Risk Management for the Software Supply Chain\

Why the Future will be Federated

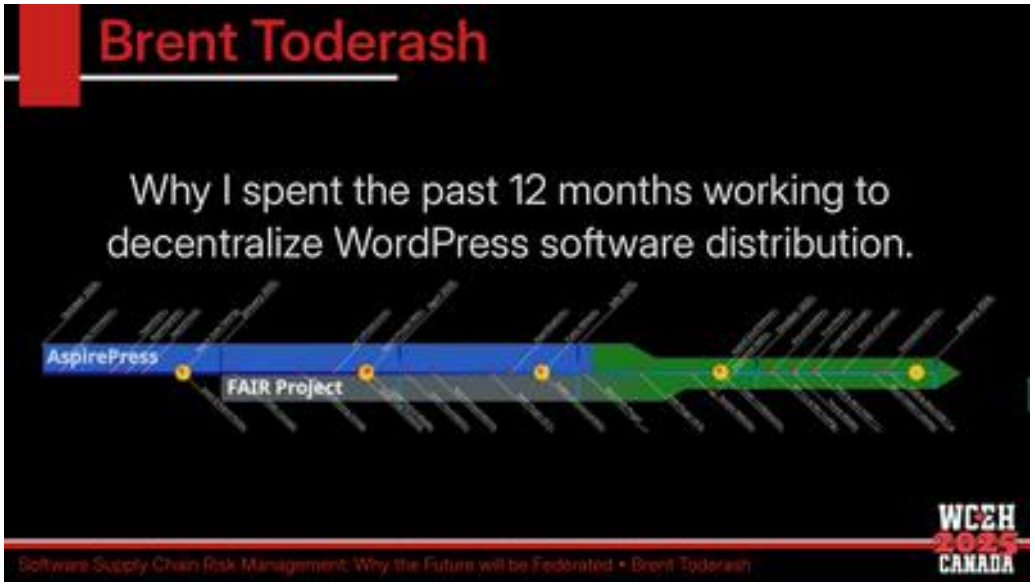
(Title Slide)

Brent Toderash

Who? b2 user. *Did I Miss Anything?*

- My path to WordPress was through open source, rather than the other way around. This is what brought me to b2, and from there of course to WordPress.
- I've been a blogger, freelancer, entrepreneur, ISP owner, web host, agency owner, web developer, project manager, open source advocate, writer, and wearer of many other hats. I'm also an iconoclast, a tester of assumptions, and freelance thinker.
- I've been using a Linux desktop for over 2 decades, and ran a Linux advocacy site and proto-blog 25 years ago around 1999 and the early 2000s.
- I've had a number of influences from that period, including *Just for Fun*, *Hackers*, *The Cluetrain Manifesto*, *The Starfish and the Spider*, *Small Pieces Loosely Joined*, *The Cathedral and the Bazaar*, *Rebel Code*, as well as others like *Geeks*, *Here Comes Everybody*, and *Free as in Freedom*.
- I went to start a new blog one day an found that b2 had three forks available. I tried them all and started my first WordPress blog. I wrote under a pseudonym, just for myself, and ended up with 2,000 daily readers and a top-ten Technorati rank in its niche. I returned to long-form writing last year, but am having trouble finding time to write all that I want or feel the need to say.
- For the past year, I've been involved first with AspirePress and then the FAIR project.
- Of course, the day job(s) continue, as we own and operate a boutique hosting service focused on managed WordPress since 2012, as well as a WordPress-focused agency. Prior to that, we took our ISP from dialup to wireless broadband while doing software and web development projects. "We" in this context is myself and my brother Scott, we've been partners on these web ventures for over 25 years, so the accomplishments are shared.
- All that to sum up by saying I'm here from the old school. These days, I don't always edit the code anymore, but when I do... Vim!

Slide 3



Brent Toderash

Why I spent the past 12 months working to decentralize WordPress software distribution.

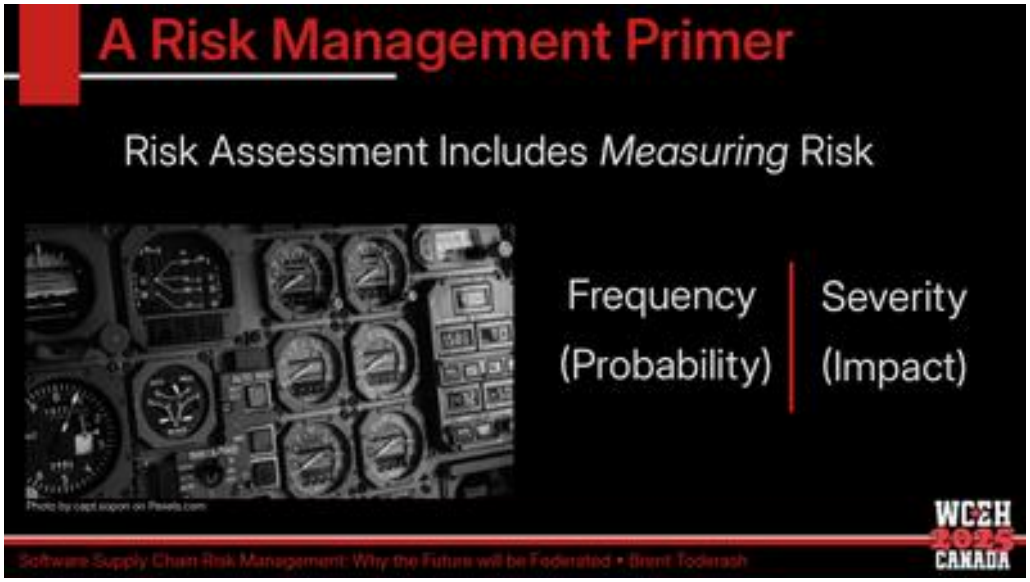
- By the end of this talk, it will make sense why we've put a year into this and are pressing hard on it. You'll come to understand why we believe it's crucial for the future of WordPress, its community, and its ecosystem.

Slide 4

Risk Management Primer

- Some people find this surprising, but pre-Y2K, I held a professional designation in the general insurance industry. I was good at it, and as I changed careers into tech and managing projects, I found that risk management principles are transferable.
- The first step in managing risk is obviously to identify it. But on its own, that's not enough.

Slide 5



A Risk Management Primer

Risk Assessment Includes Measuring Risk

- There are two primary metrics in measuring risk:
  - Frequency, which is the statistical term for how often something occurs, but we can think of it as probability. How likely is it to happen?
  - Severity, which is essentially impact. How bad would this be?
- There are risk mitigation strategies for both, and they are not always employed as an either/or strategy, but often as a combination of strategies.

Slide 6

"What's the Worst that Could Happen?"

- This is a good question for risk managers, but the real trick is what to do about it.
- Managing risk starts with mitigating, both the probability and the impact of the "Bad Things".

Risk Mitigation

We have two approaches to risk management.

1. Make Someone Else Responsible

This is called "Transfer of Risk"

- Insurance and other contracts are common mechanisms for this, but it's not always a viable approach when it might be too little, too late.

2. Reduce Risk to Acceptable Level

- An acceptable level is whatever risk – or loss – you can take in stride. To get there, you can take mitigation steps to adjust both of our risk measures.

Reduce Probability

- This is typically loss prevention, like adding firewall – most security tactics live here.

Reduce Impact

- Having a backup or a recovery plan is one example of reducing impact.

The difference between these two is this: reducing probability would be strong security measures so hackers can't get credit card numbers from your network. Reducing impact is not storing the credit card numbers.

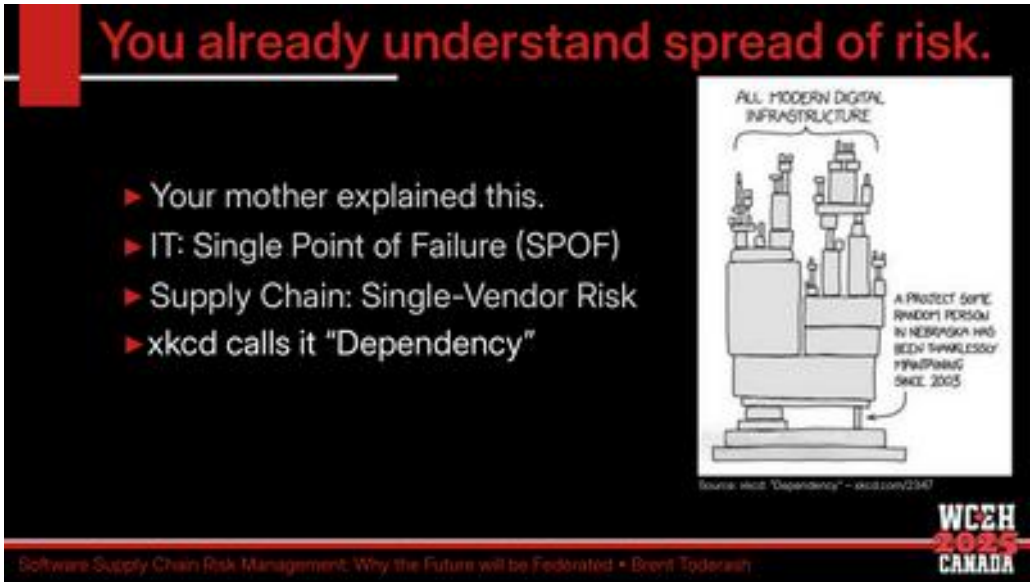
- Spread of risk is another important means of reducing risk, and it's an underappreciated tactic.

