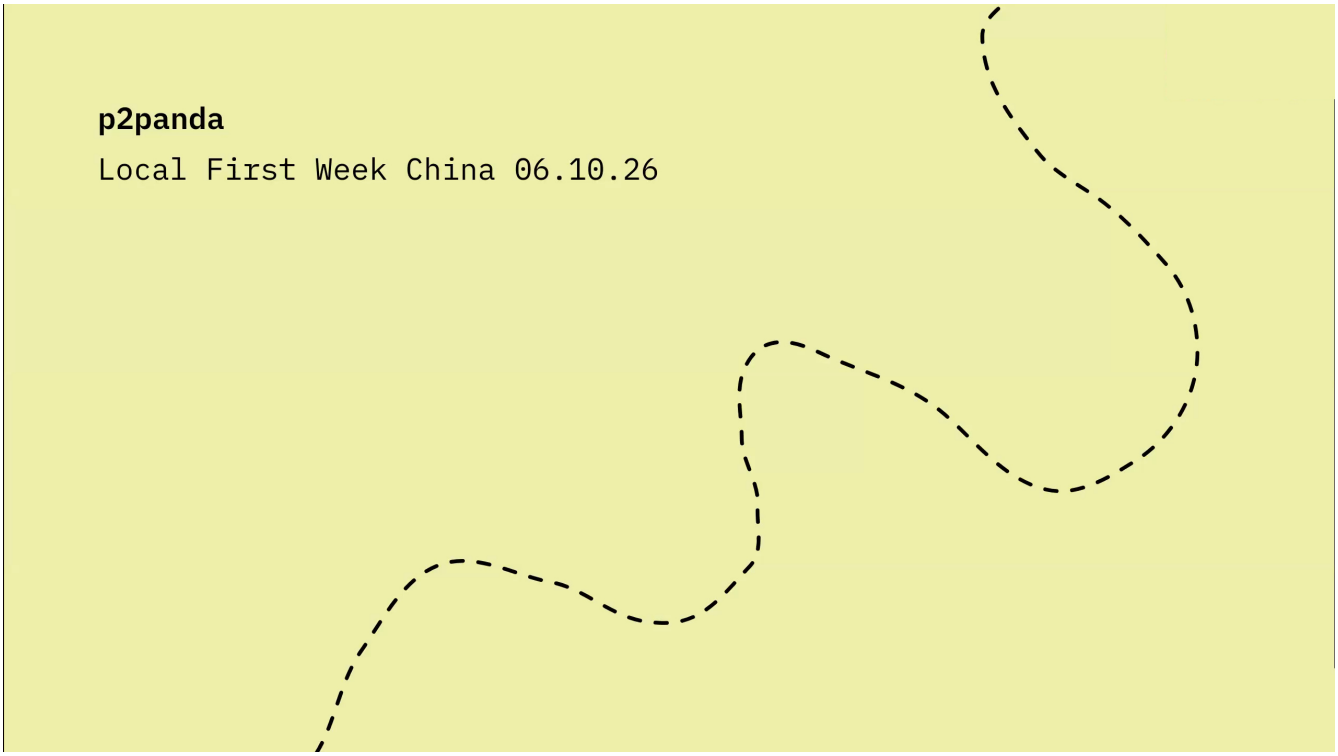
p2panda：本地优先 P2P 系统的构建模块

Sam Andreae · 2026-10-06 · 中文译文

中文译文。根据 2026 年 10 月 6 日的演讲与问答整理，保留发言顺序、例子、限定条件和交流承接，校订口头重复、断句与明显识别错误。产品状态和后续计划以分享时点为准。时间对应交付视频。

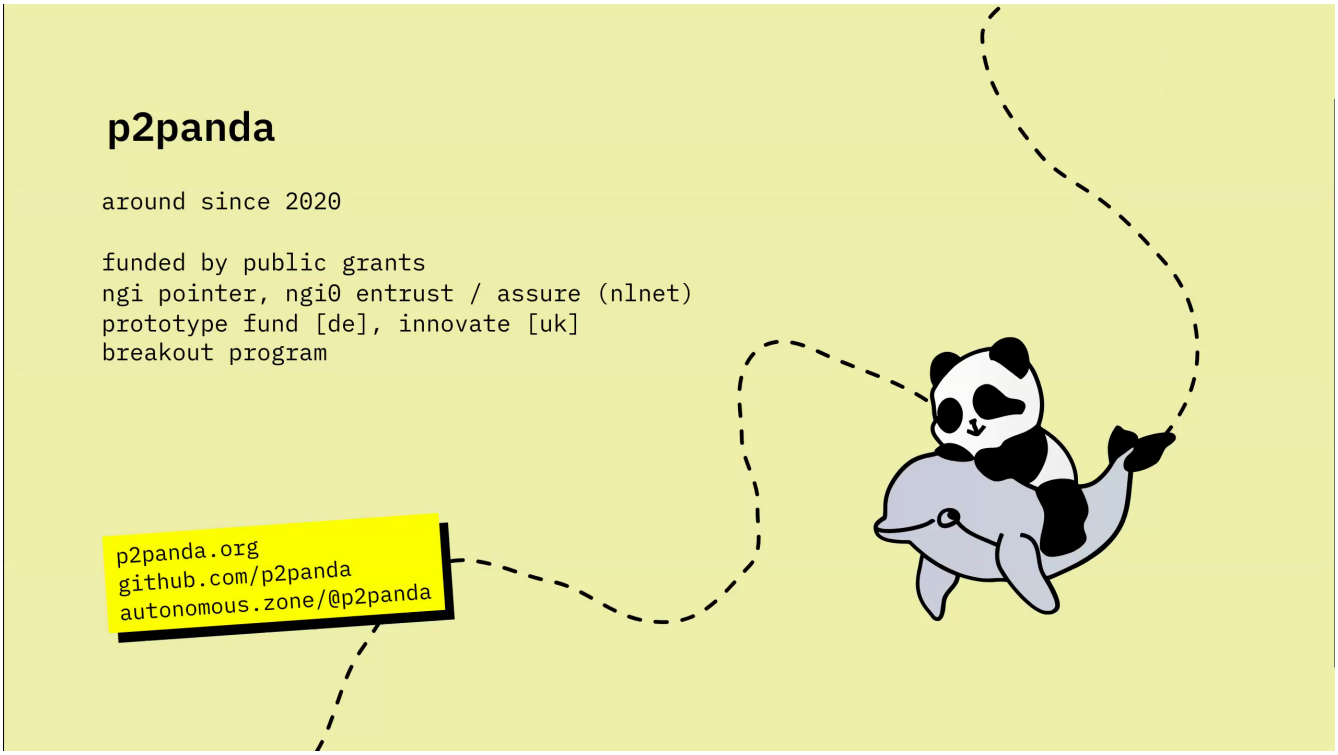
开场与 p2panda 项目介绍



p2panda · Local First Week China · 画面 00:00:16

00:00:00 · Sam Andreae

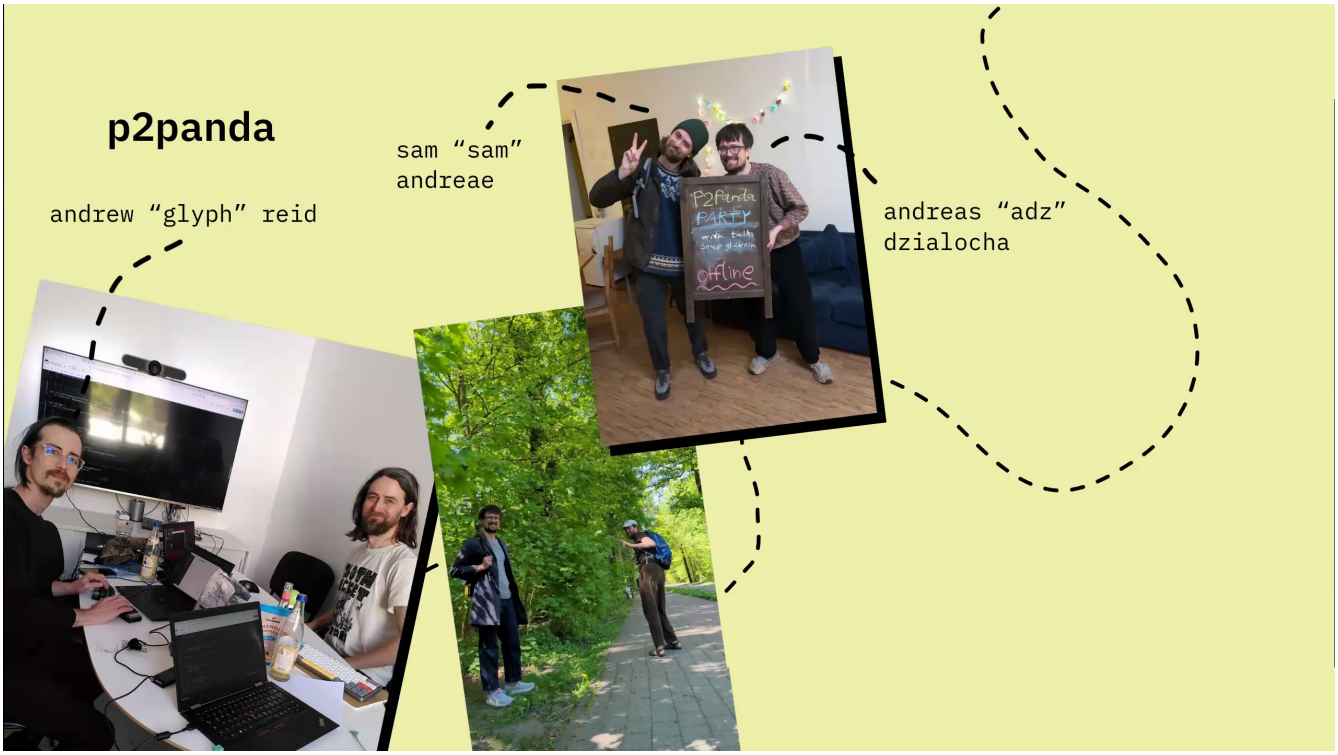
大家好，欢迎。我是 p2panda 的 Sam，今天来给大家介绍这个项目。非常感谢组织者邀请我们。看起来你们已经办了很多很棒的分享，不少讲者都是我们熟悉的人，我想后面应该还有几场。很高兴能来到这里。请看下一张。



