A clipboard for your tailnet with Telltail: Q&A with developer Ajit Singh

Parker Higgins (he/him) Tailscalar on April 6, 2023

We love cool projects that people build on top of Tailscale, so when we first saw Telltail — a mechanism for sharing a text clipboard between different machines on your tailnet — we had to know more. And we were excited when Ajit Singh, the developer behind Telltail, put together a great blog post walking through the development process and showing how he iterated in steps towards the final product that he published.

We asked Ajit a few questions about Telltail, why he wanted it, how he built it, and what he learned along the way. We hope you enjoy reading his answers here — and if they inspire you to build something fun with Tailscale that you want to share with the community, shoot us a note at devrel@tailscale.com and let us know more!

So first things first, can you tell me a little about yourself? Where are you based, and do you do programming professionally there? What kind of programming do you like to do?

Hi! I'm Ajit Singh. I'm from India, and I work for a software company where I build frontend apps.

Outside my job, rather than writing code, I like to learn about alternate possibilities in interfaces that exist or used to exist, like Emacs, Acme and Leap for editing text, the effort to bring a smartphone a decade earlier by General Magic, and this interaction in HTC HD2.

How'd you run into the problem you solved with Telltail? Was there a particular moment when you thought: I have to build a solution to this?

Photo courtesy of Ajit Singh.

Given the nature of my work, I happen to be in front of a screen quite often. Thankfully I have only two: a laptop running Fedora and an iPhone. But this means I switch between them frequently. It's quite usual for me to browse social media on my phone, finding and sharing a link of a post to my laptop so I can read on a bigger screen later. Many apps exist that let you transfer files to another device, but I never was able to find first class support for sharing text in any app. In the ones you'd find some support, you'd always have to open that app in both of my devices, then wait for them to discover each other (or you manually put one device's IP in another), but even then the connection could fail on you. I was surprised to see that none of the app offers to do it efficiently, given that even Apple has already introduced it as a feature.

When I downloaded Tailscale, I didn't do it for the purpose of solving a problem, I only downloaded it out of interest. I started checking what I can do with it — accessing a website served on another device with a name, SSHing with my phone (which will save me someday too), and then Taildrop to transfer files. The existence of Taildrop made me look if Tailscale supports sharing text snippets as well, and I came across this feature request.

Now that I knew that serving a website across my devices is easy, and as I was already going through gobyexample.com that time, I thought of making a simple web app using Go that could store and serve a text snippet. I then started building conveniences on top of it — accessing its APIs via Shortcuts on iPhone, then binding keyboard shortcuts to interact with it on laptop (which I later changed to automatically doing it with usual keyboard shortcuts that I named Sync), and then finally making an installer. Building this way meant that I could rollback easily if I feel the approach isn't right, and I could stop whenever I'm satisfied with the outcome. And because it is built incrementally, it wasn't until quite later when I figured I could make a proper solution out of it.

It's very cool that you show your iterative approach to building the tool in your blog post. Did you have any moments where you got frustrated and almost decided that the solution was "good enough" where it was? How did you find your way out of it?

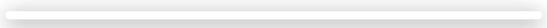
Yeah! It did happen numerous times. Because I only use two devices, there wasn't really an incentive to build Telltail for other OSes. But my curiosity pulled me into looking for a solution for them.

Another instance was when I started documenting the process to configure Telltail so that somebody could use it for themself if they want to. In the middle of writing it I said to myself, "These are so many instructions to follow that a person would give up midway, and documenting all this in such a way it would be easy grasp is a chore too." So I started working on a CLI that could install Telltail. After completing it, I weent back to replace the instructions I'd written, and I chuckled when I saw that I'd been asking the user to download Go and compile the program on their machine, and now I have a CLI that sets up everything with two lines of commands.

There were other times when I concluded the solution I have is good enough. In Windows, to watch for any changes made in the clipboard, I was able to find a blog post which showed an example with Python. Tweaking and then using it resulted in a larger executable size than I would've liked. To reduce it, I'd have look into writing it using C#, something I wasn't willing to commit to.

I also didn't like the initial size of an executable. Thanks to the internet, I've found ways with which I could compress executables to an acceptable size. But applying those tricks changes them in such a way that makes Windows anti-virus softwares believe it is malware. This is a known issue for programs built in Golang and Python for Windows. I wish this issue didn't exist.

One more: upon telling about what I built to a person I know, they suggested that I could run Linux on my home router and put Telltail Center (a web server) on it so that it keeps running even my laptop is closed. Cool idea but it is something I'm not going to do, haha.



Telltail in action. This version has been sped up.

You mention at the bottom of your post that this was your first time working with VPNs, systemd, golang, and more. That's amazing! What are some of the biggest things you learned about working with these tools for a project?

Using Tailscale is surprisingly easy and it has most things I needed. Like once I was worried about how I would be able to install SSL certificates in my phone, and then I later discovered that Tailscale can, with a command, assign certs for a device. The only thing I couldn't do is to get the tailnet name using auth key, tsnet and iOS Shortcuts. It would've made the Telltail installation even easier. Even without it, the experience was good overall.

Thanks to xclip and browsers' Clipboard API, I was able to dissect and understand what happens when you copy an element in Figma, or a file in your file explorer, or an image in a browser. The TL;DR is when you copy an image from a browser, the browser can tell the clipboard "Here's image data and here's a link, and please store both of them." The keys against which the clipboard will store these formats can vary from OS to OS.

Even learning about a small thing like clipboard doesn't seem to have an end. Linux (X11) has three types of clipboards. In Windows you can have at max 16,000 formats. I noped out of nerdwaters once I read that.

There isn't a universal way to mark a clipboard item so that clipboard history managers don't record it, something I wanted to add to Telltail.

Golang is good. I like that I can have separate files for each platform. Cross-platform compilation came in extremely handy. On the other side, struct fields being optional tripped me once in the code. Decoding JSON and getting a value back for something that wasn't present in the encoded form felt odd to me. Hopefully I could find linters and checkers that could assist with these.

If people want to follow any future development on Telltail or other things you're working on, what's the best way to find you online?

I'm @hemarkable on Twitter and @ajit@hachyderm.io on Mastodon. I'd post updates and tell about new things there.

Finally, I noticed your site has a "last listened to" section. Do you have any favorite music to play while you're hacking?

I don't listen to music if I'm really focusing on something. Otherwise in day-to-day coding, I listen to my music library that ranges across genres: R&B and Soul (Bill Withers, dvsn, Daniel Caesar), Jazz (Chet Baker, BadBadNotGood) Electronic (Kaytranada, SG Lewis, Zedd), Rock (Kaleo, Royal Blood, The Black Keys), Pop (Zayn, Valley, The 1975), and obviously Indian (Bollywood, Punjabi and Indi-pop).

Also John Mayer, though I'm not sure what genre I would put him into. Usually one of the genre remains in heavy rotation at a time, and right now it is Electronic.