Slide 8

You already understand spread of risk.

- Your mother explained it to you when she told you not to put all your eggs in one basket.
- IT or DevOps calls this a single point of failure (SPOF).
- The supply chain calls it single-vendor risk.

Slide 9



- xkcd calls it “Dependency”. (See, I told you you already understood this.)

Slide 10

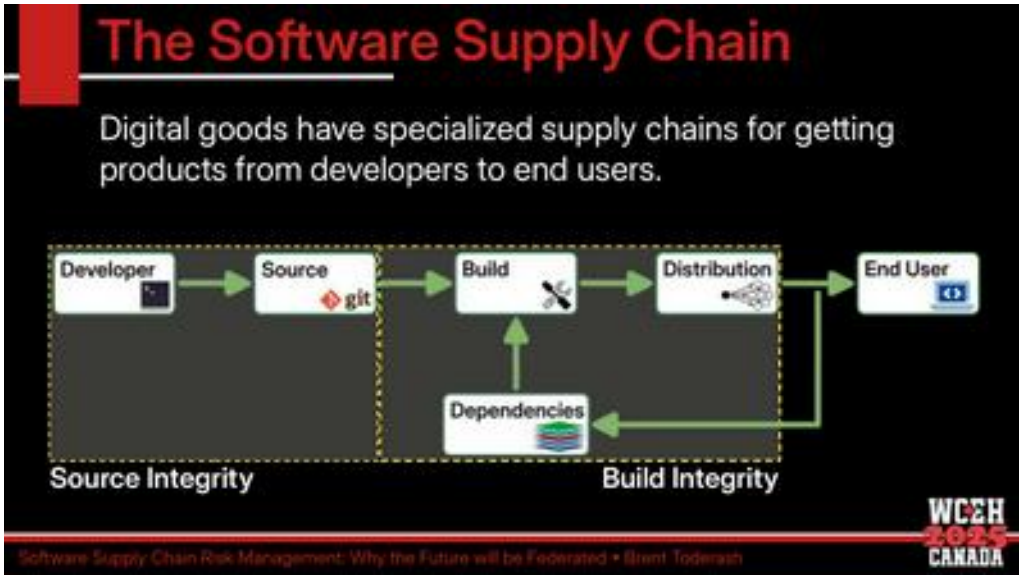
Managing Supply Chain Risk

If you're not mitigating risk, you're not managing it – you're accepting it.

Slide 11

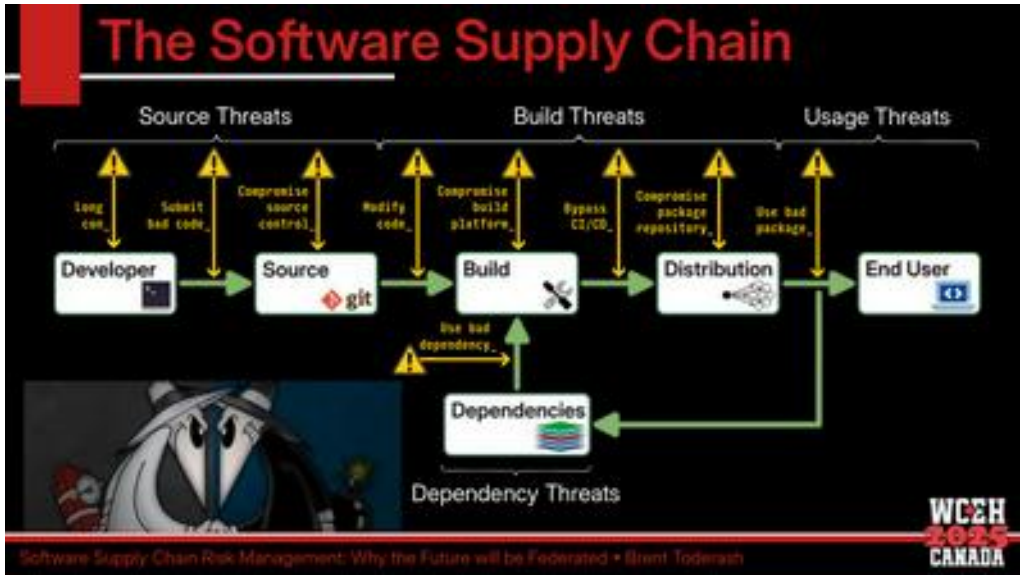
Software Supply Chains

- Software may not come in a box anymore, but it still has a supply chain – even without being delivered on a truck. The supply chain isn't just a COVID-era scapegoat for lack of product availability.
- Software has a supply chain, and many or most of you are part of it.



The Software Supply Chain

- Digital goods have specialized supply chain, typically illustrated with a diagram like this one.
- We want to look at two areas within this chain:
  - Source integrity
  - Build integrity



The Software Supply Chain

- We can group supply chain threats in four areas:
  - source, build, usage, & dependency threats
- There are many attack vectors for bad actors: we can point out nine of them here.
- We can show examples of the types of attack that might be made in each of these vectors. Obviously some are easier to mitigate than others.
  - The “long con” doesn’t usually show up in these example diagrams, but we could think of the XZ Hack where the attacker spent time to gain a position of trust. This is difficult to address, but is still preventable.
- *What’s Missing* from all of these diagrams is *single-vendor risk*. The supply chain security diagrams generally focus on technical solutions, but we need to address non-technical risks as well.
- Since it’s not primarily a technical problem, single-vendor risk doesn’t typically employ a technical solution, but this is not always the case. For example, a host’s network will be multi-homed using BGP so they are not reliant on a single upstream gateway provider or peer. The astute host will ensure that the physical fibre path for those peers do not share a conduits into the building or on other major fibre routes.

Slide 14

WordPress Supply Chain Risk

- A CDN with multiple datacenters may address some single points of failure, but single-vendor risk is a SPOF that is unresolved by that technical approach.
- A year ago now, people in the WordPress community began looking at wordpress.org as a single point of failure, exposing the risk of having a single-vendor for distribution of our WordPress plugins and themes.
- Over 40% internet relies on *this* supply chain, which is a massive single-vendor risk to be addressed.
- It's important to be clear on this: it doesn't matter who runs it, the fact is it shouldn't rely on a single entity.
- Supply chains hate uncertainty, which is to say that enterprise companies who employ risk managers will flag this as a problem. Left unchecked, this will not just impair, but eventually halt the growth of WordPress in the enterprise. Should that happen, the SME market will take notice and begin to follow.

Slide 15

It's Not Just WordPress.

- We're not just picking on WordPress here; we've now seen a parallel in the Ruby community, where Ruby Central "asserted control" over Ruby Gems. This was ostensibly done to reduce risk, which I find ironic at best.
- In fairly short order, the community responded, and you can already bypass Ruby Gems to pick a different horse. This is a mirror solution, so will be interesting to watch where it goes. The lesson may be that centralized control is seen as a kind of censorship that the internet will treat as damage, and route around it. The internet generally doesn't respond well to a "you must" kind of directive that it doesn't believe in.

Slide 16

Other Supply Chain Threats

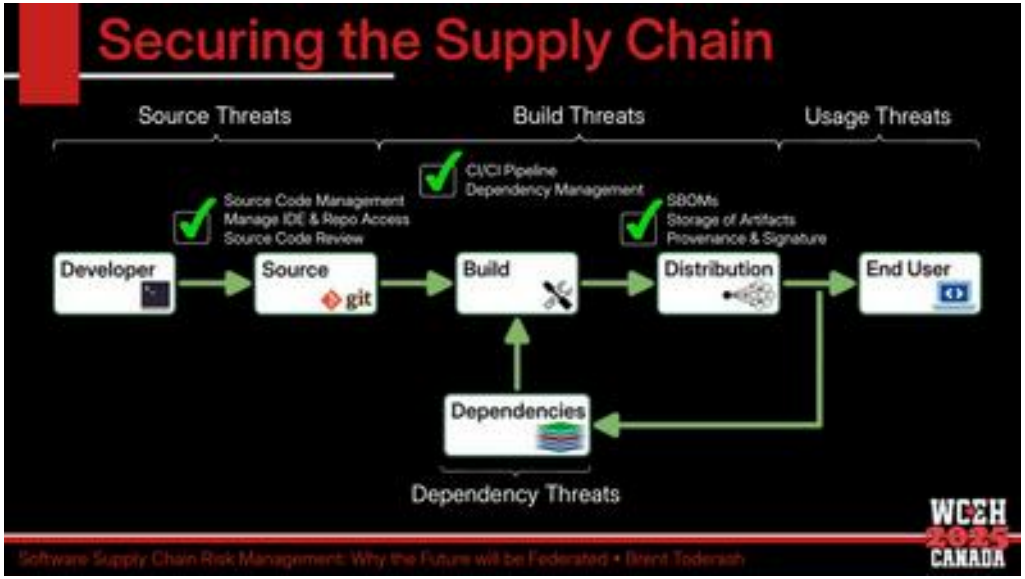
- When you have centralize repositories, you have centralized threats: when all the gold is in Fort Knox, it draws all the attacks.
- I'm thinking here about the recent npm supply chain attacks. In that case, GitHub stepped in even though not the primary attack vector, because a GitHub action was used as part of the attack.
- As an aside, GitHub is becoming too large a point of failure, but they are more actively managing much of the risk. They aren't really a single-source vendor for hosting code, but there's another kind of risk inherent in having too large a market share – but that's a different talk.