项目介绍、公共资助与链接 · 画面 00:00:46

00:00:31 · Sam Andreae

我们是 p2panda。这个软件项目从 2020 年开始，到现在一直在做。我们主要依靠公共资助，目前的资助来自欧盟和英国，包括 NGI、NLnet、Innovate UK，以及 eQualitie 的 Breakout Program。我们有自己的网站，代码目前放在 GitHub 上，这里还有我们在联邦宇宙的账号。



p2panda 团队 · 画面 00:01:36

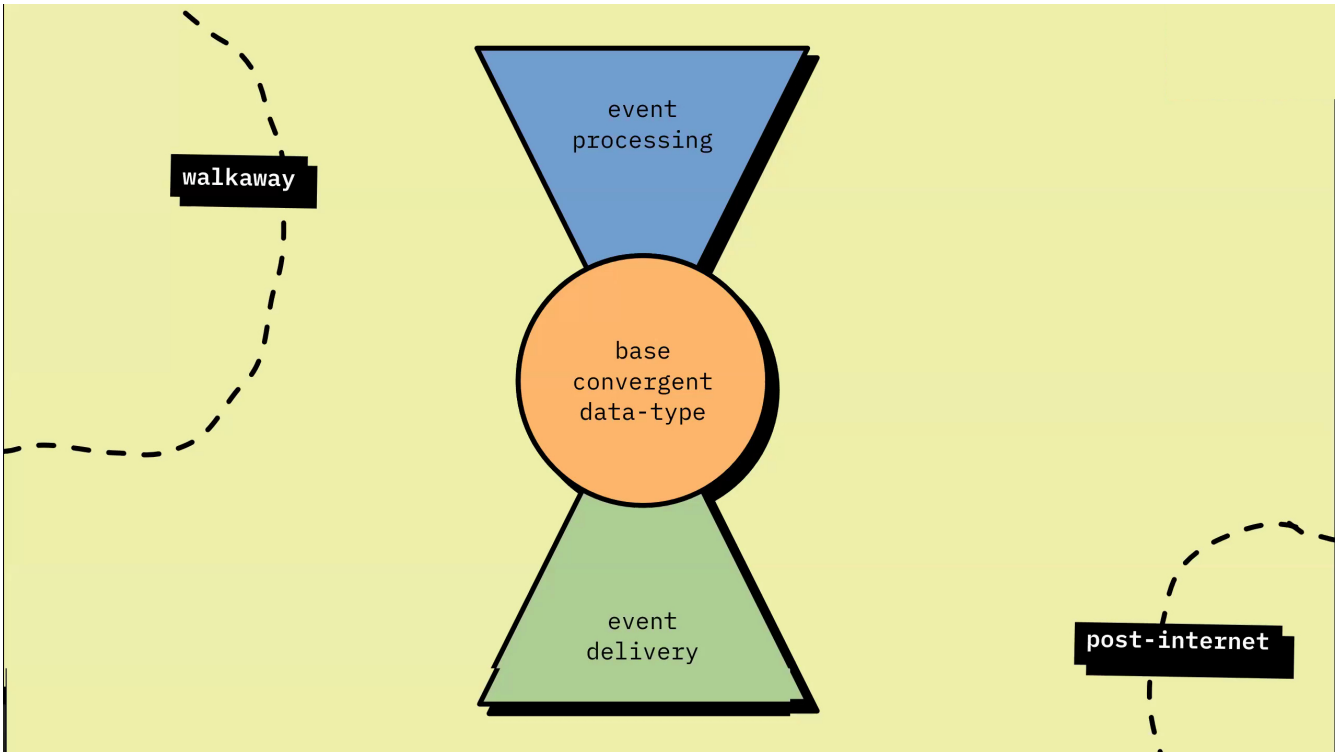
00:01:16 • Sam Andreae

我说到“我们 p2panda”的时候，核心团队就是我，Sam，我住在伦敦；还有 Glyph 和 Andreas，他们两位都住在柏林。我们参与 p2panda 都已经很多年了，在这之外，我们在 P2P 相关领域活动的时间还要更长一些。不好意思，请退回上一张。

00:01:56 • Sam Andreae

我们是核心团队，但正如大家接下来会看到的，我们开展了很多合作，社区里有许多项目。我觉得，积极参与 p2panda、为它作贡献的人，有二三十位甚至更多：有人写代码，有人做应用，也有人做设计和用户体验。我们的社区其实涵盖了不少不同的方向。

Walkaway Stack：可替换的三层架构



Walkaway 与 post-internet：事件处理、收敛数据类型、事件传递 • 画面 00:03:36

00:02:32 • Sam Andreae

首先，我想稍微讲讲我们的设计理念，也就是我们在 P2P 领域里怎样设计架构。这里有一个概念叫“Walkaway Stack”，这个说法并不是我们独自提出来的，周围的一些项目也参与其中。它围绕三个层次展开：最上面的蓝色三角形叫事件处理层；最下面的绿色三角形叫事件传递层；中间则是基础的收敛数据类型层。

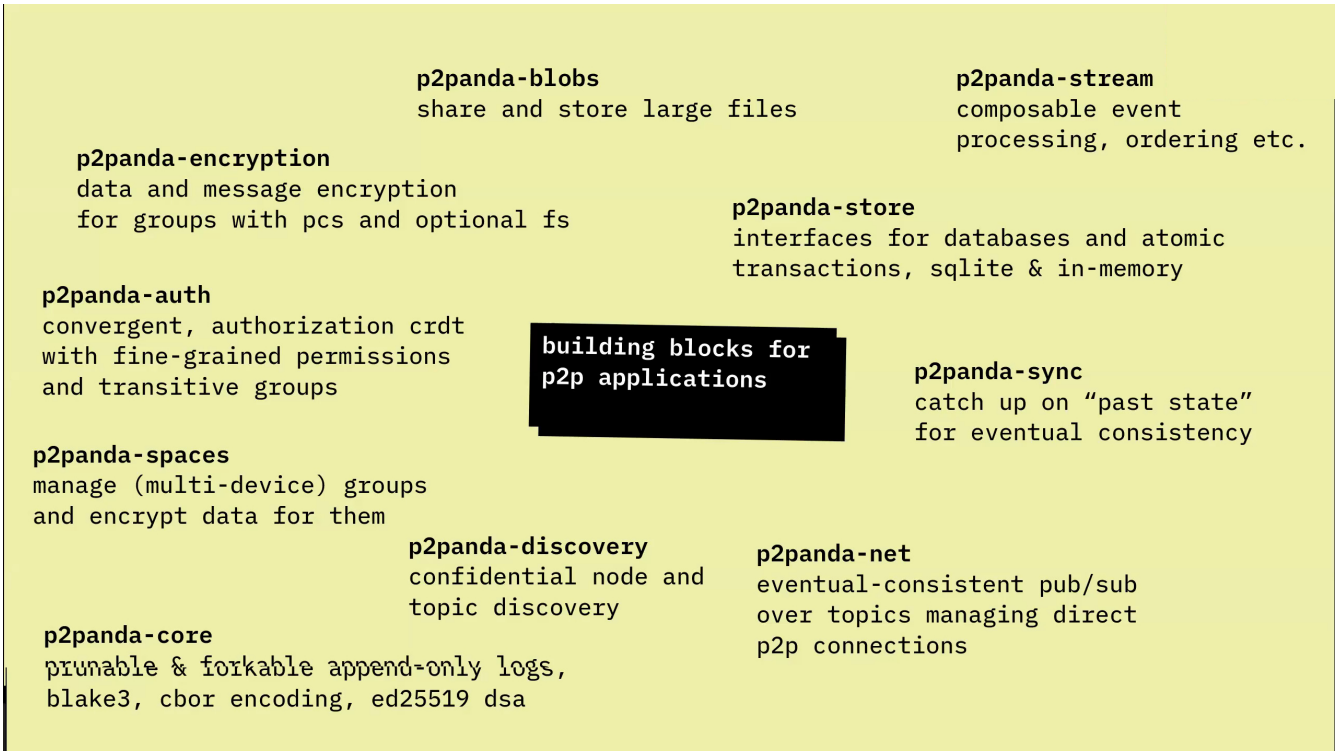
00:03:20 • Sam Andreae

它之所以叫“walkaway”，是因为有些层可以被拿走、替换掉，而其他层仍然应该能够工作。这里的关键在于，基础收敛数据类型和事件处理对应的，其实是用户数据与应用逻辑，属于应用自身的领域，也就是你的本地数据。事件传递层与这些部分分开，因此可以替换。

