Review: Helm Personal Server gets email self-hosting (almost) exactly right



Enlarge / The Helm Personal Server, *in situ* in my office during the review.

reader comments

104 with 55 posters participating, including story author

Share this story

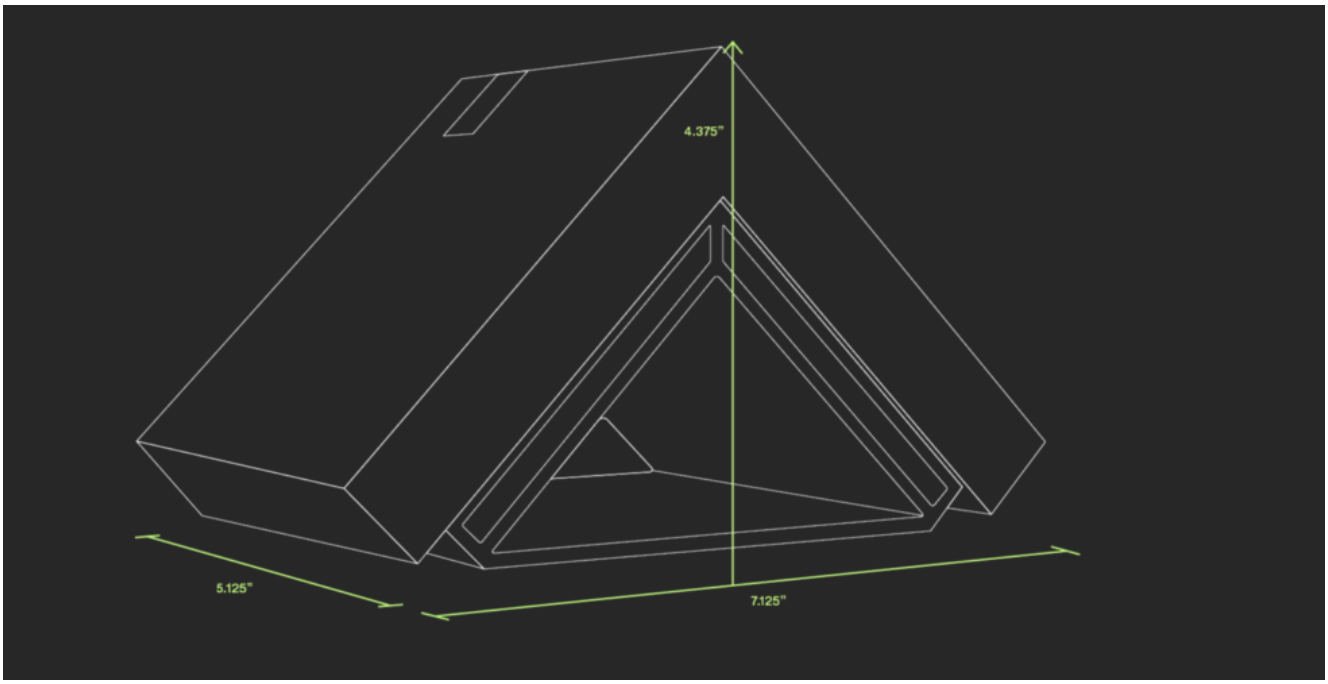
Specs at a glance: Helm Personal Server

CPU	Quad-core 1.6GHz ARM Cortex-A72 w/TrustZone crypto module
RAM	2GB ECC
Storage	128GB NVMe SSD w/256-bit AES-XTS encryption
Connectivity	802.11ac/a/b/g/n, Bluetooth 4.2, Gigabit Ethernet, 2x USB-C 3.0
Dimensions	111.1mm x 180.9mm x 130.1mm (4.375″ x 7.125″ x 5.125″)
Price	$499.99 at the Helm store (plus $99/year subscription, waived for first year)

As Ars security-meister Dan Goodin noted in his initial write-up back in October, the Helm Personal Server is a small-ish ARM-based email server that sits in your home and does for you what Gmail or Outlook.com or whomever your current email provider does for you. It’s a full-featured, single-domain (for now) MTA in a box that you can use with an unlimited number of email addresses and accounts, and it gives you 128GB of space to use as a mail store for those accounts. It also gives you CalDAV calendaring, notes, and CardDAV contacts, and it does it all with open-source applications that are chosen and configured in a way that demonstrates a solid bias toward individual security and privacy.