Slide 17

Warning Signs

- Last month, we saw an open letter with almost a dozen signatories telling us that sustainability is a problem for open source repositories.
- Don't mistake for the tragedy of the commons, which some are already calling this. What we're actually seeing here is a *centralization* problem, not a *commons* problem. The Tragedy of the Commons is still the biggest FUD-based myth in all of open source, applying a thoroughly discredited idea to the wrong thing in the wrong way – but that's a different soapbox.
- You don't need an economist to tell you that when you centralize the repository, you centralize the cost, and this group is telling you that centralized costs are a problem.
- The supply chain does need to be sustainable, so we're looking for some reassurance on that front as well.

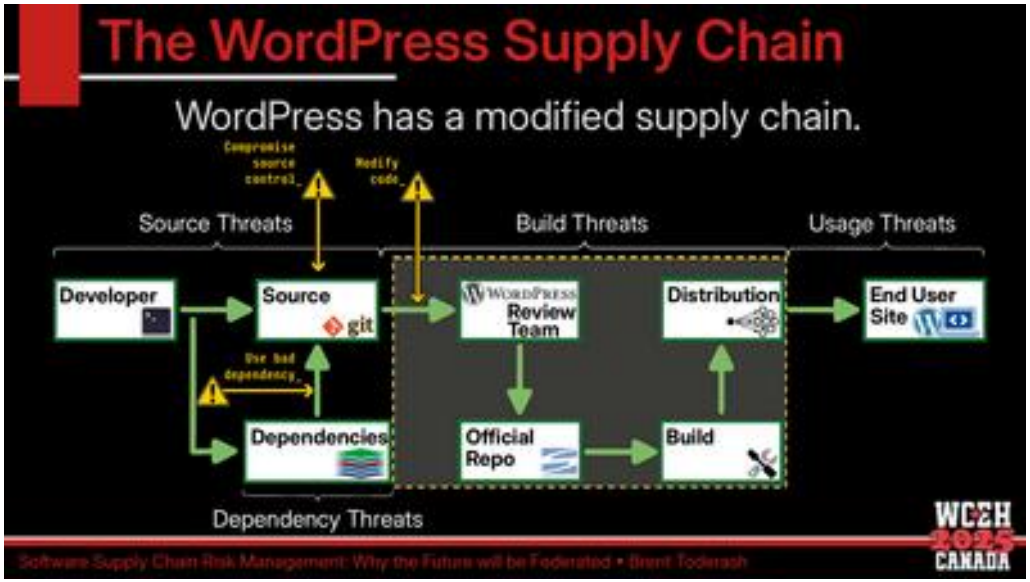
Slide 18



Securing the Supply Chain

- Back to our supply chain, with our four groupings of risk, where we're going to look mainly at Source Threats and Build Threats. We can list some specific tactics or vectors we need to secure for risk mitigation in each area.

Slide 19

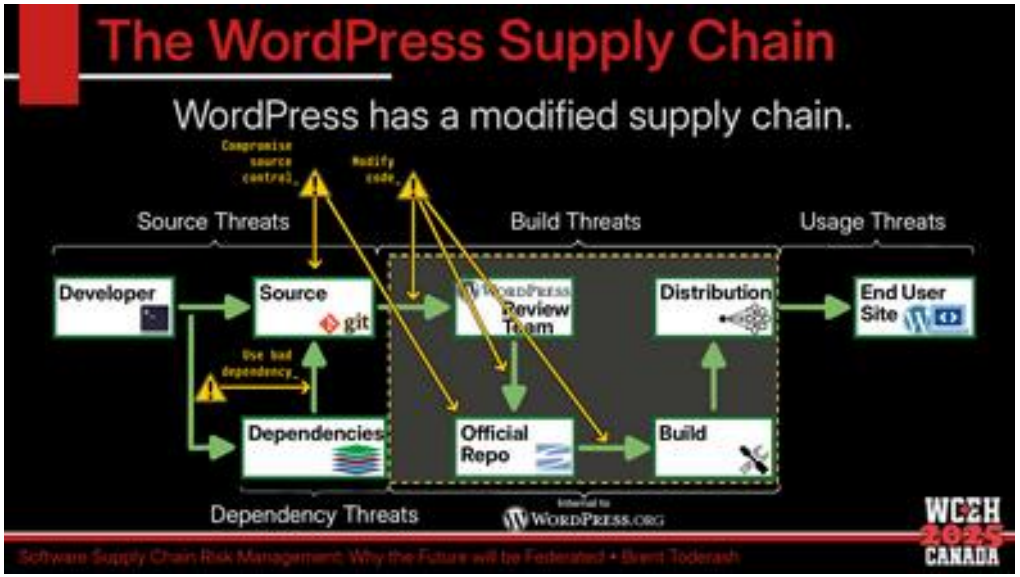


The WordPress Supply Chain

We've been looking at diagrams of a typical software supply chain, but WordPress has a modified supply chain.

- Here we find two extra steps in the build stage:
  - The Review Team that vets incoming plugins & themes, and
  - the official Subversion repository, which is in addition to anything the publisher uses for source code management – usually Git.
- The same attack vectors are present...

Slide 20



- ...but those vectors have additional attack surfaces, additional places where they can be attempted which if successful can still affect the entire supply chain.
- These are are areas which either don't have reliable safeguards, or they're opaque because they are internal to wordpress.org.
- We also notice where dependencies are added in this model – not at the build stage, but earlier, in the development stage. Without a software bill of materials (SBOM), who's checking those, and how? Because of this approach, the dependencies are not updateable separately from the package with which they're integrated. In the case of a security update for a dependency, without the SBOM, you may not know there's a vulnerability until the publisher decides to update the bundled package.

Slide 21

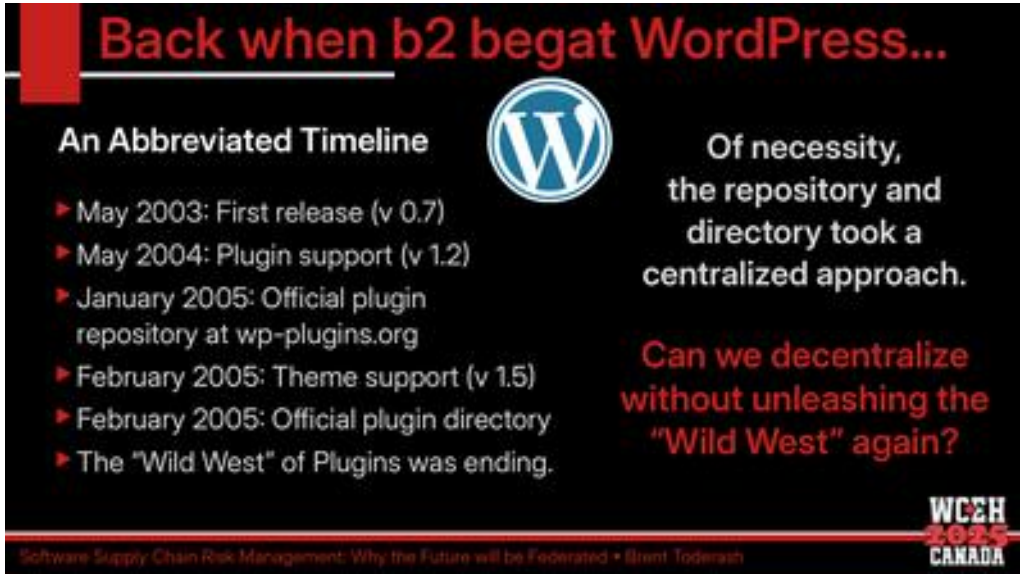
Where Did We Get this Model?

This is more of a "when" question.

Circa 2004...

- Software distribution was done through download sites like Tucows & download.com.
- You'd usually download as well as upload by ftp, broadcasting your password in plain text because the times were so innocent back then.
- Subversion was the dominant source code management platform, and remained so until 2010.
- Major software projects were still using Waterfall methodologies over Agile, which had yet to gain much traction.
- It's significant that these last two points reflect software development practices of the cathedral, not the bazaar.
- Back then, broadband just surpassing dialup, and blogging was in the zeitgeist.

This sets our stage.



Back when b2 begat WordPress...