00:04:02 • Sam Andreae

这也指向了我们所考虑的那种“后互联网”环境。事件通常通过互联网传递，也可能经过服务器，但实际上，这一层可以换成很多其他的传输方式或路由协议。它们甚至可以记住自己在发送什么，设法把数据送到那些很难连接到的节点。不过，这不是上层需要操心的事。只要你收到了基础收敛数据类型所需要的内容，就可以正常工作。我们的很多思考都基于这个框架。下一张。

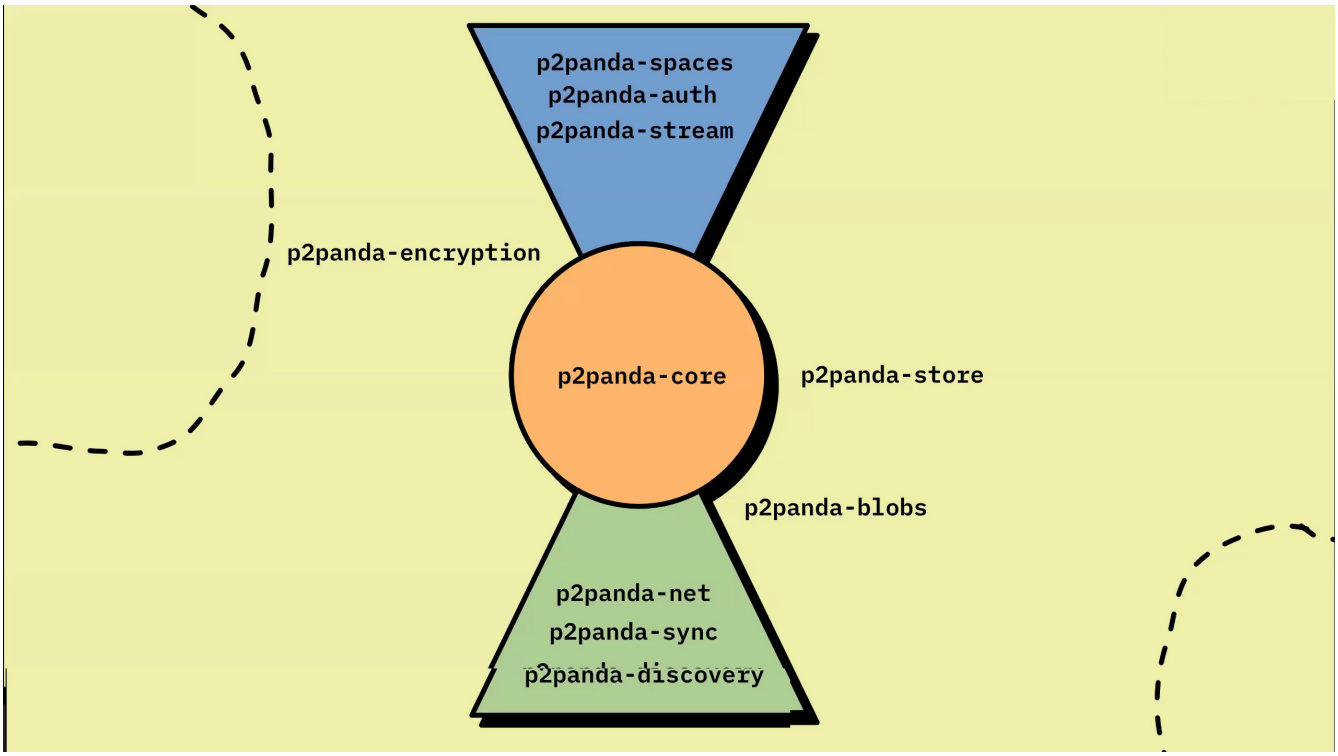
模块化 Rust 库与分层



p2panda 模块与职责 • 画面 00:05:16

00:04:53 • Sam Andreae

那具体是什么意思呢？我们在 p2panda 里究竟写些什么？我们写了很多 Rust crate。这张图里密密麻麻地列了一大堆，数量多得有点让人眼花。我们的系统非常模块化，尽量把不同职责分开。这是我们喜欢的一种软件开发方式，也确实从中受益。很多 crate 提供的是通用接口，并不一定依赖 p2panda 的其他部分。这有助于模块化，也呼应了刚才讲的“各个部分可以拆开”的想法。我翻到下一张；总之，这些就是我们那一组模块化的组件。



模块在沙漏架构中的位置 • 画面 00:07:56

00:05:53 • Sam Andreae

这里，我试着把这些组件放进 Walkaway Stack 的架构里。这张幻灯片是新的，今天才做，有些组件该放在哪里，并不是那么显而易见。不过，底下的事件传递层肯定包括 p2panda-net、p2panda-sync 和 p2panda-discovery。Discovery 用来以保护隐私的方式，发现和你对相同内容感兴趣的节点；Sync 负责实际交换数据；Net 则处理传输相关的各种事情，并协调前面两个层。这些都是 p2panda 提供的组件，也是我们自己写的 crate。

00:06:49 • Sam Andreae

不过，它们处在下面这一层，所以我们并不一定要依赖它们。我们可以换用其他传输方式，也可以采用不同的同步方法。最近我们做了一些更偏向网状网络的工作。在广播场景下，我们需要对同步采用稍有不同的思路。这就是刚才说的后互联网、纯广播的环境，也许只有无线电。我们提供这些组件，但再强调一下，上层并不依赖它们。

00:07:22 • Sam Andreae

p2panda-core 包含我们的核心数据类型。我们使用仅追加日志，同步过程也很大程度上依赖这种日志。我们在这里提供这些数据类型，但你仍然可以把它们拿出来，接上另一种传输方式，它们照样能够工作。再看上面蓝色的事件溯源层，可以把这里的大部分组件理解成事件溯源流水线中的处理环节。我们定义好预期接收哪些消息类型，只要这些消息以某种方式送到了，我们就能处理。

00:08:13 • Sam Andreae

Auth 负责去中心化环境中的群组管理，这是一个很难处理的问题。Spaces 则负责面向群组加密数据。它使用 p2panda-encryption；我把后者稍微放在了侧面，因为它可能涉及多个层次。这张图刚做出来，我觉得这里仍有些模糊之处：p2panda-encryption 处在事件溯源流水线里，但它也会对即将发送的内容进行加密，所以它对事件传递层同样非常重要。

00:08:52 • Sam Andreae

侧面还有 p2panda-store 和 p2panda-blobs，它们究竟归在哪一层，也不是完全清楚。所以，并不是每个组件都能划分得一清二楚。不过，可以说，这张图展示了这些 crate 在我们设计理念中的位置。下一张。

面向应用开发者的统一 API



开箱即用的 p2panda 高层 API • 画面 00:10:16

00:09:13 • Sam Andreae

并不是每个人都想逐个研究这些模块，再弄清楚怎么把它们接到一起。把这些部分整合好，也是我们的责任。因此，我们提供了 p2panda 这个顶层组件，也就是面向应用开发者、开箱即用的 p2panda API。上一张幻灯片里的东西都被整合进一个 crate，提供统一的 API。我们希望应用开发者能够很容易地使用它。它仍然遵循事件溯源的 API 模式，所以用它开发应用的人，大概也会希望沿用这种模式。不过，这已经是一种相当成熟的做法，并不是 P2P 领域独有的东西。

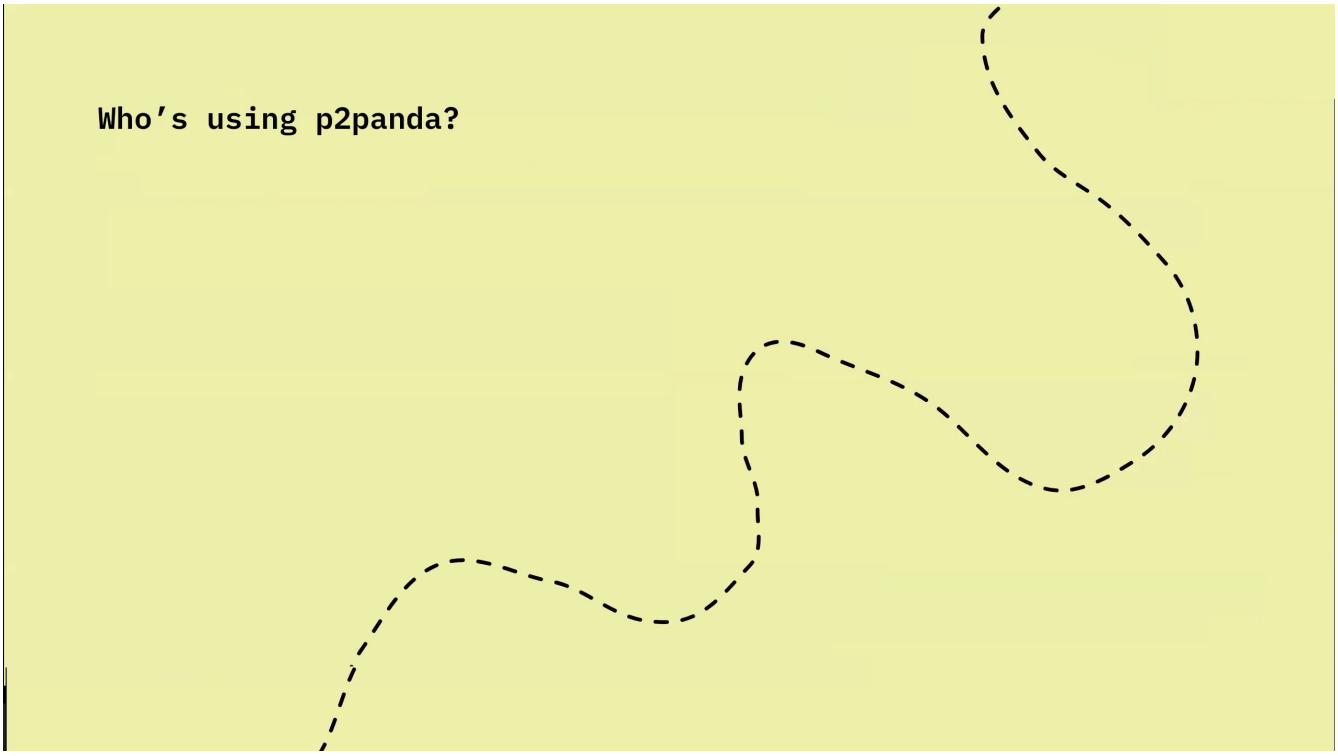
00:10:11 • Sam Andreae

它提供的能力都列在这里。这个整合方案包括：秘密主题的保密发现、高效同步、因果排序、事件流、临时消息、直接连接、群组加密、细粒度访问控制、多设备支持、崩溃恢复能力，以及垃圾回收。