And I like it. I like it a lot. I didn't think I would, but after spending a week with the device, I'm *almost* ready to spring for one—almost. And that's high praise, coming from a paranoid email self-hoster like me.

Based on my short time with the Personal Server, the praise is properly earned. The Helm team based its product mostly around the same mail stack that I personally prefer and use—the holy trinity of Postfix for SMTP, Dovecot for IMAP, and SpamAssassin for keeping things clean. The device properly uses SPF, DKIM, and DMARC—and handles all the DNS stuff necessary to make those things work. End-user data is smartly encrypted at rest and in flight. Clever use of tunneling to AWS-based gateways transparently works around common ISP blocks on email service ports. And, perhaps most importantly, you don't have to know what any of that stuff means to use the device securely—casual folks who maybe just want to lessen their reliance on Google or Microsoft will find the transition to Helm relatively painless, and there aren't many ways to screw it up and make yourself less secure.



Enlarge / Here's how big the thing is.

But technical users might balk at some of its shortcomings and annoyances—the underlying Postfix/Dovecot configuration (along with the constellation of smaller apps like OpenDKIM that are

necessary to make it work) can’t be viewed, changed, or edited. If you’re bringing an existing domain to the Helm service, you currently need to transfer your domain’s authoritative DNS to Helm’s AWS-based DNS servers so that the service can manage the necessary MX and TXT records. The choice of AWS for DNS means Helm currently doesn’t offer DNSSEC support. And a few other minor issues might make experienced email sysadmins hesitant—though conversations with Helm’s support team during the review have me convinced of the company’s willingness to evolve the product in a direction more compatible with the (sometimes difficult) demands of power users like me.

All that being said, the tl;dr here is that I like Helm. I like what the company is doing, and I like the way it’s doing it. With a few minor changes (and some more exposed knobs and levers so I can tweak things a bit), I’d happily buy a device and transition my email hosting off of my current setup.

Now let’s dig a little deeper into what we’ve got here.

How Helm fits into the email-hosting dilemma

In early 2014, I penned a four-part series about how to host your own email for your own domain, based on my own adventures in self-hosting. Although the guides are at this point in dire need of updating, they’re among the most popular things I’ve written in my entire tenure at Ars (eclipsed only by the time I talked non-stop about farts for a whole week). Doing my own email hosting has on the whole been a rewarding and challenging endeavor that’s yielded tremendous amounts of knowledge, experience, and gray hair—and, after doing it for a bit more than five years, I have no plans to stop.

There are significant benefits to self-hosting your email, but they come with significant downsides, too—most notably, you’re

on the hook for any mistakes or problems. “Email,” said Past Lee, “is like a puppy, and once you step up and own your own puppy, you’ve got to take care of it, clean up after it, and make sure evil people don’t infect it with horrible viruses and transform it into a zombie.” Looking after an email server does occasionally require work—a responsible sysadmin needs to keep up with updates, keep an eye on the log files, check regularly on RBLs, be mindful of deliverability and sender reputation, and other miscellaneous sysadmin-y tasks. It’s not overly onerous, but it’s not hands-off, either.

Helm aims to give you the best of both worlds—the assurance of having a device filled with sensitive information physically under your control, but with almost all of the heavy sysadmin lifting done for you.

The primary difficulty Helm will face here is the marketing message—who is this thing for? Most folks in the angry graybeard set (of which I count myself a member) are either already self-hosting or have dismissed self-hosting as far more trouble than it’s worth; the casual-user set doesn’t really think much about how email works or what “hosting” really means.

Helm therefore has decided in its marketing literature to lean heavily on the privacy aspects of self-hosting. Helm’s website leads with that message and plays up how switching to Helm is a way to take ownership of one’s online identity and divorce oneself from technological dependence on giant data-hungry corporations:

> *Your most critical data (like emails, search history, passwords, photos, and videos) is stored on a massive corporate server outside your home.*
>
> *Increasingly, this leaves you vulnerable to hacks, companies profiting from your data and online behavior, and mass government surveillance.*

It’s a marketing message not without some necessary caveats—as long as you’re exchanging emails with *other* people who use those big corporate services, you’re just as vulnerable to mass surveillance and data harvesting as before, since marketeers (along with any interested three-letter agency) will simply vacuum up your message on the receiving end rather than the sending end. And Helm’s usage of AWS for so much of its infrastructure—even with a responsible eye toward encryption and keeping sensitive data properly siloed—means you’re still depending heavily on one of the *most* data-hungry corporations of them all.