- May 2003: First release (v 0.7)
- May 2004: Plugin support (v 1.2)
- January 2005: Official plugin repository at wp-plugins.org
- February 2005: Theme support (v 1.5)
- February 2005: Official plugin directory
- The "Wild West" of Plugins was ending.

In those days, with a much smaller community, it was harder to find plugins and themes, and when you did, it was hard to know whether you could trust them. The dot-org directory addressed that as people gradually moved their software into the repo. Until then, you had to just figure it out – so I'd find a theme by Brian Gardiner, and I'd say "Hey, I know this guy, I like his work, and I'd trust the theme." (That's how I learned to write WordPress themes, by dissecting his work.)

So of necessity, the repository and directory took a centralized approach.

Question: Can we decentralize without returning to the wild west?

- If we're going to fiddle with the system, we should improve it, and (spoiler) we can.

Slide 23

Verdict on Our Supply Chain?

- Doing nothing is always an option, just not often the best one.
- Maybe it's fine...
  - ...maybe it's not.

Slide 24

It's time to shift.

- Because doing nothing accepts too much risk.

Response Began October 2024

- AspirePress – began with a vision for distributed & federated repositories for WordPress. Founded by Sarah Savage, it grew as a suite of software projects toward this goal.
- When I found the project, what I found was a group of people asking the right questions, which is more important than thinking you have the right answers. It was also important that it was a group. A year ago, there were a lot of people who were ready to tell you how to create your own mirror of the WordPress repository, but that's simply not practical for most people to just spin up a mirror of 60,000 plugins. At the time, it was mostly individuals sharing code for doing it, which is a Bus Factor of 1.
- This is the second important part about AspirePress – the plan was to stand up infrastructure and provide a publicly-accessible mirror, so nobody had to do it on their own. We did this with Fastly, who came on as a partner early on.
- At AspirePress, I guess I opened my mouth a few too many times and found myself project managing large portions of the project. We had some great developers contributing, so with about six months of development & a private beta period, we launched version 1.0 this past June 6, 2025 at AltCtrl.org in Basel, Switzerland with a demo showing how you could update your WordPress site from AspireCloud, which indexed our mirror of the plugins and themes.

A FAIR Response

- FAIR – Federated And Independent Repositories – was formed from talks that began in early 2025, and formed under the Linux Foundation using their governance model. The Linux Foundation oversees about 1,000 projects, so they have a proven track record in this area, and we could lean on their experience.
- Since our goals were in close alignment and the communities had significant overlap, several of us in AspirePress joined in these early discussions and affirmed a unity of purpose.
- FAIR's formation announcement was coordinated with AspirePress at AltCtrl.Org in Basel, and a press release from the Linux Foundation. The announcement included the release of the FAIR plugin for Technical Independence, which accessed AspireCloud to update plugins and themes, and added other functions which replaced connections to WordPress.org, increasing privacy and performance.
- FAIR was announced on stage at the Linux Foundation's OpenSource Summit a few weeks later in Denver. Both announcements brought an overwhelmingly positive public response.
- Interestingly, both the AspirePress and FAIR projects were erroneously called forks in their early days, but the goal for both is to serve the existing WordPress community and ecosystem.

Slide 27

A FAIR & Aspirational Response

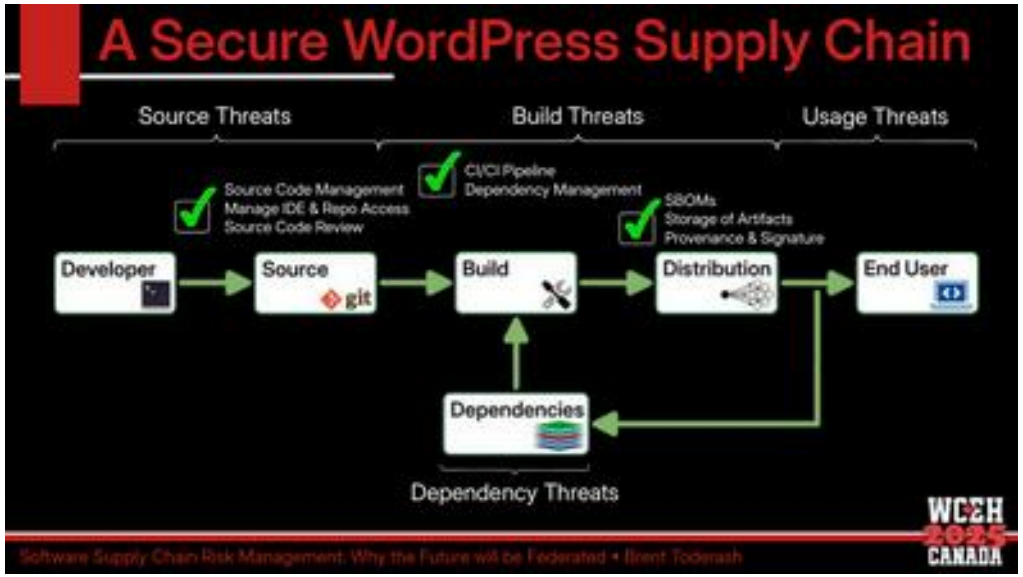
- The primary goal for both is to decentralize and federate software distribution for WordPress.
- The approach is generalized so it could apply to other digital goods, but our implementation has WordPress-specific extensions to serve the main goal.
- FAIR and AspirePress together provide disruptive potential for how we distribute software through a secure supply chain, and AspirePress is now moving much of its projects and effort under FAIR, sharing resources between them.

Slide 28

What's a FAIR Response?

- FAIR represents a protocol and an architecture for achieving secure, independent, and federated repositories. It leverage existing standards and protocols from sources like the W3C and Bluesky for decentralized identities (DIDs), AT Protocol, and others. Bluesky's PLC server is used for provisioning and resolving DIDs.
- The structure is a general protocol with WordPress-specific extensions, and we've seen some interest in what we're doing from outside the WordPress ecosystem. FAIR is creating an architecture that implements the protocol and enables others to do the same through a suite of software tools we are releasing under the GPL and MIT licenses.

Slide 29

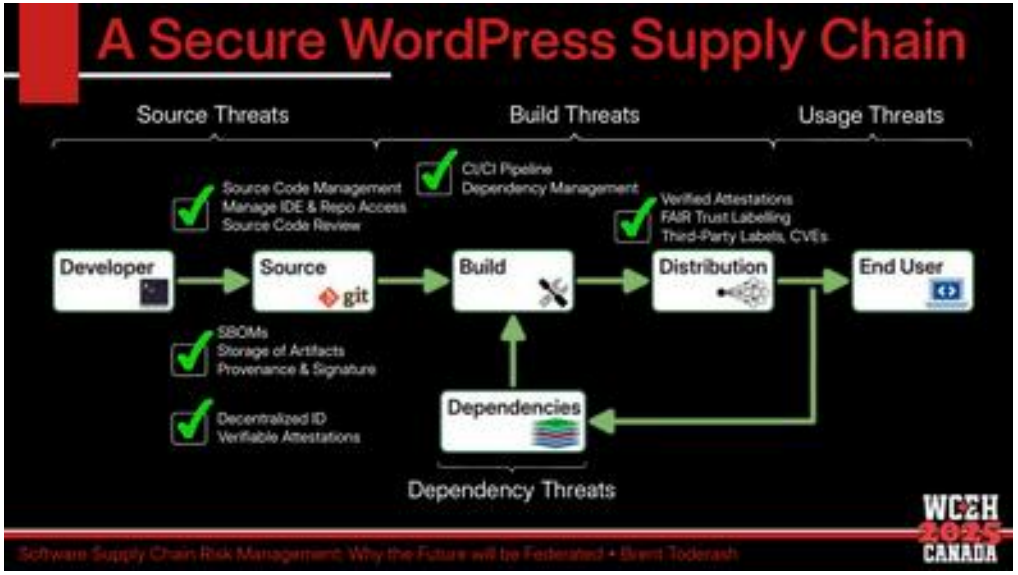


A Secure WordPress Supply Chain

Back to our supply chain.

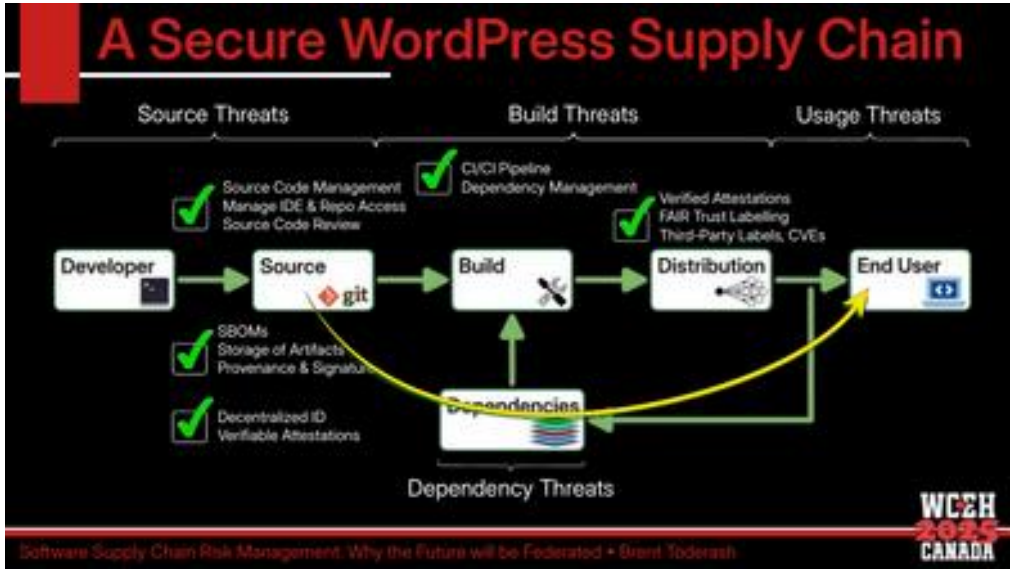
- We want to bring our supply chain closer to the typical diagram structure while making it work securely as a distributed supply chain.