00:10:38 • Sam Andreae

这里有一个小例子——哦，等一下，不好意思，请退回上一张。这里可以看到一个简单的 API 示例：先随机创建一个私有主题，再实例化一个节点。订阅这个主题时，你会得到一个流的发送端和接收端。这些流具有最终一致性。如果你向其中发送消息，另一个人稍后才来订阅，那么预期，也就是这里提供的保证，是你们最终会达到相同状态。如果对方漏掉了一些消息，最终也会收到。下一张。

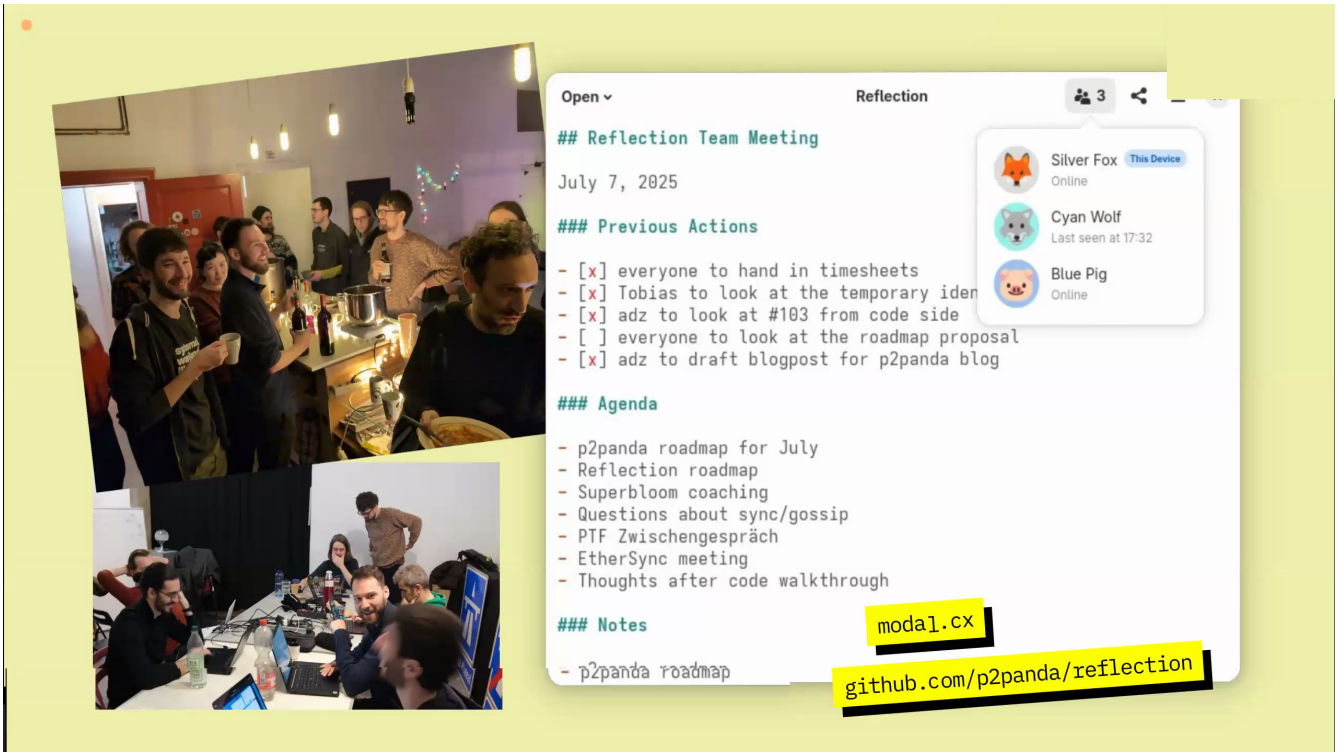
应用案例与协作社区



Who's using p2panda? • 画面 00:11:36

00:11:28 • Sam Andreae

就像前面说的，我们周围有一个很好的社区，其中很多人都在使用 p2panda。有些项目我们也参与其中，另一些则没有参与。我想在这里给大家介绍几个。都有谁在用 p2panda 呢？我来举几个例子。



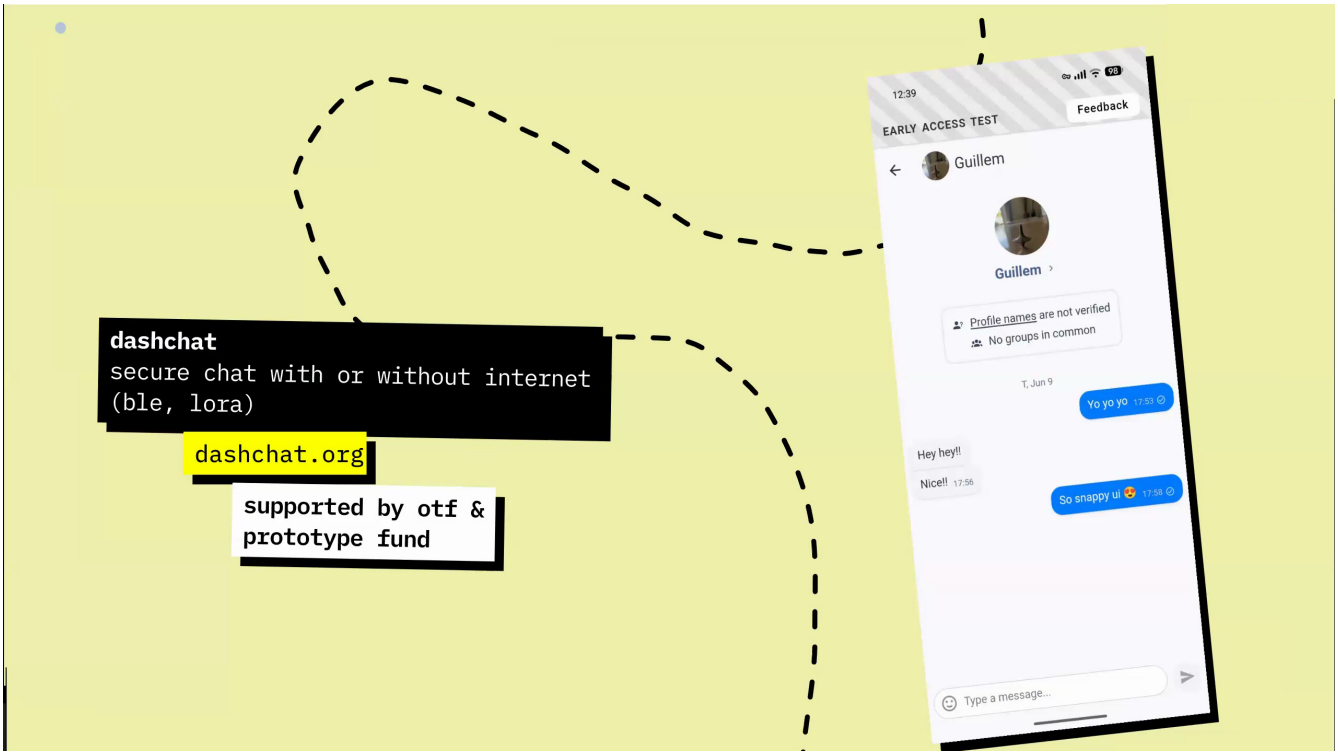
Reflection: 协作笔记 • 画面 00:12:16

00:11:51 · Sam Andreae

这是 Reflection，一个由 GNOME 社区成员开发的 GTK 应用，可以在 Flathub Beta 上找到。它使用 p2panda，最初就来自 GNOME 与 p2panda 之间的紧密合作。多年来，他们的反馈确实帮助我们逐渐形成了今天提供的这些 API。

