



Well-Architected for Amazon Connect — Building for What's Next

Amazon Connect User
Group - Americas
(Virtual)

Agenda

- This user group, this platform, me
- Well Architected: What is it, how we use it, why we saw a need, what the result was
- A demo
- Homework, and what comes next

How We Build Now

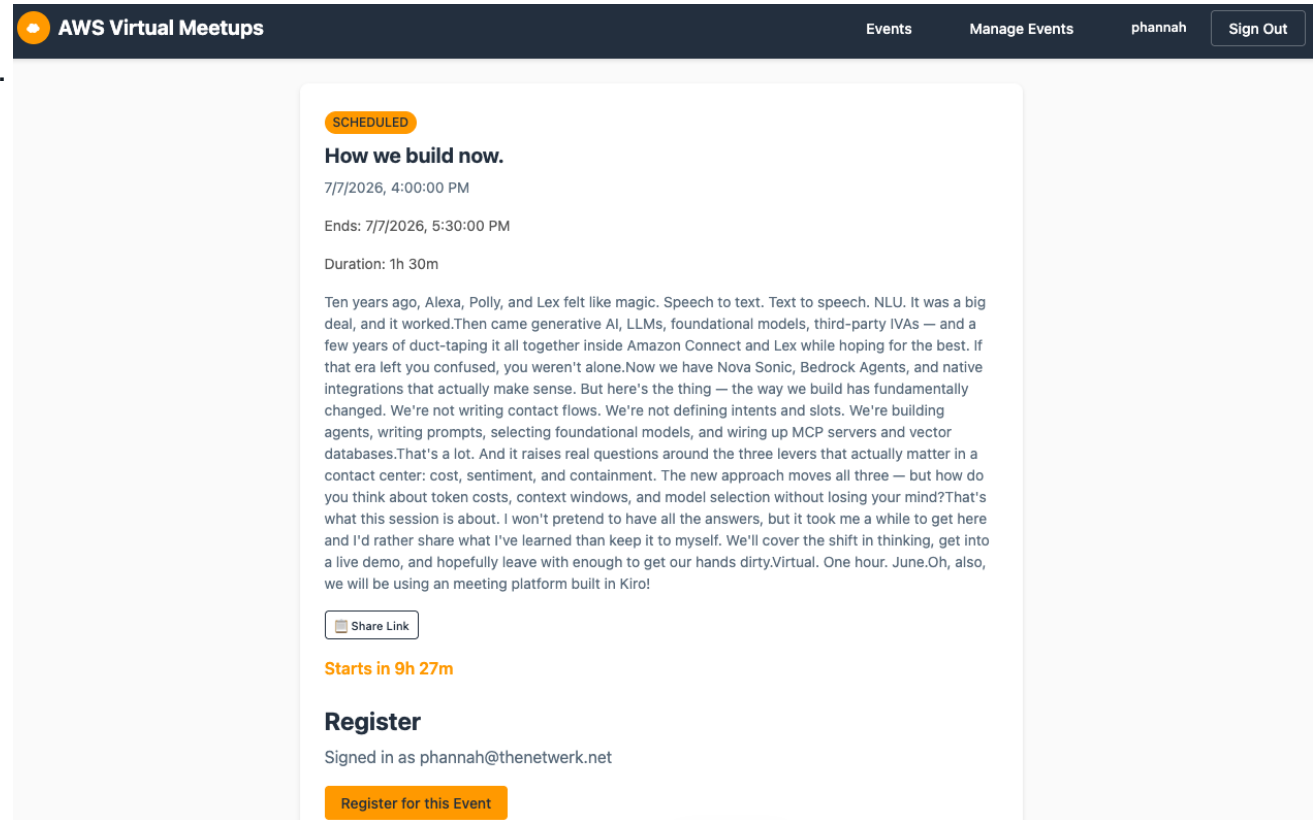


AWS User Groups

- We are part of the AWS User Groups Community: <https://builder.aws.com/community/user-groups>
- Connect focused and virtual; originally incepted by someone at AWS, I took it over some time ago when they stepped down. Last one (July) was my first - There was a mix of folks who pre-registered and joined on demand); Mostly were registered (as compared to anonymous).
- Without a doubt, in the 9+ years since Amazon Connect was launched, interest and investment has grown significantly, so I thought it was time to do something with the group.
- I don't want this to be about me – I want this to provide real value to the community (that's you), and I want the community to engage – virtual is hard (in-person is hard as well, some overlapping, some different reasons).
- Engage means how we stay in touch between sessions, how we produce and share material, and the way that material is presented.
- I have a sibling user group in suburban Atlanta (more AWS general in nature) I'm working to get going, which means that we can make this user group in person or hybrid from time to time as well (if you have specific region in mind, let's talk!)