Slide 30



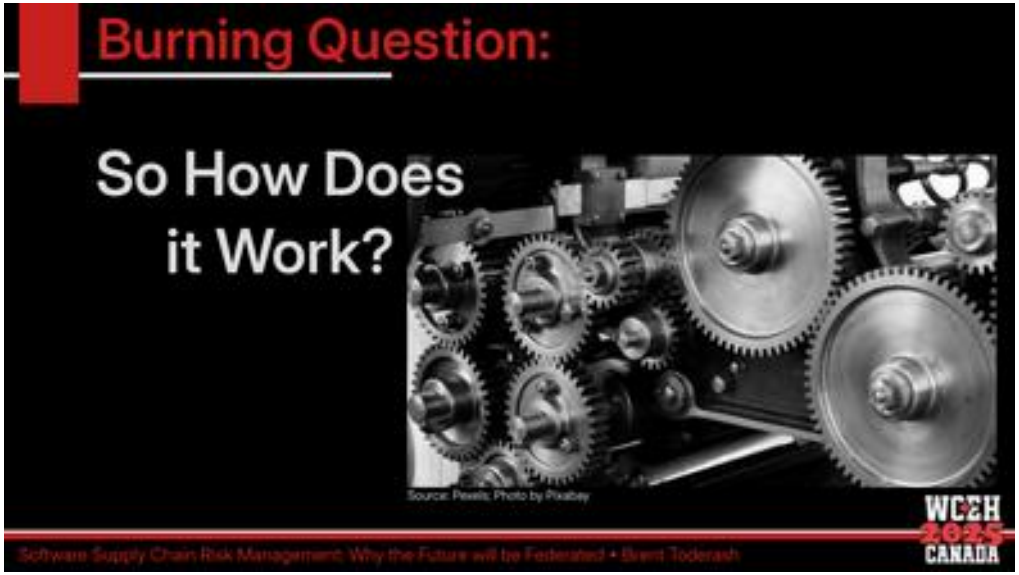
- To do this, we had to move some things, and add some things.

Slide 31



- Then we tightened the connection between the canonical source and the end user, and this is important. In this model, downloads come directly from their federated independent canonical source. They are cryptographically signed to ensure a bit-for-bit copy, even if the package has been cached somewhere along the chain.

Slide 32

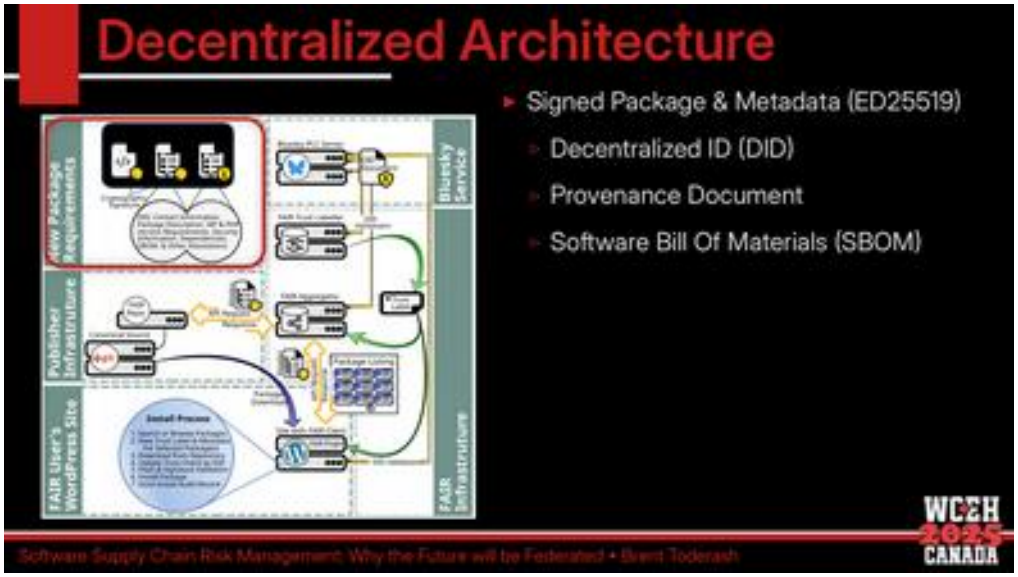


Burning Question:

How does it work?

(Interstitial Slide)

Slide 33

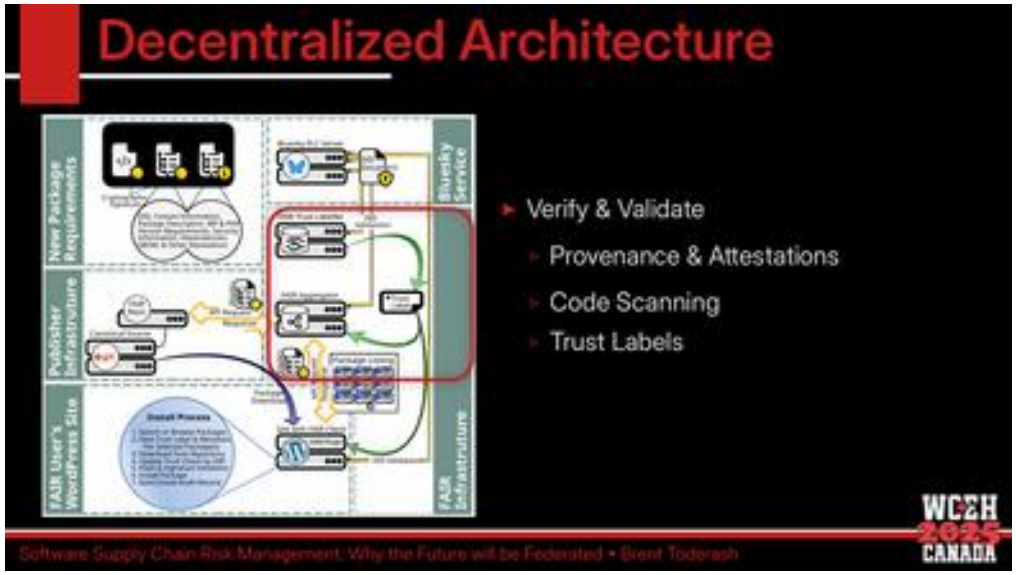


FAIR Architecture

What's in a Package?

- We're extending the metadata provided with the package, and adding a cryptographic signature to the metadata as well as the software package (ED25519), so the metadata is also verifiably consistent with what the publisher provides. Each package will have a Decentralized ID (DID) so we always know we're talking about the same thing. It will also include a provenance document with verifiable attestations from the publisher and a software bill of materials (SBOM).

Slide 34

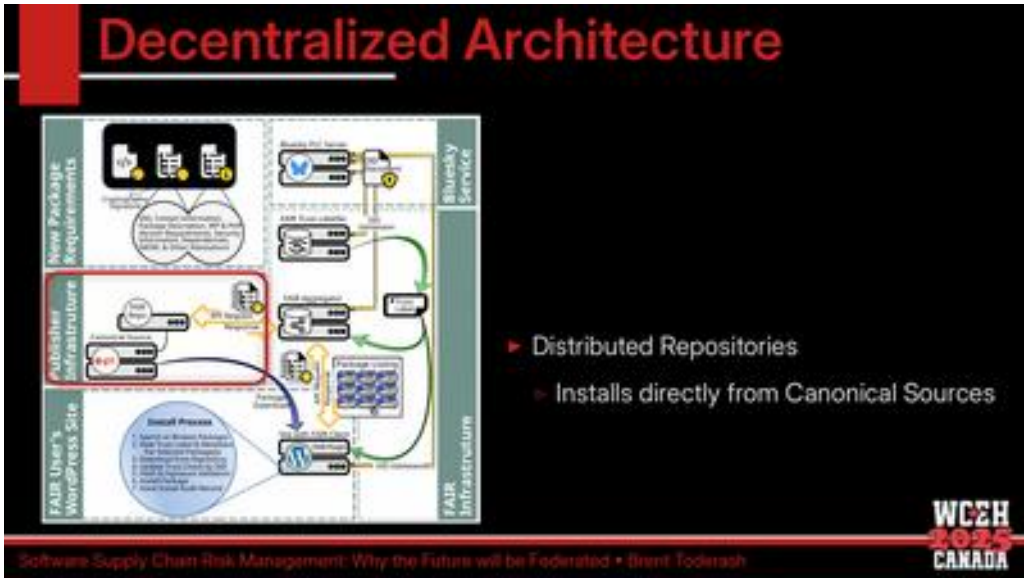


FAIR Architecture

FAIR Infrastructure

- FAIR will verify and validate the package's provenance and attestations using tools like code scanning and third-party checks to apply a Trust Label.