00:12:31 · Sam Andreae

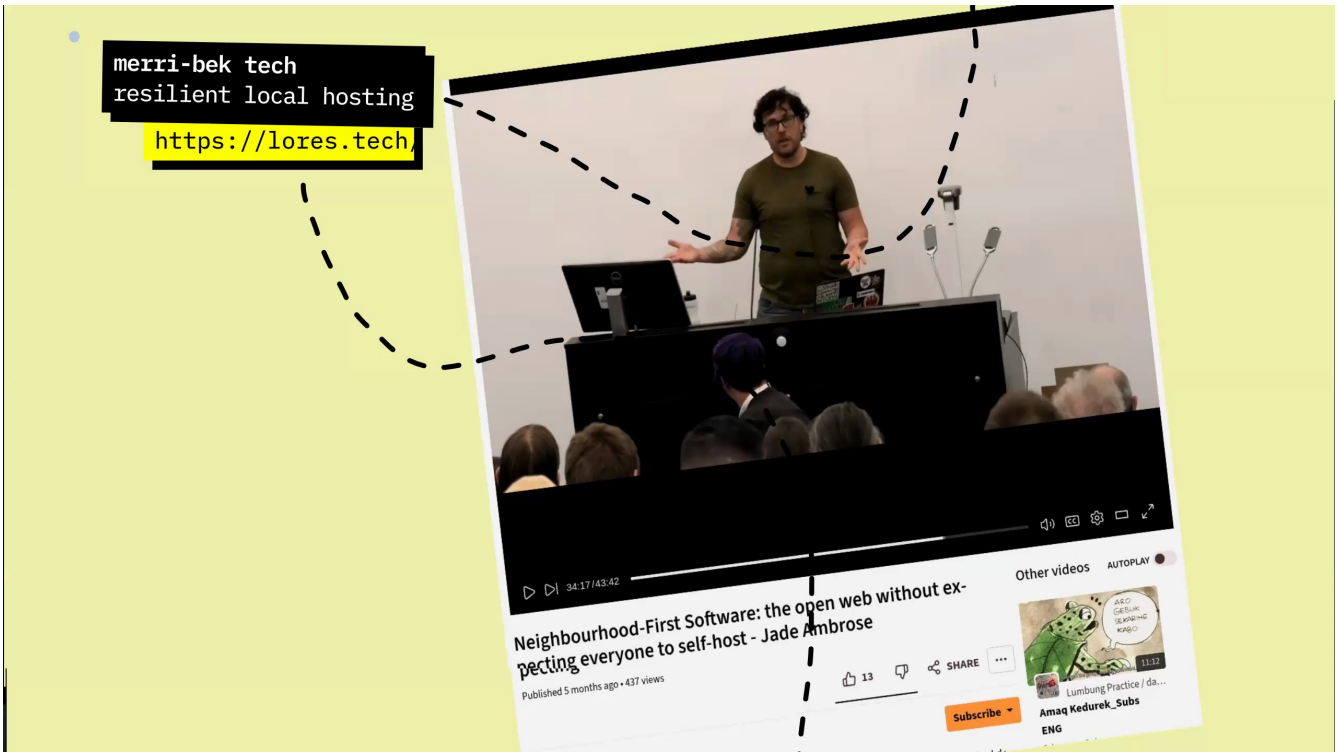
这是一个协作文本编辑器。我们每天工作都会用它，也知道其他项目在用。以目前的形态来说，它很稳定，也很可靠。当然，还有一些我们想加入的功能，不过我推荐大家试试看。下一张。



Dash Chat: BLE 与 LoRa 离线聊天 · 画面 00:13:16

00:13:04 · Sam Andreae

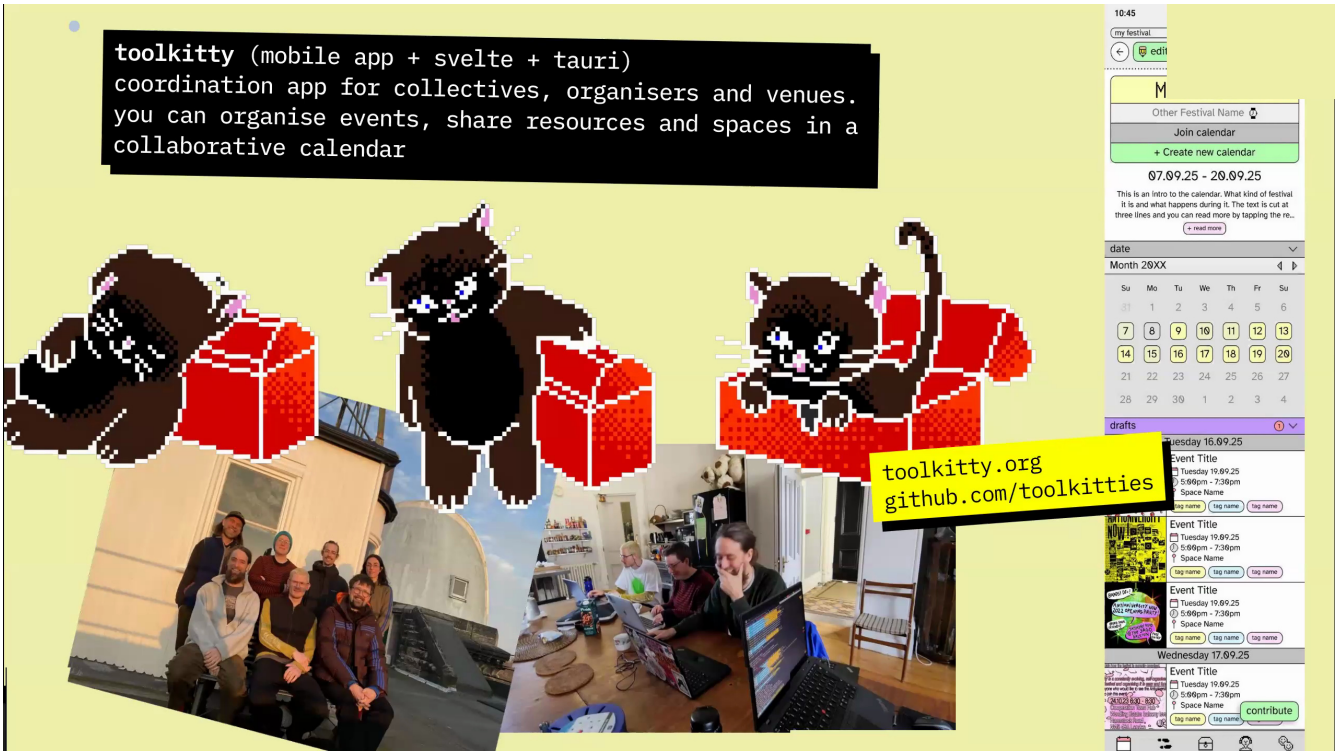
这是 Dash Chat 项目。这个团队正在基于 p2panda 开发一款安全聊天应用，有互联网或没有互联网都能使用。他们有自己的资助和其他资金来源。我们通过咨询协作的方式与他们紧密合作：我们帮助他们，他们也给我们的 API 提供很多反馈。他们推进得很快，我们一直在试用这个应用，感觉很不错。



Merri-bek Tech：具有韧性的本地托管 · 画面 00:14:26

00:13:54 · Sam Andreae

这是 Merri-bek Tech，幻灯片里可以看到 Jade。他们更关注面向本地社区、具有韧性的联邦式系统，尤其是在面对气候灾害的时候。我想，这是他们主要关注的方向：让社区内部仍能保持连接，然后在条件允许时，与其他社区建立联系。他们做了很棒的社区工作，教大家怎样部署自己的系统，也教大家怎样维护。我觉得这是一个很好的项目。下一张。