Side Quest: This platform and Kiro

- If you are hearing me, and seeing my screen, it was a success. Otherwise, it's a great Linked In post about failing in public.
- The platform was built with Kiro. If you've not heard of it, it's an IDE with AI assisted development built in, with a prescriptive leaning towards spec driven development.
- Even though we are here to talk Connect (and the ecosystem around it), you might find value in Kiro in your own work — so give it a try.
- This platform is available on GitHub as well — fork, contribute, etc.: <https://github.com/ronin48llc/aws-virtual-meetups>
- What went wrong last time? My screen share wasn't recorded!



The screenshot shows the AWS Virtual Meetups website. The header includes the AWS Virtual Meetups logo, navigation links for Events, Manage Events, and a user profile for phannah with a Sign Out button. The main content area features a scheduled event titled "How we build now." with the following details:

- SCHEDULED**
- How we build now.**
- 7/7/2026, 4:00:00 PM**
- Ends: 7/7/2026, 5:30:00 PM**
- Duration: 1h 30m**

The event description reads: "Ten years ago, Alexa, Polly, and Lex felt like magic. Speech to text. Text to speech. NLU. It was a big deal, and it worked. Then came generative AI, LLMs, foundational models, third-party IVAs — and a few years of duct-taping it all together inside Amazon Connect and Lex while hoping for the best. If that era left you confused, you weren't alone. Now we have Nova Sonic, Bedrock Agents, and native integrations that actually make sense. But here's the thing — the way we build has fundamentally changed. We're not writing contact flows. We're not defining intents and slots. We're building agents, writing prompts, selecting foundational models, and wiring up MCP servers and vector databases. That's a lot. And it raises real questions around the three levers that actually matter in a contact center: cost, sentiment, and containment. The new approach moves all three — but how do you think about token costs, context windows, and model selection without losing your mind? That's what this session is about. I won't pretend to have all the answers, but it took me a while to get here and I'd rather share what I've learned than keep it to myself. We'll cover the shift in thinking, get into a live demo, and hopefully leave with enough to get our hands dirty. Virtual. One hour. June. Oh, also, we will be using a meeting platform built in Kiro!"

Below the description is a "Share Link" button. At the bottom, it says "Starts in 9h 27m" and "Register". The user is signed in as phannah@thenetwork.net, and there is a "Register for this Event" button.

Me

- I really liked phones. Somehow, I lucked myself into a job that let me get paid to play with them, which is a lot cheaper than building a phone network on my own.
- That was 25 years ago.
- Avaya came into the picture; Oracle Oracled; I got bored and ended up in the cloud.
- Arguably the phone network was a form of the cloud, but with the cloud, the phone network got a lot more capable.
- Now we are seeing that with AI/ML/GenAI/AgentAI.
- I'm currently in Atlanta (suburban), but spent most of my life in South Florida – I did a lot of work with AWS user groups and the tech community in general, and I'm hoping to do the same for this group as well.



Getting to go-live is just the starting point

- New capabilities land in Amazon Connect monthly. Generative and agentic AI move faster than that, and they have driven service deprecations at a pace we have not seen before.
- Agentic CX Designer is on the horizon (let's hope it lives up to the hype - Connect AI Agents are already a beast).
- The instance you went live on eighteen months ago does not look like it does today. It should not look the same at the end of this year either.
- If it does look the same, that is the problem, not the rate of change.
- Your mission, your business, your customers' expectations: none of it stands still. So why would your contact center?
- Lead with capabilities. Do not hinder with legacy approaches.