Slide 35

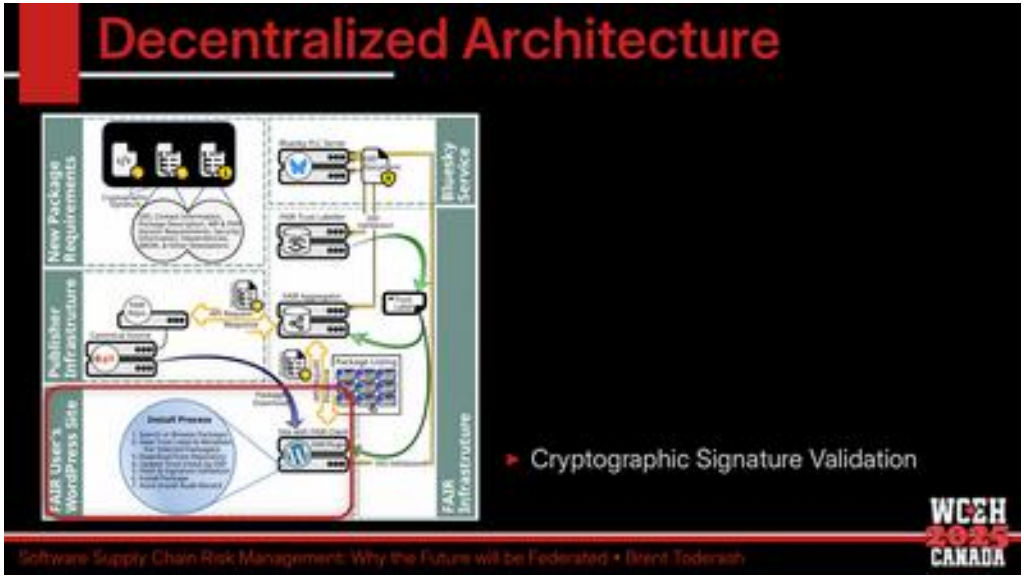


FAIR Architecture

Publisher Infrastructure

- With distributed repositories, the publisher decides where the package is hosted. Installation of the package is done directly from its canonical source.

Slide 36

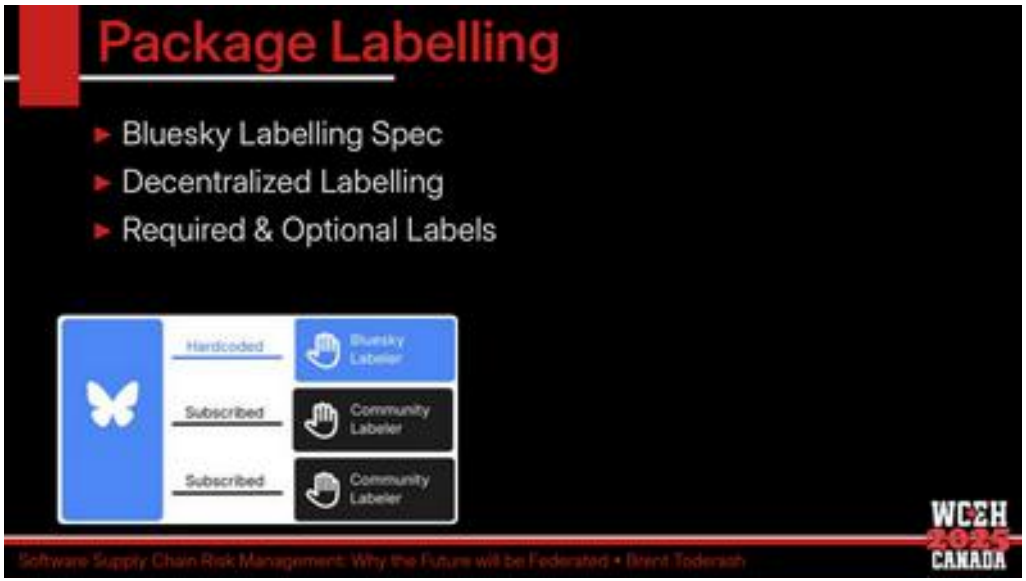


FAIR Architecture

End User Site

- The FAIR Plugin validates the cryptographic signature for the package before installing it. No matter what trust signals have been given or validated to this point, if the signature check fails, the install is aborted.

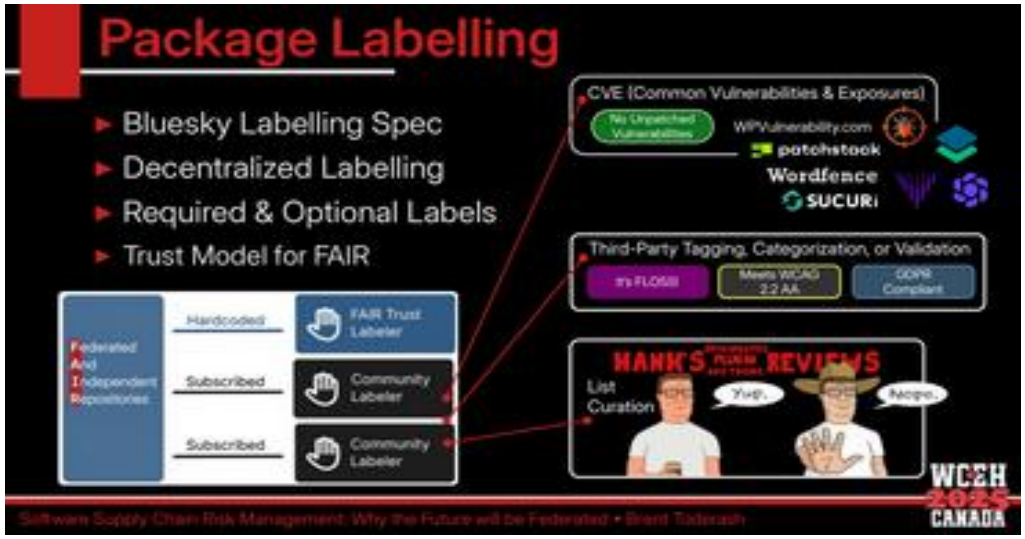
Slide 37



Package Labelling

- We had a good look at Bluesky's decentralized labelling model, which includes both hardcoded required labels and optional ones to which anyone can subscribe. This is present now on Bluesky, where they add their version of a "blue checkmark" to confirmed journalists and other public figures. Beyond that, other people can operate a labeller to vouch for other accounts or recommend people to follow.

Slide 38



- In creating a Trust Model for FAIR, we wanted to use the same approach for implementing it, so we have a required labeller for FAIR-assigned Trust Labels. We also have support for community labellers which can be subscribed to for applying a variety of labels from trusted sources. For example:
  - CVEs could be provided by any number of vendors, whether under a free API or a paid subscription. In November this year (2025), we'll be doing a hackathon sponsored by Patchstack leading into CloudFest in Miami, and we're going to build this with an API provided by Patchstack, so you'll be able to see whether your plugins or themes have any listed CVEs.
  - Other uses may be simply to categorize, tag, validate or score the package, perhaps to confirm it uses an open source license and is GPL-compatible, or that it meets the WCAG 2.2 AA standard, or that it is GDPR compliant.
  - Someone might create a curated list to offer reviews, ratings, and recommendations, and if you trust the source, perhaps you configure your site not to install anything that gets a "Nope" from Hank.

Slide 39

Digital Trust is Kind of a Big Deal.

A Trust Model?

