielsweise Kreise um einen Samba-LDAP-Server, wenn es um Speicherbedarf oder Anfragen pro Sekunde geht. Bei diesen „Make or buy"-Entscheidungen ist allerdings die Kompatibilität zu den existierenden Clients der wichtigste Faktor: Microsoft AD ist bei den Standardprotokollen wie LDAP und Kerberos sehr kompatibel zu den RFCs, nutzt die Erweiterungsmöglichkeiten dieser Protokolle aber auch voll aus.

Um herauszufinden, wen man bei der Codearbeit bei Problemen im Einzelfall befragen kann, dienen in `git` die Befehle `git blame` und `git whatchanged`. Für jede Zeile im Code gibt `git` dann aus, wer sie zuletzt geändert hat, und für jede Datei ist die komplette Versionshistorie verfügbar. Daraus erkennt man als Entwickler sehr gut, wer der Experte für einen bestimmten Teil des Codes sein könnte.

Aus dieser mikroskopischen Sicht wird aber oft nicht klar, warum ein Teil des Codes auf eine bestimmte Art und Weise funktioniert. Ist ein unverständliches Stück Code also aus guten Gründen so komplex, wie es ist, oder ist die Komplexität eher ein Unfall? Die Frage „Ist das Kunst oder kann das weg?" ist bei Samba an sehr vielen Stellen relevanter, als Außenstehende vielleicht denken würden.

Wer weiß denn so was?

Um solche Designfragen zu diskutieren, braucht man als Entwickler Ansprechpartner. Für Insider ist es ziemlich einfach, den oder die Richtigen zu finden: Das Samba-Team ist ein ziemlich alter, untereinander gut vernetzter Club. Von den Top 20 der laut openhub.net Beitragenden des letzten Jahres sind 16 seit mehr als fünf Jahren dabei, 11 mehr als 10 Jahre und drei sogar mehr als 20 Jahre aktiv.

Bevor die Coronapandemie es unmöglich machte, hat sich ein großer Teil des Samba-Teams – wer auch immer Zeit hatte – zweimal im Jahr persönlich getroffen. Einmal im Frühjahr auf der SambaXP in Göttingen und einmal im Herbst auf der SNIA SDC in Santa Clara, Kalifornien. Zu diesen Gelegenheiten werden die wenigen Dinge besprochen, die das Samba-Team als Organisation in der Software Freedom Conservancy SFC entscheiden muss, und man lernt sich bei Pedro's „Chips and Salsa" in Sichtweite von Intel besser kennen.

Die SambaXP in Göttingen ist eines der persönlichen Treffen für das Samba-Team, hier ein Bild aus dem Jahr 2012.

Wenn man sich persönlich kennt, führt rein elektronische Kommunikation viel seltener zu Missverständnissen. Smileys können in keinem Fall ein echtes Lächeln oder einen erschreckten Blick ersetzen, sondern bestenfalls ergänzen.

Wer bereits viele Jahre lang an Samba arbeitet und dabei auch die Arbeiten der Teamkollegen verfolgt, weiß also, wer für welches Fachgebiet der richtige Ansprechpartner ist. Dennoch spricht man zunächst meist mit den Kollegen innerhalb der Firma, mit denen man ohnehin regelmäßige Teammeetings hat.

Die Wahl der Kommunikationsmittel ist dabei flexibel: Verwendet wird, was gerade verfügbar ist – E-Mail, Chat, Telefon oder Videokonferenzen. Beim Chat nutzte das Team früher IRC auf freenode oder einem eigenen IRC-Server, heute kommt das Projekt Matrix zum Einsatz. Für die direkten telefonischen Anrufe ist es dank angrenzender Zeitzonen durchaus praktisch, dass sich die meisten Samba-Entwickler in zwei Hotspots befinden: Australien und Neuseeland sowie Deutschland.

Änderungen an Samba

Aus technischer Sicht wird man ein Mitglied des Samba-Teams, wenn man einen Account auf dem Server besitzt, der im Samba-Team die Continuous Integration (CI) durchführt und nach erfolgreicher CI die Patches in das Master-Repository überführt.

Heutzutage würde man dafür vermutlich GitLab verwenden. Samba hat CI aber schon praktiziert, bevor es den Begriff gab: Bei Samba hieß das, was sich heute CI nennt, autobuild und build farm. Das autobuild-System gibt es heute noch und es bildet die Basis für CI-Pipelines auf GitLab, die die build farm ersetzt haben.

Jeder Patch in Samba muss von zwei Mitgliedern des Samba-Teams begutachtet werden. Das heißt, wenn jemand aus dem Team einen Patch schreibt, muss er ein weiteres Mitglied finden, das ein „Reviewed-by:" beisteuert. Im Prinzip gilt dieselbe Regel für Beiträge von Nichtmitgliedern: Auch hier müssen zwei Teammitglieder ihr „Reviewed-by:" abgeben.