Print the docs and you get 11,948 pages

- Four guides: Administrator Guide, API Reference, Connect Developer Guide, Agent Workspace Developer Guide. The Administrator Guide alone is 5,758 pages.
- It is reference material, so it reads like reference material. Different from Lego, different from Ikea. (At least, unlike Oracle, you get screenshots on occasion.)
- Buried in there: US toll-free numbers are potentially more resilient than local numbers. If you have ever asked why I am hesitant to offer an alternative to ACGR (Amazon Connect Global Resiliency) that is part of why.
- It took me five minutes to find that page. Good thing I have an okay memory. For now.
- Material gets added and removed. EVT? Instance-to-instance contact transfers? Yes, there are release notes and a document history - they lag the platform.

Three things I think are true

- Connect, AWS, AI/ML, your mission, technology, the world: all of it keeps moving.
- One person's memory cannot hold, find and fix architectural deficiencies at that pace. Mine included, and I have been doing this 25 years.
- I have not yet run into a focused approach for managing these challenges with Connect specifically.

So what did I do?

- I AI'ed my way through those 11,948 pages and turned what came back into best practices for Connect, with an eye on how the platform evolves. (Remember when the URL format changed?)
- That became a custom Amazon Connect lens: 6 pillars, 118 questions, 602 best practices.
- Then automation around discovery: 281 checks that answer roughly 30% of the lens by brute-forcing APIs. No AI in the path yet so expect it to only improve coverage-wise.
- And a skill on top, so I can ask "is it a good idea to use the default version in a Lex bot?" and get an answer with a citation instead of a gut check.
- The whole thing rebuilds. Documentation changes, the lens changes. Post-incident and post-implementation reviews feed back the same way.

Well - Architected, in three pieces

- Most people have heard of the Framework. Fewer have used the Tool. Almost nobody has written a lens.
- The Framework: AWS's design-review method. Six pillars of questions with documented trade-offs. Not a product, not a certification. A structured argument about a workload.
- The Tool: free in the console. Define a workload, answer the questions, and it maintains a risk register, an improvement plan and milestones, so a re-review shows movement instead of opinion.
- A Lens: that question set scoped to a workload type. AWS ships them for Serverless, SaaS, Financial Services, Generative AI and more. A custom lens is JSON you upload: same Tool, same reporting, your questions.
- Write valid JSON and the Tool does the scoring, the reporting and the history for you. That part is genuinely nice. The collection, not so much, and anything that cannot be asserted through technology is subject to opinion.

Why Connect needs its own lens

- The generic pillars ask good questions about the wrong things. Four properties of this workload break the default assumptions.
- Operational as much as technical: you are judged on handle time, containment, occupancy and adherence right alongside IAM and encryption. No AWS lens has vocabulary for the operational half.
- Serverless by construction: nothing to right-size, nothing to patch. Failure moves to quotas, API throttling and integration timeouts, which is exactly where the generic pillars point you at the wrong evidence.
- A platform, not a component: Connect is not a service inside the workload, it is the workload. Flows, queues, routing profiles, security profiles, hours and recording settings are the architecture, and none of them show up in a CloudFormation drift report.
- Extensibility means integration, not code: you do not fork Connect. Lambda, Lex, Cases, Customer Profiles, contact records to Kinesis, the Streams SDK, third-party CTI. Every one of those boundaries carries its own quota, retry semantics and failure mode.

Six pillars, and what each one holds

- Operational Excellence: flow logging, metrics, testing, change management. 20 questions, 110 practices.
- Security: identity, encryption, recording controls, PII/PCI/HIPAA handling, security profiles. 20 questions, 103 practices.
- Reliability: quotas, Global Resiliency, error branches, integration failure paths, capacity headroom. 20 questions, 114 practices.
- Performance Efficiency: routing design, queue and profile structure, latency, network, agent experience. 20 questions, 102 practices.
- Cost Optimization: telephony and usage charges, storage tiering, licensing, retention, feature enablement. 20 questions, 98 practices.
- Sustainability: data footprint, retention discipline, storage lifecycle, media you did not need to capture. 18 questions, 75 practices.
- The schema caps a pillar at 20. Five of six sit at the ceiling. Sustainability at 18 is where the Connect surface honestly runs out.