The message, however, isn’t wholly inaccurate—you’re better off self-hosting. It remains an open question, though, of whether the average consumer (or even the average tech-savvy Ars reader) would be willing to spend $499 and an additional $99 each year for a nebulous and difficult-to-fully-quantify increase in security. Convincing people to do so will be Helm’s greatest challenge—probably a great deal harder than actually designing the service in the first place.

The Helm service—or, “where does my $99 a year go?”

I want to talk just for a moment about exactly what that ongoing $99/year charge pays for. Obviously, Helm has employees, and they need to make payroll; the Helm service requires ongoing work behind the scenes for it to remain long-term functional. But the subscription fee also covers a fair amount of real AWS-based infrastructure costs that come with each Helm device sold.

Helm utilizes Amazon’s Route 53 DNS service on the back-end, which gives you a robust DNS setup with a large number of fast, geographically distributed resolvers. To work around ISP restrictions on ports and to do away with the Sisyphean task of trying to use a residential ISP IP addresses for email

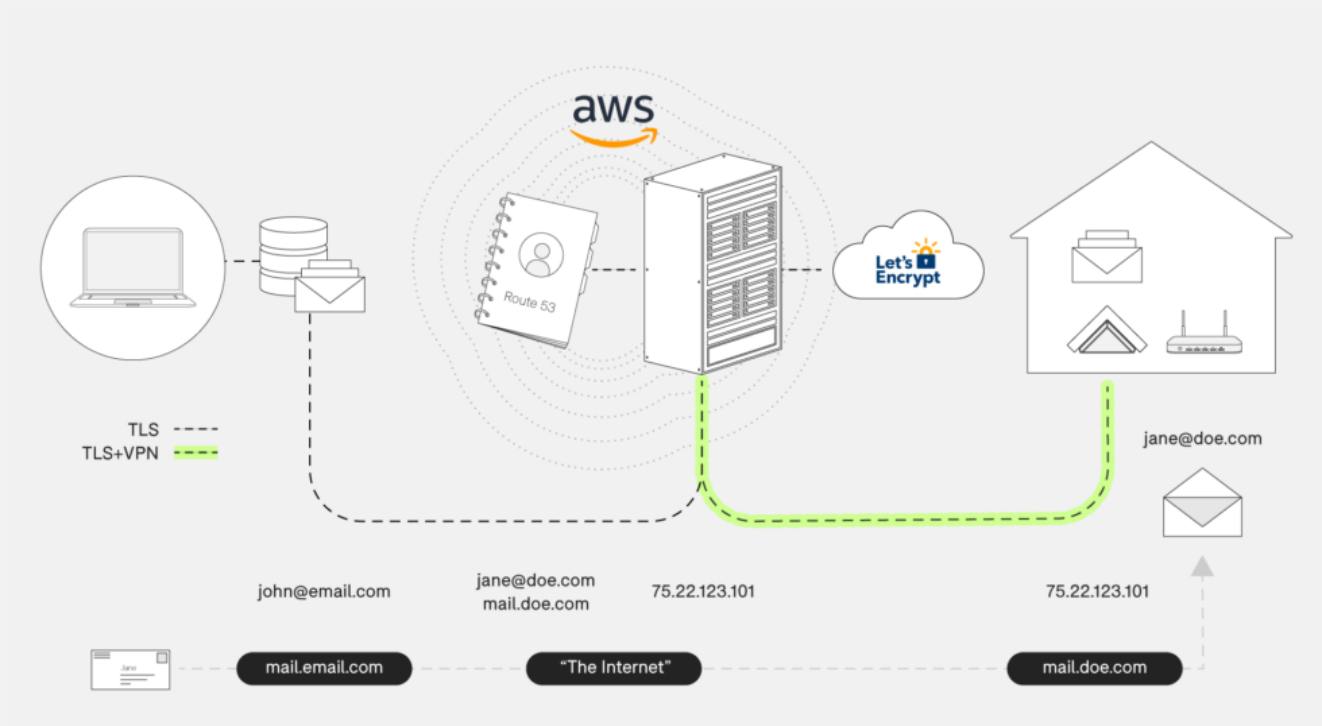
service, the company spins up Amazon EC2-based gateway machines that establish tunnels to the residential Helm devices. “All the gateway does is forward packets back and forth,” explained Helm CEO and co-founder Giri Sreenivas to Ars. “All TLS terminates on this device. All we’ve done is introduce an extra hop on the Internet. We’re funneling encrypted traffic.”