Bis vor wenigen Jahren durchlief ein Patch an Samba einen ähnlichen Weg wie beim Linux-Kernel: Er ging per Mail an die Liste samba-technical. Dort wurde er öffentlich kommentiert, begutachtet und dann ins autobuild-System des Samba-Teams geschickt. Außer bei sehr einfachen Patches ist es aber heutzutage quasi nicht mehr möglich, einen Patch beim ersten Versuch durch die vielen Tausend Tests des autobuild-Systems zu bringen. Das bedeutet für das Samba-Team, dass man sich mit dem Push eines Patches von Externen potenziell viel Arbeit aufhalst, weil man dem Entwickler des Patches mindestens die Fehler im autobuild kommunizieren muss.

Von Externen einen Lauf durch das autobuild zu verlangen, ist auch nicht so einfach: Der autobuild-Server des Samba-Teams mit 32 Kernen und 128 GByte RAM benötigt bis zu zwei Stunden für einen kompletten Durchlauf. Das sind für heutige Verhältnisse zwar keine wirklich großen Anforderungen an Server, jedoch deutlich mehr Ressourcen, als man typischerweise in einem Laptop hat.

Ein weiterer Nachteil des Ansatzes des Patch-Zyklus per Mailingliste ist, dass Patches in Vergessenheit geraten können, wenn sich niemand aktiv darum kümmert. Schließlich gibt es keine zentrale Liste von Patches, die auf Kommentare warten. Für neue Beitragende kann es frustrierend und abschreckend sein, immer wieder auf eigene Arbeiten hinweisen zu müssen.

Neue Wege über GitLab

Um dem Samba-Team Arbeit abzunehmen und die Hürden für externe Beitra-

gende zu verringern, hat das Team begonnen, auf GitHub eine Präsenz aufzubauen und die dortige CI-Infrastruktur zu nutzen. Bedenken bezüglich der Freiheit der Infrastruktur haben dann dazu geführt, die Präsenz auf GitLab zu fokussieren.

Möchte heutzutage jemand Patches zu Samba beitragen, geht der Weg also über einen Fork auf GitLab. Da die CI-Pipeline von Samba jedoch viel zu groß für kostenlose Accounts ist, bietet das Team die Möglichkeit, Beiträge in ein Repository des Samba-Teams zu pushen. Die Berechtigung zu diesem Repository kann das Team relativ freizügig vergeben, weil das Master-Repository nicht auf GitLab liegt, sondern auf dem internen CI-Server, auf den nur Samba-Teammitglieder Zugriff haben.

Sobald die Pipeline nach einem Push durchlaufen ist, wird ein Merge Request erstellt, der dann kommentiert wird. Stand März 2022 gibt es jedoch noch keinen Automatismus, mit dem GitLab direkt ins Samba-Master-Repository pushen könnte. Diesen Schritt muss ein Samba-Teammitglied weiterhin manuell durchführen.

Konfliktdiskussion und -lösung

Natürlich hat auch Samba seine drei Jahrzehnte der Entwicklung nicht konfliktfrei hinter sich gebracht. Wenn man den einen Mechanismus zur Lösung von Konflikten benennen sollte, dann ist es, dass funktionierender Code über allem steht. Samba hat enorm viel funktionierenden Code, den zu verbessern oder gar zu ersetzen eine extrem große Hürde sein kann. Als Beispiele dienen zwei Konflikte bei der Entwicklung von Samba: Samba TNG und Samba 3/4.

Samba TNG war ein Versuch, die auf Distributed Computing Environment / Remote Procedure Calls (DCE-RPC) basierende Infrastruktur in Samba an die interne Windows-Struktur anzupassen. Ein Windows-Domänenmitglied spricht mit seinem Domänencontroller über die DCE-RPC-Protokollfamilie. DCE-RPC ist ein Konkurrenzprodukt zu dem von Sun initiierten ONC-RPC (Open Network Computing), auf dem NFS und NIS basieren, und hat mit SMB zunächst überhaupt nichts zu tun. Zum Implementieren einer zu NetWare Bindery analogen Benutzerdatenbank brauchte Microsoft einen flexiblen Rahmen, um komplexe Strukturen einfach über ein Netzwerk zu übertragen. DCE-RPC bot sich an, weil es nicht auf TCP/IP als Transportprotokoll festgelegt war, sondern flexibel auch mit damals aktuellen Protokollen wie IPX und SMB über NetBEUI arbeiten konnte. Um Mitglied einer Windows-Domäne zu werden oder sogar einen Windows-Domänencontroller zu implementieren, musste Samba also DCE-RPC implementieren.