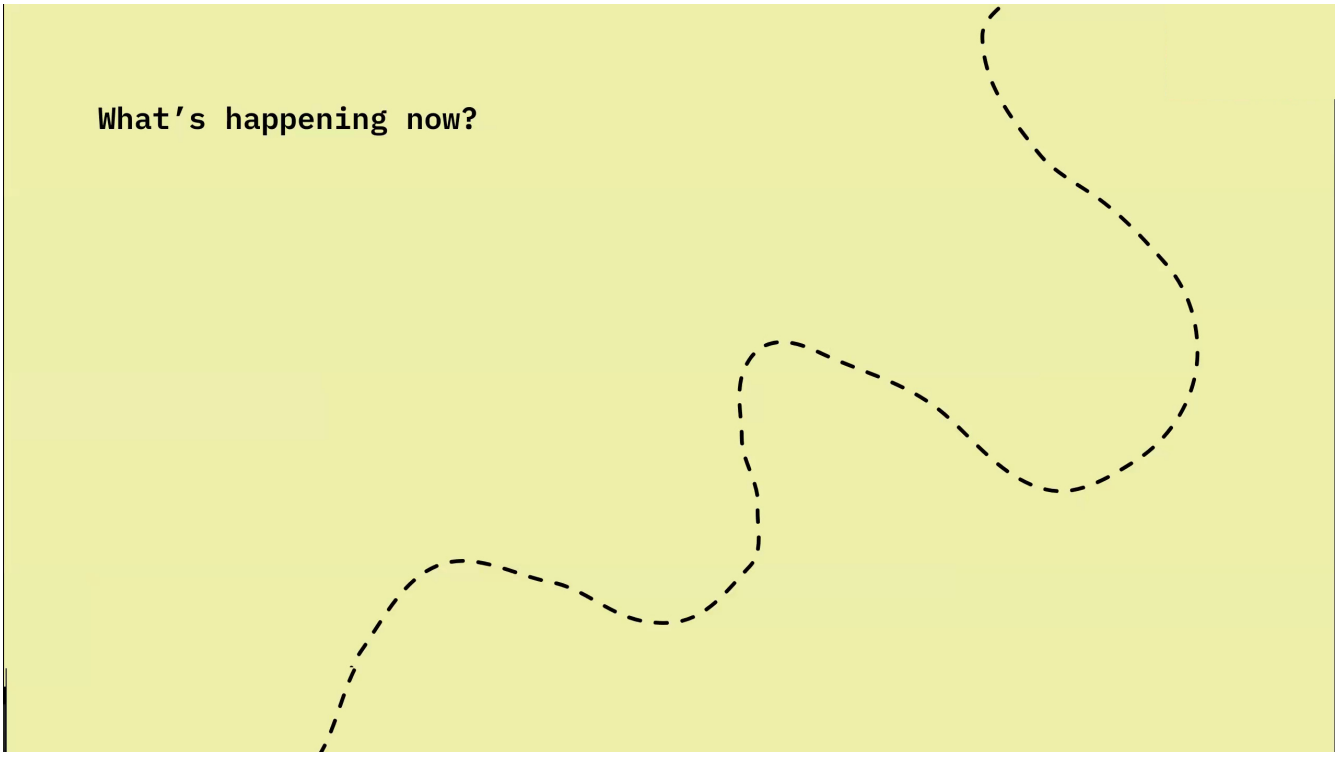
Toolkitty：组织、空间与资源协调 · 画面 00:15:56

00:15:02 • Sam Andreae

这是 Toolkitty。我们深入参与了它的设计和应用原型开发，项目由 Innovate UK 资助。它的起源甚至比 p2panda 还早。当时，我和 Andreas 在柏林参与音乐领域的节庆组织和出版活动。那时我们就已经对去中心化系统感兴趣，也希望用去中心化的方式来组织活动。Toolkitty 的前身叫 Hoffnung3000，是一个让参与者自行策划节庆活动的平台。

00:16:03 • Sam Andreae

Toolkitty 也类似，用来以去中心化的方式共享资源、组织活动，而且运行在去中心化网络上。现在，它基于 p2panda 运行。我们已经把它做到原型阶段，完成了不少不错的设计，也做过验证。不过，项目目前暂时停了下来。我们还需要一些资金，才能让它重新推进起来，我们正在为此努力。

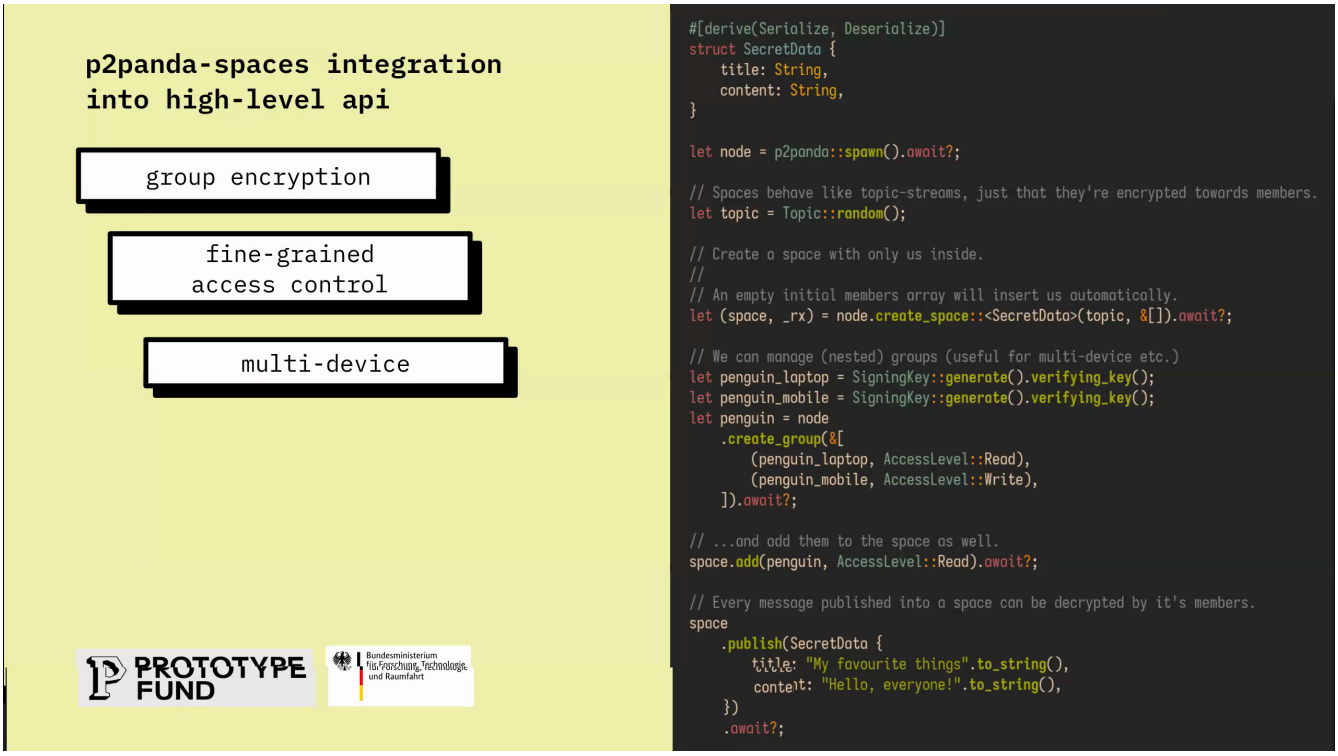


What's happening now? • 画面 00:16:51

00:16:44 • Sam Andreae

这些就是我们的一部分合作伙伴和相关项目。分享的最后一部分，我想讲讲 p2panda 目前在做什么。接下来有不少工作，我会逐一介绍。

Spaces：群组加密与权限



将 p2panda-spaces 集成进高层 API • 画面 00:17:46

00:17:03 • Sam Andreae

我们目前在做的一项主要工作，是把 Spaces，也就是带有群组加密的订阅，整合进高层 API。这样就能在 Spaces 本身提供的能力之上，加入重放、崩溃恢复等事件溯源 API 模式。Spaces 提供群组加密和细粒度访问控制，这些能力又能支持多设备，因为 Spaces 的群组管理允许嵌套群组，可以用这种方式来建模多个设备。

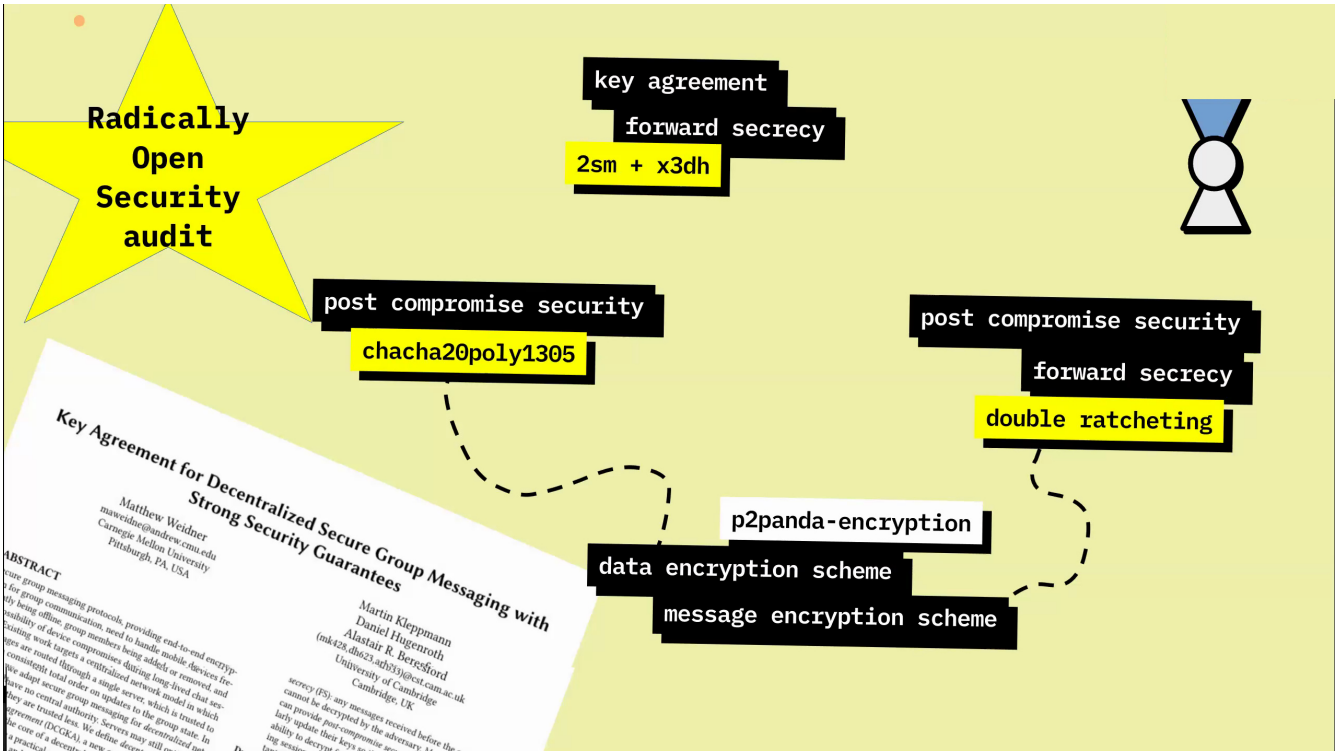
00:18:05 • Sam Andreae

这会带来许多功能。我们长期以来都觉得，要做出一个真正有用的 P2P 框架，这些功能至关重要。因此，这对我们来说是一个很大的里程碑，也让我们很兴奋。目前进展顺利，我们已经完成了很多工作，正在进入最后阶段。虽然还有不少事情要做，但整体推进得很好。这是一项由 Prototype Fund 资助的项目的一部分，同时也是与 Dash Chat 团队的合作。