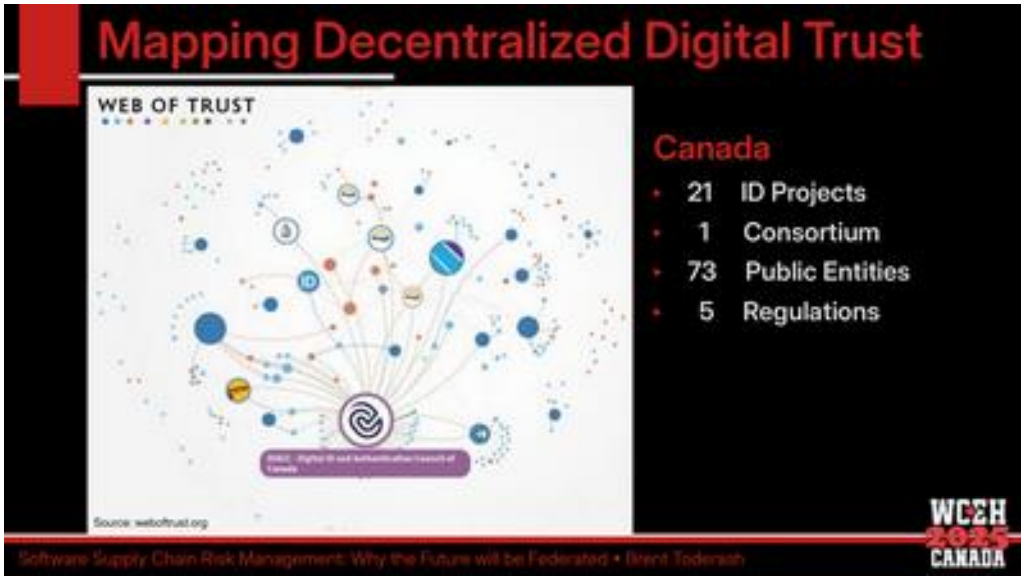
- Digital trust is getting a lot of focus right now, and this is part of where the web is headed. The Web of Trust project has catalogued thousands of public and private projects and initiatives in this space. Some of these projects will be familiar as ways of verifying identity while maintaining regulatory privacy requirements.
  - We may think people will resist this, but you may have had to upload your passport to get on an airplane; I had to take a photo of my driver's license to check into my hotel.
  - These kinds of digital trust can be used to verify things like age, citizenship, where you got your Ph.D., or perhaps at some point, that you're the authorized publisher of a software package.

Slide 40

Mapping Decentralized Digital Trust

- Globally, the Web of Trust Map includes 267 ID Projects, 43 Consortia, 1,205 Public Entities, 42 Regulations, 83 Standards or Protocols, and 134 DID Methods.

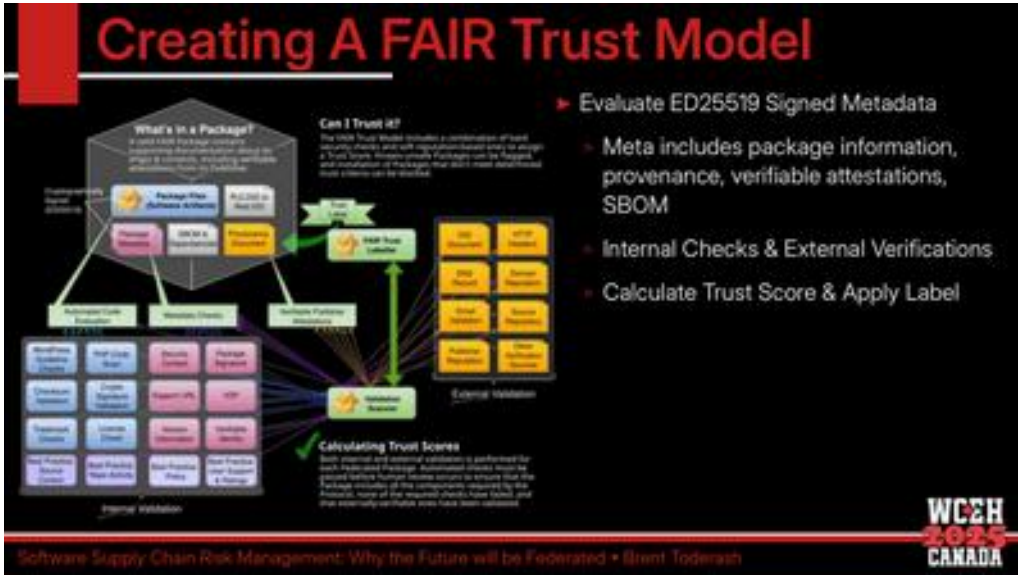
Slide 41



Mapping Decentralized Digital Trust

- Here in Canada alone, it's 21 ID Projects, 1 Consortium, 73 Public Entities, and 5 Regulations in addition to private or commercial entities.

Slide 42

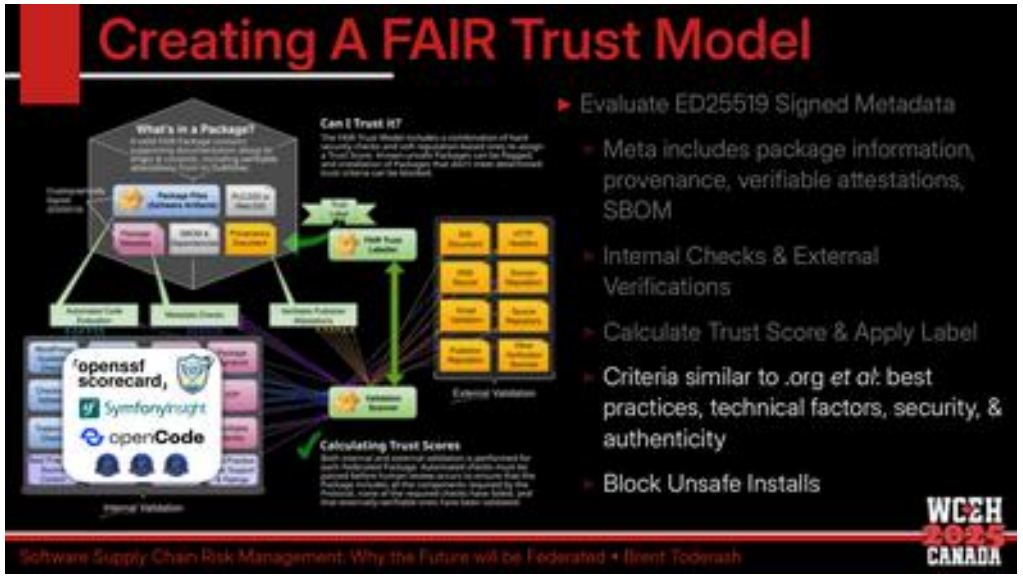


Creating A FAIR Trust Model

Our Trust Model evaluates a number of "Trust Signals", some required, others optional.

- The package itself contains some of these in its signed metadata, extended to include not only the typical package information, but provenance documentation, verifiable attestations from the publisher, and an SBOM.
  - FAIR can use internal & external validation & verification methods with this data to build a trust profile for the package so it can calculate a Trust Score and apply the appropriate label. These signals include things like verifying a domain alias for the package's DID, if one is provided. Other factors might be considered, such as domain reputation and confirming from HTTP headers that your repository has a valid TLS certificate in place and that the email addresses provided are deliverable.
  - Once the automated checks have been passed and verified, human review can occur and the label can be applied based on the package's Trust Score.

Slide 43



Creating A FAIR Trust Model

- In addition to external validation, static scans or runtime checks can be done with the package to evaluate other factors, similar to the criteria applied by WordPress review teams and others, such as SLSA, the OpenDirectory Badge program, OpenSSF Best Practice Scorecard, and the EU's Cyber Resilience Act (CRA) requirements list. Generally, the criteria relate to security, authenticity, the use of best practices, and other technical factors.
- Some of these checks are used to block unsafe package installs, while others are used to report how trustworthy a package is believed to be in order to let the user decide whether to install it.

Slide 44

"The Times, They Have A-Changed."

- We structure software projects differently: we've abandoned Waterfall for Agile methodologies.
- We organize development teams differently: we've gone from centralized to decentralized with distributed teams (from Cathedral to Bazaar).
- We do version control differently: CVS and Subversion have fallen out of favour, replaced by Git as the dominant SCM platform for the past decade.
- We even use software differently: we don't even install software like Microsoft Office anymore.
- We should distribute software differently.

We can democratize the publishing of software.

Why the Future Will Be Federated

1. Distributed is the natural, healthy state of the internet.

Why the Future Will Be Federated

In a perfect world, a central dispatcher (a piece of software) would know where every bit needs to go and would route each the most efficient way. But that router's every hiccough would have the effect on the rest of the system of a cardiac arrest. So the Internet was designed to have many decentralized routers, each making decisions about where to send packets next. If one of the routers goes offline… the packets are simply sent to another router. The internet routes around disruption.

The difference is between on the one hand, having your automobile club lay out a map that shows you a direct route from New York to San Francisco and, on the other hand, navigating by asking gas station attendants along the way who give replies such as, "Gosh, I don't know how to get you to San Francisco, but I think you'll be closer if you drive northwest to the next Sunoco station and ask again.

When it comes to packets in a highly dynamic highway system, the stop-and-ask technique turns out to be not only more robust, but more efficient. This only surprises us because we have long assumed that centralized power and efficiency go hand in hand.

– David Weinberger, *Small Pieces Loosely Joined: A Unified Theory of the Web* (2002)

The internet has always relied on decentralization to establish resilience.

The title of Weinberger's book is a beautiful image of this: *Small Pieces Loosely Joined*.