Eine der treibenden Kräfte hinter der Entwicklung von DCE-RPC in Samba war Luke Kenneth Casson Leighton, der darüber sogar ein Buch geschrieben hat (siehe „Quellen“). Seine Entwicklungen haben wesentlich dazu beigetragen, dass Samba in Windows-Domänen als Mitglied und Domänencontroller teilnehmen kann.

Von einigen seiner Ideen konnte er den Rest des Samba-Teams jedoch nicht überzeugen, weil sie ihrer Zeit zu weit voraus waren. Dies ist in einer Zeit passiert, als ein Fork eines Projekts noch mehr als ein Klick auf einer GitHub-Seite war. So entstand Samba TNG, das Luke zusammen mit ein paar Mitstreitern ein paar Jahre gepflegt hat. Am Ende konnte Samba TNG nicht genug Ressourcen aufbringen und wurde nicht mehr weitergeführt.

Luke hatte zwar die richtigen Ideen, aber sie waren zu ihrer Zeit sehr radikal, und Samba funktionierte bereits als Domänencontroller. Dass die Architektur der RPC-Dienste komplett „falsch“, also anders als bei Windows war, war in der Bewertung weniger wichtig als leidlich funktionierender Code. Erst mit Version 4.16 verwirklicht das Samba-Team mehr als 20 Jahre später die wesentlichen Ideen von Samba TNG: das Bereitstellen von MS-RPC-Servern in separaten Programmen, die über Sockets kommunizieren.

Ein zweiter SMB-Server

Die zweite große Entwicklung, die am Ende in einer Sackgasse geendet hat, war Andrew Tridgells Versuch, den SMB-Server von Grund auf neu zu schreiben. Nach Jahren der Entwicklung an Samba hatte er angefangen, auf der Basis seiner Erfahrungen mit dem SMB-Protokoll einen neuen SMB-Server mit verbesserter Architektur zu entwickeln. Einen, der asynchron arbeitet und vor allem das SMB-Protokoll korrekt implementiert. Da das damals noch aktuelle SMB1 bis heute nicht zufriedenstellend dokumentiert ist – und es wohl auch nie mehr werden wird –, hat er mit großem Aufwand Tests geschrieben. Die haben jeden nur möglichen Aspekt des Protokolls gegen existierende Windows-Server geprüft, um sie dann in seinem neuen Server genau so zu implementieren.

Andrews Entwicklungen haben zu einer funktionierenden zweiten Implementierung eines SMB-Servers geführt, die bis heute im Samba-Quellcode vorhanden ist, in jedem Entwickler-Build mit gebaut und in Tests benutzt wird. Jedoch hat es diese zweite Umsetzung nicht geschafft, den existierenden Code vollständig zu ersetzen, obwohl Andrew versucht hat, den Rest des Samba-Teams von der zweifellos viel besseren Architektur zu überzeugen.

Auch hier liegt der Grund darin, dass der existierende Code leidlich funktioniert hat und zu viele Benutzer davon abhängig waren. Also auch hier: Funktionierender Code steht über allem.

Im Laufe der Jahre hat sich Andrews Fokus vom korrekten SMB-Server zunächst in Richtung einer vollständigen Testsuite für das SMB-Protokoll entwickelt. In diesem Rahmen hat er auch das mit Vista eingeführte SMB2-Protokoll decodiert und implementiert. Noch etwas später hat er zusammen mit anderen eigene DNS- und LDAP-Server entwickelt und diese mit dem Key Distribution Center (KDC) Heimdal zu einem Active Directory Domain Controller zusammengeführt. Dieser AD-Controller hat als solcher funktioniert, nur leider war er architektonisch eng an seinen neuen SMB-Server gebunden. Der SMB-Server-Fraktion des Samba-Teams erschien es zu riskant, den funktionierenden `smbd` zu ersetzen. So führte das Team beide Entwicklungszweige wieder zusammen, um ein geschlossenes Produkt zu präsentieren.

Fazit

Samba ist zwar ein sehr komplexes Projekt, hat aber eine sehr einfache Ausrichtung: Kompatibilität zu existierenden Clients und Servern. Die Frage nach richtigem oder falschem Code ist damit einfach zu beantworten: Funktioniert es mit Windows oder macOS und Linux-Clients? Innerhalb dieses Rahmens gibt es genügend Raum für Entwicklungen, die auch mal in einer Sackgasse enden können. (avr@ix.de)

Quellen

Luke Kenneth Casson Leighton;
DCE/RPC Over SMB: Samba and Windows NT Domain Internals;
Macmillan Technical, 2000

Volker Lendecke

ist Mitglied des Samba-Teams und Mitbegründer der Service Network GmbH in Göttingen.