Helm also puts some care into the AWS IP addresses assigned to the Helm gateway, doing all the necessary legwork to vet those addresses against the ever-changing list of anti-spam IP address blacklists utilized by most email servers.

The AWS costs for Helm also include built-in cloud redundancy; the company uses Amazon’s US-West-2 region for its gateways and keeps machines in all three of the region’s availability zones. The company uses a separate region, US-East-1, for all of its data storage—that is, the place where your gigabytes of encrypted email back-ups live.

The mail flow

Those EC2-based gateways are one of the keys to making Helm work. When your local Helm server powers on, it establishes an IKEv2-based tunnel to an EC2 gateway. That EC2 instance is assigned the public IP address referenced in your domain’s MX records, and utilizes good ol’ iptables to forward select packets through the tunnel to your Helm server.



Enlarge / A visual overview of the mail delivery path with Helm.

The path is the same whether you're an IMAP client checking your inbox or another email server transmitting a message over SMTP—everything goes via the gateway.

Interestingly, this means that email clients on the same LAN as the Helm server still run their traffic across the Internet, through the tunnel by way of AWS. When you configure your client, you point it at "helm.yourdomain.com" for SMTP and IMAP and forward DNS lookups on that hostname will always return the AWS gateway's Internet IP address.

I asked Helm support about bypassing the tunnel and addressing the Helm server's mail ports directly on the LAN, which might be something a customer with split-horizon DNS would do. The response was reassuring: "This shouldn't be a problem," the support people said. "If the user sets up DNS to point directly to the local Helm IP address for local clients, those ports are currently open, and so it should work. This isn't a configuration that we have tested, however, but technically we don't see any issue with this."

(While the SMTP and IMAP ports are locally addressable,

there’s not much else you can actually do to the Helm server directly—there is no way to log into it. Configuration tasks are all done via an app, which we’ll get to shortly.)

The decision to wrap all server comms in a tunnel pleasantly sidesteps the two major issues that often come with trying to self-host an email server on a residential ISP connection. Namely, you don’t have to worry about your ISP blocking inbound connections on common email ports, and you don’t have to worry about the fact that most residential IP addresses are permanent residents on just about every blacklist ever. By shifting the connection point to Amazon’s cloud and being choosy with the IP address pool they have to work with, Helm saves you no shortage of headaches. (You also don’t have to worry about updating a DNS entry when/if your home IP address changes, but that’s mostly a solved problem at this point anyway.)

Unboxing and setup

The server comes packaged with a quickstart guide, a network cable, a good-quality branded AC adapter, a USB key for storing your backup encryption key(s), and a Helm sticker (our review device also came with a separate USB stick loaded with press assets and images).

Setup is pretty darn easy. You unbox the server, peel off the plastic protective film around it and its AC adapter, plug it in, and connect it to your LAN. The device has built-in 802.11ac and can be run wireless, ~~but you need to set things up wired first before you can configure Wi-Fi~~. **Update:** Helm notified me after the review went live that the initial setup can indeed be done wirelessly if desired.

After you’ve plugged in the network cable, you turn the thing on, install the mobile app on your iOS or Android device, and follow the steps.

- Here it is: your Helm Personal Server. Well, the box, at least.
  Lee Hutchinson
- The top of the box comes off, revealing a pointy package.
  Lee Hutchinson
- The box bottom contains the rest of the package: a network cable, a sticker, a quickstart guide, an AC adapter, and a very important USB stick for your encryption keys. The green PNY stick is just for reviewers and contains a presskit with photos (some of which are scattered through this review).
  Lee Hutchinson
- The Helm Personal Server, facing the camera.
  Lee Hutchinson
- The Helm Personal Server, facing away from the camera. Visible are two USB-C ports (one reserved for power, since the AC adapter terminates in a USB-C plug), power button/status light, and gigabit Ethernet.
  Lee Hutchinson
- The peak of Helm's roof contains a small cut-out housing another USB-C port—this one for attaching a stacking expansion unit (when/if one is eventually released).
  Lee Hutchinson
- One of the device's eaves houses a removable (but glued in place) 2.5″ SATA bay.
  Lee Hutchinson
- Apologies for the quality of this photo—it was annoyingly difficult to take. This is a peek into the 2.5″ SATA bay.
  Lee Hutchinson