00:18:50 • Sam Andreae

右边是一个 API 示例，和我稍早展示的例子类似。订阅一个主题，就和订阅一条流一样。在此基础上，额外提供的是群组管理 API。好，下一张。

密钥协商与安全审计



开放安全审计与加密设计 · 画面 00:19:46

00:19:18 · Sam Andreae

我们已经介绍过这些 crate。spaces 所做的一部分工作，就是把 p2panda-encryption 与提供群组管理功能的 p2panda-auth 整合起来。密钥协商协议实现在 p2panda-encryption 中，依据的是 Martin Kleppmann 等人的论文《Key Agreement for Decentralized Secure Group Messaging with Strong Security Guarantees》，也就是为去中心化安全群组消息提供强安全保证的密钥协商。

00:19:51 · Sam Andreae

我们很高兴，这份实现刚刚接受了 Radically Open Security 的安全审计，过程很顺利。我们收到了一些反馈，还有一些地方需要修改，但总体来说，这是一次非常积极的经历。我们希望很快就能分享这次审计及其结果。我们盼望这件事已经有好几年了，所以能走到这个里程碑，我们真的很高兴。

Support nodes：共享基础设施与隐私



支持节点：键值存储、配额与数据展平 • 画面 00:21:56

00:20:31 • Sam Andreae

接下来有两个项目与同一个主题有关，我先介绍这个主题：支持节点。这是本地优先或 P2P 应用开发中另一个非常重要的功能。它要解决的问题是：当所有人都离线时，你怎么取得数据？或者说，怎样提供一些共享的基础设施？我们把这样的节点称为支持节点。这里要考虑的是：在共享基础设施的同时，怎样仍然保护敏感数据的隐私？

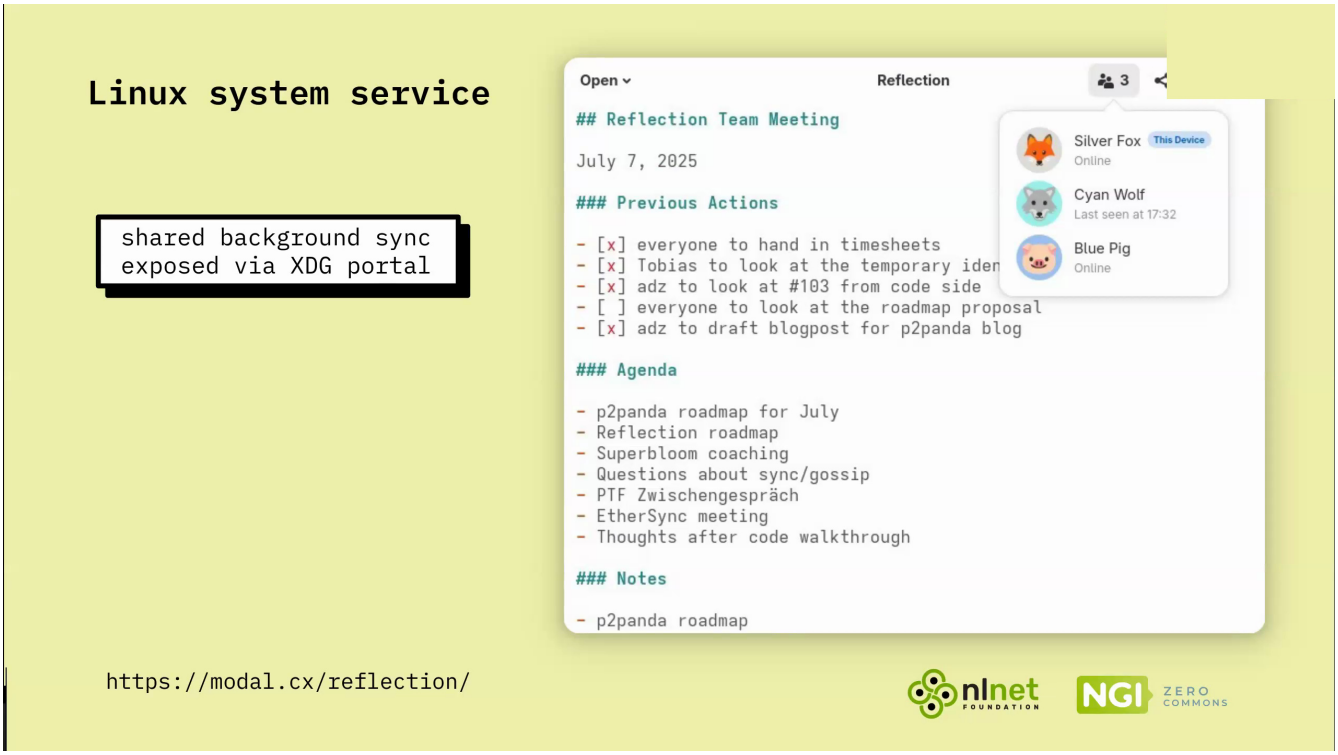
00:21:16 • Sam Andreae

借助刚才介绍的群组加密工作，我们现在可以把所有数据都加密。接下来的问题是：怎样存储和共享这些数据，同时尽量少暴露元数据？通过 spaces，我们可以对操作进行加密，几乎不暴露元数据。这样，支持节点的设计就可以非常简单：一个键值存储就够了。你可以直接把完全加密的数据放进去，支持节点甚至不知道这是 p2panda 的数据。它只知道：“我替你保存这些以内容哈希标识的数据，你以后可以来取。”

00:22:16 • Sam Andreae

支持节点不需要了解我们的数据类型和协议的具体细节，因此可以自行决定垃圾回收逻辑，或每个用户可以占用多少资源，例如使用 time-to-live（存活时间）或 space-to-live（可用存储空间）这样的限制。访问控制也可以采用支持节点实现所选择的任意机制，比如访问令牌。幻灯片展示了如何在这个存储中表示 p2panda 的操作。这些就是接下来两个具体项目的背景。

GNOME 的系统级 P2P 服务

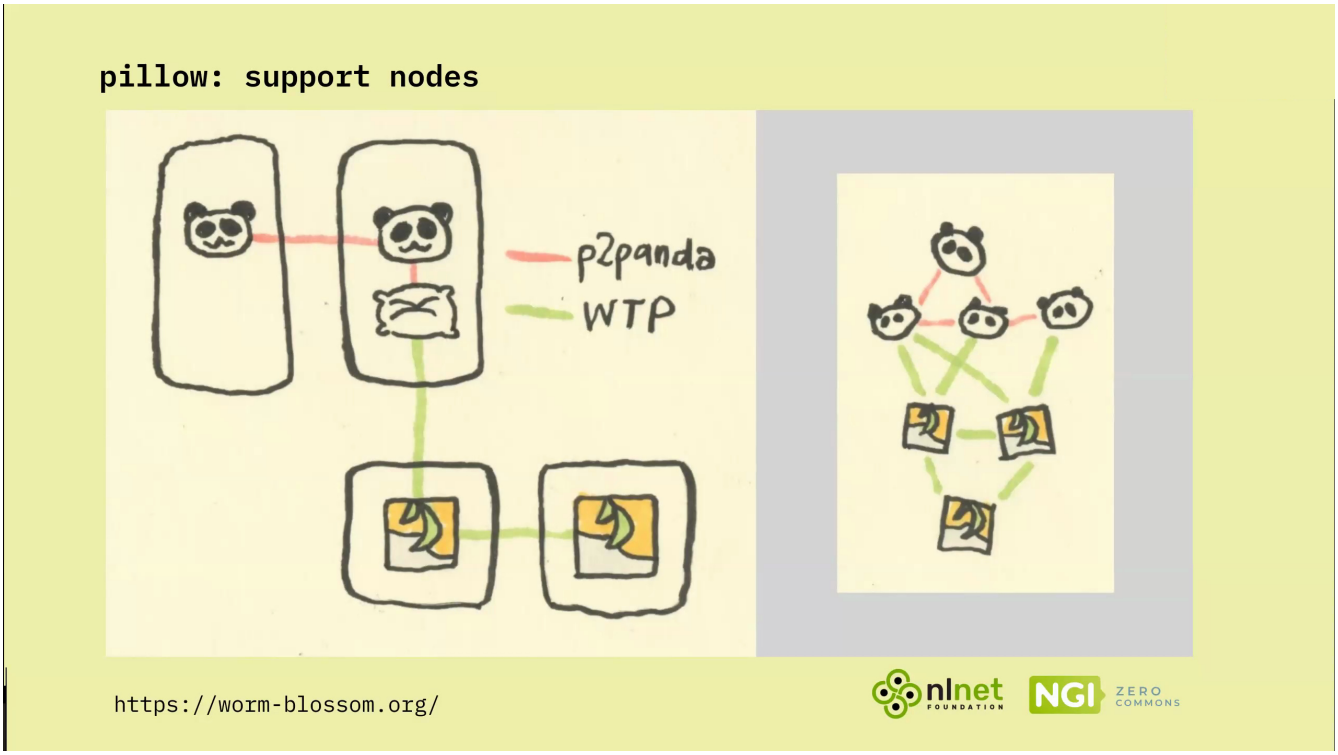


Linux 系统服务与 XDG portal • 画面 00:23:46

00:23:17 • Sam Andreae

第一个是 Linux 系统服务。这是一个由 NLnet 资助、同样与 GNOME 社区合作的项目。我们正在探索构建一个系统级的 P2P 服务，让应用通过 portal 接口申请使用。这个后台服务本质上就是一个支持节点，为系统中的多个参与方提供服务。因此，这个项目的一部分，就是实现一个作为系统服务运行的支持节点。这里的例子又用了 Reflection，因为它就是可能会使用这项服务的应用之一。