Legacy state is a finding, not a footnote

- Instance access domains: legacy is alias.awsapps.com/connect/, current is alias.my.connect.aws.
- The old format blocks Global Resiliency outright, and there is no published deprecation date, which is exactly why it belongs in a lens instead of a ticket.
- The practice is not "have you switched." It is "do you have an inventory and a plan," because the change touches seven things owned by different teams:
 - SAML relay state (needs new_domain=true appended), firewall allowlists, S3 CORS AllowedOrigins, browser AutoplayAllowlist policy, approved origins for embedded CCP and CTI adapters, agent bookmarks, and every piece of training material.
- Voice ID reached end of support on May 20th, 2026. It is in the lens as a decommissioning item, not as "did you configure it." Grading a customer on a dead feature is worse than not asking.
- When AWS does announce a date, an unprepared customer does all of the above under a deadline.

The half that is not in the PDFs

- Everything on this slide and the next came out of delivery, not documentation. Each one got doc-checked, then became a practice.
- Lambda retry budget: Connect retries on error and throttle up to three times inside the total timeout budget, not per attempt. Three retries on an eight-second function does not buy you twenty-four seconds.
- Lex draft aliases in production: \$LATEST via TestBotAlias carries far lower concurrency, and updating the draft terminates in-progress conversations. A draft alias in a live flow is an incident waiting for volume.
- Contact records vs contact events: records are delivered at least once and deduplicate on ContactId plus LastUpdateTimestamp. Events are best-effort and unordered. Events can never be your system of record.

Three more that cost somebody a weekend

- Deletion does not propagate: DeleteContactData does not reach streamed archives. If records went to Kinesis and onward, you propagate the deletion downstream explicitly or the erasure request is not satisfied.
- Default flows are templates: clone them, never point production at a default flow. Check for it, because it is invisible until the day the behavior changes underneath you.
- EVT is off the meter: external voice transfer legs do not count toward concurrent active calls, so the standard 80%-of-quota alarm is blind to that path. The real ceiling is SBC session capacity and SIP trunk sizing, outside AWS quotas entirely.
- This is the portion of a review cannot automate and cannot read out of a guide. It is also the half a customer can tell the difference on inside ten minutes.

Why bother with a formal review?

- It converts opinion into a documented position. Every finding cites a page, so the conversation stops being about whose architect is more senior.
- Everyone gets the same vocabulary: customer, partner and AWS account team reading one risk register, in a tool all three already have access to.
- It is repeatable, so improvement is measurable. Same questions, same scoring, next quarter, with milestones showing movement instead of asserting it.
- It sizes the real work before anyone signs for it. A review is the cheapest way to find out that the migration scope on the table is wrong.
- And if you are a partner: reviews sit inside the AWS partner programs and the funding attached to them, which changes who pays for phase zero.

The readout is the product

- Hand over six hundred red rows and you have produced a document nobody will fund work against. I have written that document. It did not help anyone.
- A findings dump: everything weighted equally so nothing is prioritized, no source so every item is arguable, findings asserted from evidence nobody collected. It ends in a meeting.
- This readout: severity comes from the database, and nothing gets promoted or demoted in the room. Every finding carries its citation, the command that proves it, and the fix.
- Met only when the evidence shows it. Not met only when the evidence shows its absence.
- And a "Not assessed" section that names what would settle each open item. That is a feature, not an apology, it stops a review from claiming coverage it did not earn (and it is usually the section that sells the next phase).

What is deliberately not in here

- Amazon Connect configuration shipped. 118 questions, 602 practices, running in the Well-Architected Tool today.
- The surrounding AWS stack, everything the contact center integrates with. Out of scope on purpose. That is a different lens that we started but has a lot more work to put into it.
- Contact center operations, workforce management, forecasting, scheduling, adherence, quality, coaching. The non-technical half, and a third lens. That's coming 😊
- The gaps are the roadmap. External voice transfer went from a customer conversation, to a backlog item, to a lens practice.
- The lens is only worth what the next engagement puts back into it.

Demo

- Well Architected

How We Build Now



Conclusion + Homework

- Five minutes to a working contact center? Still true. Getting to well-architected takes a *little* bit longer.
- Needs should drive the tech. Sometimes it runs the other way (be honest with yourself about which one you are doing). Either way, close the loop as you have more data about your contact center than you have ever had.
- Homework: take one item off the two field-knowledge slides, go check it on your own instance this week, and come back and tell the group what you found. Check out the Lenses, Check out the Tool, and keep skilling yourself, the platform is not going to wait for you.
- How are you keeping up? I do not have a good answer either — let's work on it together, and tell the group what you want from the next meetup.

Thank you!

9/15/26



AWS User Groups