Those steps, pictured in a gallery below, will see you first pairing your Helm server with your smartphone via Bluetooth to kick off the setup. The Helm server and your smartphone perform a token exchange via Bluetooth that enables your phone to continue being used for admin tasks; configuring new

smartphones to work with the Helm app requires physical proximity and another Bluetooth connection. (Configuring smartphones to send and receive email with Helm, though, does not—you only need to worry about the one-time Bluetooth connection for devices you're going to use to administer the server.)

It's important to pause and reiterate that, after the initial token exchange via Bluetooth, *all* communications between your management smartphone and the Helm server happens over the Internet via the AWS gateway. That's the case even if your phone and the Helm server are on the same LAN segment. This is actually a relief for folks like me who are perhaps more than a little paranoid about putting a device we can't manage or upgrade on a trusted LAN segment with other trusted devices—since there isn't any direct communication between your management device and the Helm server, there's no reason you can't permanently plop the Helm server on your IoT VLAN if you have one or even onto its own isolated segment.

Per a note from Helm support, the only thing the box needs to be able to do is send outbound traffic on UDP ports 500 and 4500 and on TCP port 443. You don't need any incoming port-forward rules.

The review units Helm sent out were preconfigured with test domains (I got "helmdomain21.com"), but if you're an actual customer, you'll either have purchased a domain from Helm when you bought the device or you'll have switched your existing domain's authoritative DNS over to Helm's DNS servers when you bought the device.

Regardless of how you got there, the setup continues by showing you the domain assigned to you and asking you to enter an activation code Helm sends you at purchase. After that, you create your first mailbox for the domain, which is also your administrator account.

The setup assistant then prompts you to insert that flashy silver USB stick into one of the Helm's ports. At this point, the server generates an encryption key pair and writes the private key to the stick. The encryption keys are used both for encrypting the Helm's local filesystem and also for encrypting the server's automated backups, which are automatically uploaded to AWS for you. If your Helm server dies or is stolen, you'll need this physical key to restore your email backups and configuration to a replacement Helm device.

- After unboxing your Helm and powering it on, you install the app and are greeted with this.
- You must pair your smartphone with the Helm server over Bluetooth to exchange a token. You must do this with any smartphone or tablet you want to use to admin your Helm server. (You do NOT need to do this if you just want to send and receive Helm email on a device—only if you want to install the Helm app and do admin stuff.)
- Next you enter an activation code and verify that the domain preconfigured on the server matches your email domain. (For reviewers, Helm sent out devices preconfigured with generic domains for testing purposes. I got "helmdomain21.com".)
- Then you create an admin account, which is also an email inbox.
- Give it a password and allow notifications (which you can disable as desired later).
- Then the setup process asks you to insert the silver USB stick that came with the server. Your Helm server's encryption key is written to the stick. Store it in a secure place: you'll need it if you ever have to restore a backup.
- Next—at least on iOS—the Helm server will deliver you some preconfigured profile files so that you don't have to manually add your Helm inbox to your phone.

- Tapping “install” kicks off the profile installation process.
- One profile, three accounts.
- And there they are.
- Tapping on the email account shows pre-populated account details.
- Calendar account details.
- Contacts account details.
- Once the profile is installed, you’re ready to start using the server.
- The app’s admin interface presents you with a list of additional tasks you might want to do.
- The email account shows up in your list of email accounts in the default iOS email client.
- Account details from the iOS settings screen.
- Calendar account details.
- Contacts account details.

Finally—at least on iOS, which is what I use—the application generates a set of profiles containing your Helm account’s email, calendar, and contacts settings. Then the application installs all of that for you, though you have to hit OK a few times to shepherd the process along. This is a nice convenience that eliminates you potentially needing to copy-and-paste (or, worse, write down and manually enter) a bunch of server and account settings.

Once the profiles are installed, you’re immediately able to send and receive email from your new domain. After that, the app has a set of wizard-like tasks in the main window to show you how to import email from external accounts, create more email addresses, and set per-device passwords on your Helm account so it can be accessed on other devices like a laptop or tablet.

https://arstechnica.com/?p=1421247