Slide 47

Why the Future Will Be Federated

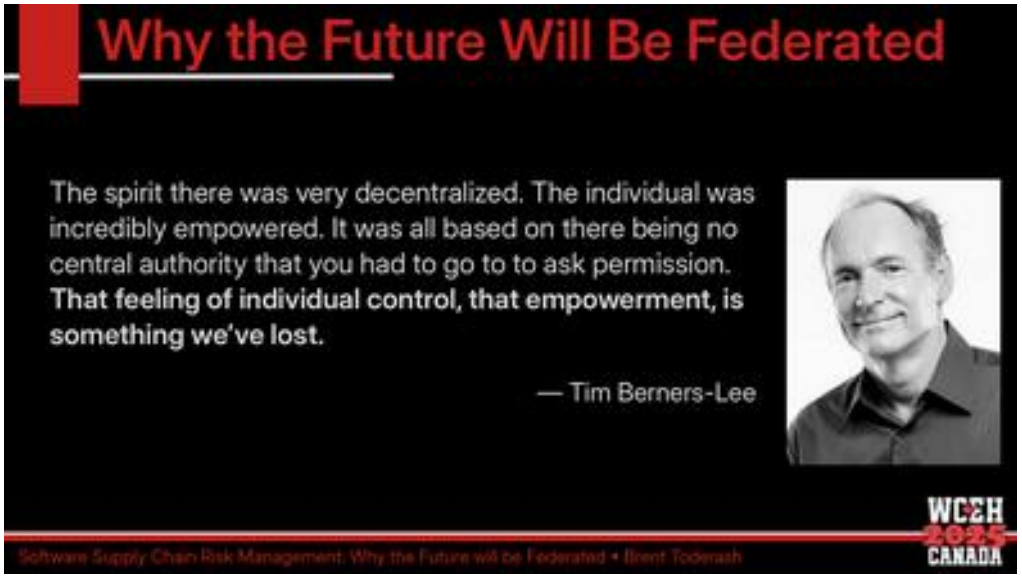
The spirit there was very decentralized. The individual was incredibly empowered. It was all based on there being no central authority that you had to go to to ask permission. That feeling of individual control, that empowerment, is something we've lost.

— Tim Berners-Lee

Notice what Tim Berners-Lee is saying, that the internet has lost something that needs to be recovered.

In other words...

Slide 48



Why the Future Will Be Federated

Web 3.0 is just the web rediscovering its roots.

Why the Future Will Be Federated

2. The web has always been a starfish, not a spider.

Why the Future Will Be Federated

With a spider, what you see is pretty much what you get. A body's a body, a head's a head, and a leg's a leg. But starfish are very different. The starfish doesn't have a head. Its central body isn't even in charge. In fact, the major organs are replicated throughout each and every arm. If you cut the starfish in half, you'll be in for a surprise: the animal won't die, and pretty soon you'll have two starfish to deal with.

Starfish have an incredible quality to them: If you cut an arm off, most of these animals grow a new arm. And with some varieties... can replicate itself from just a single piece of an arm. ...They can achieve this magical regeneration because in reality, a starfish is a neural network—basically a network of cells. Instead of having a head, like a spider, the starfish functions as a decentralized network. Get this: for the starfish to move, one of the arms must convince the other arms that it's a good idea to do so. The arm starts moving, and then—in a process that no one fully understands—the other arms cooperate and move as well. The brain doesn't "yea" or "nay" the decision. In truth, there isn't even a brain to declare a "yea" or "nay." The starfish doesn't have a brain. There is no central command. Biologists are still scratching their heads over how this creature operates.

– Ori Brafman & Rod Beckstrom, *The Starfish and the Spider: The Unstoppable Power of Leaderless Organizations* (2006)

TL;DR – A spider is a WYSIWIG creature; head, body, legs. Cut off the head and it dies. Starfish on the other hand are decentralized organisms. Cut one on half or cut off an arm, and you'll end up with two starfish.

Slide 51

Why the Future Will Be Federated

The harder you fight a decentralized opponent, the stronger it gets.

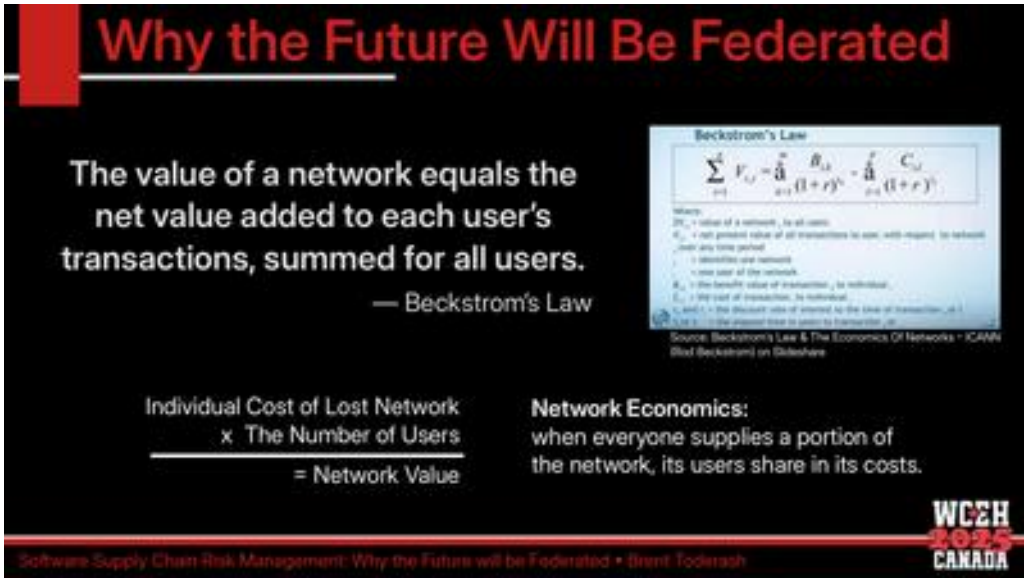
- This is like the Hydra of Greek mythology, where if you cut off a head, two more grow in its place.
- If you doubt this, ask the record industry about how they ended file sharing by destroying Napster.

Slide 52

Why the Future Will Be Federated

3. Centralization brings zero network value

Slide 53

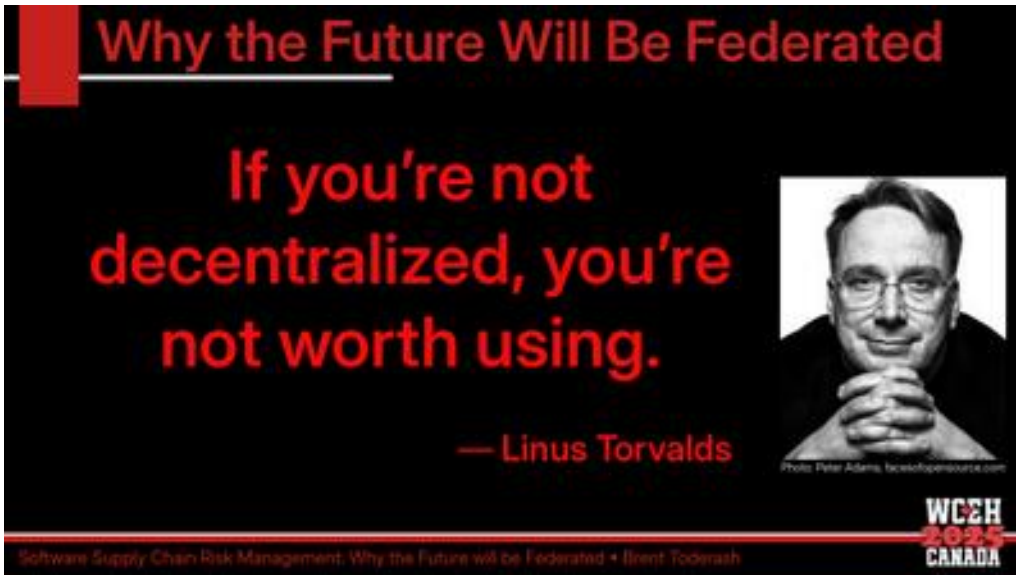


Why the Future Will Be Federated

> The value of a network equals the net value added to each user's transactions, summed for all users.
>
> — Beckstrom's Law

- This has some complicated math or some simple math, but at the end of the day the network economics are this: when everyone supplies a portion of the network, everyone shares in the cost, which improves its sustainability.

Slide 54



Why the Future Will Be Federated

> **If you're not decentralized, you're not worth using.**
>
> – Linus Torvalds

- Linus was talking about Git over CVS or Subversion, which he refused to use because centralized source code management approaches did not support the decentralized development of the Linux kernel.
- Aside: he built Git in 10 days during April 2005; 10 years later it was dominant SCM platform worldwide.

Slide 55

Why the Future Will Be Federated

4. It's time. The future starts now.

Slide 56

Why the Future Will Be Federated

Innovation tends to happen when the time is right.
— John Pierce (Bell Labs, 1937 through 1960s)

- He was referring to satellites here, but the observation applies equally to transistors and to the telephone itself. In each of these cases, there were many people pursuing the same thing at the time when the breakthrough innovation occurred.
- We're seeing this now in the thousands of projects and entities around the globe working to establish secure, reliable forms of decentralized digital trust. This is the missing piece that previously drove us toward centralization, but it's now been resolved on multiple fronts, with more coming.

Slide 57

Why the Future Will Be Federated

The ethos of the Open Web is a Decentralized Web.

Slide 58

Federated. Independent. Future.

The Future will not be Centralized.

End Notes

Q&A (Time Permitting)