Willow 与 worm-blossom 合作

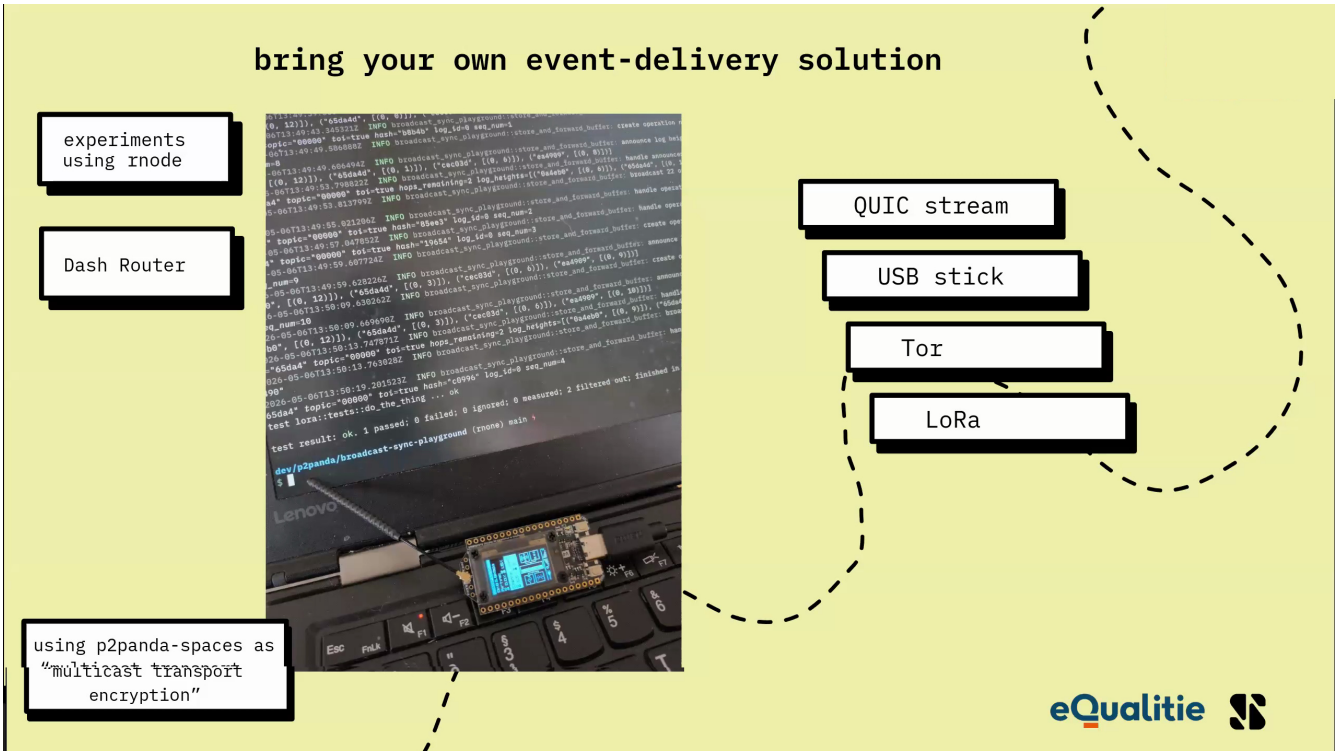


pillow：支持节点 • 画面 00:24:56

00:24:23 • Sam Andreae

另一个项目也围绕支持节点展开。这次我们与 Willow 协议背后的 worm-blossom 组织合作。他们是我们的老朋友，也是在这个领域并肩探索多年的伙伴。他们同样会实现支持节点，不过会使用自己的数据类型，处理支持节点之间的同步、与终端节点的同步，以及访问控制。他们有自己的访问令牌，也就是可委托的能力凭证。这样就可以实现一种模式：支持节点把访问权限授予某个系统服务，再由这个系统服务把权限进一步委托给系统中的应用。这里有一些很棒的系统示意图。

Dash Router 与多种网络传输



自选事件传递方案：QUIC、USB、Tor、LoRa • 画面 00:26:36

00:25:45 • Sam Andreae

再回到 walkaway stack 的架构。这里主要涉及事件传递，也就是下面那个可拆卸的层。我们目前有几个项目涉及这一层。其中一个是与 Dash Chat 的合作。前面谈到了他们将会使用的 spaces 集成，而这个项目的另一个方面是网状网络路由协议。他们正在实现 Dash Router，这是一种具有存储转发特性的网状网络路由协议方案：节点会替其他节点转发数据，或者先保存数据，再转发出去。

00:26:43 • Sam Andreae

另一个相关项目，是我们将要开发的 组播——或者说多传输方式——API，以及 p2panda 的一种新传输方式的具体实现，由 eQualitie 资助。这两个项目都可以理解为新的事件传递实现，而 p2panda 的技术栈已经完全准备好接入这些实现。

thank you!

any questions?

p.s. we host online “community hours” sessions from time to time (advertised on fedi and rocketchat)

these are a great chance to come and ask in-depth questions or just hear what we’re up to ^_^

问答与社区交流渠道 · 画面 00:27:56

00:27:26 · Sam Andreae

好，谢谢大家。这就是对 p2panda 及其相关项目的一次比较快速的介绍。接下来进入提问环节。我知道还有几位伙伴在这里，Andreas 肯定在，还有……嗨。哦。

00:28:08 · 主持人

还好吗？

00:28:11 · Sam Andreae

不好意思，我以为自己掉线了，不过好像还在。稍等一下。

00:28:18 · 主持人

没事。

00:28:21 · Sam Andreae

还听得到我吗？有点奇怪。好，太好了。非常感谢大家收听。我们现在进入提问环节，我很乐意回答大家的问题，我想 Andreas 和 Glyph 也都在这里。

00:28:38 · 主持人

大家有什么问题，可以现在开始提问。聊天框里已经有一个问题。

00:28:53 · Sam Andreae

好。

问答：根权限与日常管理员

00:28:54 · 主持人

问题是：“使用 p2panda-auth 时，你会建议怎样对彼此分开的根权限方和日常管理员进行建模？”

00:29:08 • Sam Andreae

这是个好问题。我想，这相当于在问：如何为超级用户管理员建模？这里可以使用嵌套群组。p2panda-auth 有嵌套群组的概念：你可以建立一个根群组，再把其他群组作为成员加入其中。这些群组的成员，也会通过这种传递关系成为根群组的成员。

00:29:47 • Sam Andreae

这样就可以用根群组来表示根权限方，再加入访问权限较低的成员。这些成员的访问权限永远不会超过根群组。这只是一个很简要的解释，具体使用场景当然还会涉及很多细节，但这是可以采用的一种方式。

问答：极低带宽网络的瓶颈

00:30:36 • 主持人

我有一个问题。当互联网不稳定，或者带宽极其有限时，比如使用 LoRa 或分组无线电通信，主要的瓶颈会是什么……？

00:30:56 • Sam Andreae

这是个好问题。我不确定我们是否已经有明确的答案。问题里提到的那些方面——有效载荷大小、元数据开销、密码学开销，或者数据对账——确实都是挑战。我也想听听 Andreas 或 Glyph 的看法，因为他们在不同的场景中都深入做过网状网络和加密方面的工作。我也许可以先说一点……还是先听听他们吧。Glyph、Andreas，你们对此有什么想法？

00:31:46 • Andreas Dzialocha

我可以谈一点看法。我觉得你刚才其实已经说到了：还需要更多模拟，以及在真实的大型网络中的部署，才能真正知道瓶颈在哪里。这些都还在研究当中。我们缺少这些系统在真实环境中的部署，因此还不能确切知道问题集中在哪里。

00:32:11 • Andreas Dzialocha

我们目前已经能看到、也确实在努力缩减的一个方面，是有效载荷。在 LoRa 中，每个数据包的上限是 255 字节。当然可以引入分帧机制，但我们也希望把实际操作的数据量压缩到这个大小。这很难，尤其是密钥协商方案，因为在开始通信之前，需要先发送很多密码学材料。这里肯定还有一些值得探索的取舍。

00:32:51 • Andreas Dzialocha

我们目前主要在改进两个因素：发送的消息数量，以及每条消息的大小。针对消息数量，我们正在研究一种数据对账或同步协议，灵感来自巴塞尔大学一位朋友的研究。我们尝试使用布隆过滤器，在节点之间交换状态向量。这个布隆过滤器可以缩小到 255 字节以下，用来表示一个包含 64 条消息的缓存——抱歉，我想是 128 条，或者说最多 128 条。第一次探索就能看到：我们可以用一条 LoRa 消息传递状态向量，这个结果很令人高兴。

00:33:52 • Andreas Dzialocha

仅追加日志也是类似的。如果一个应用只用了很少几条日志来组织数据，那么只需一条消息就能公布当前状态，内容就是序列号，或者你所持有的最新操作的哈希值。这些都是我们在考虑的做法。如果你对这个方向感兴趣，想看看在仅追加日志上探索得更深入的项目，我推荐 tinySSB。我可以把链接发到聊天框里。

00:34:27 • Andreas Dzialocha

他们也在尝试移除消息头中可能多余的元数据。比如，我们现在会在每条消息里重新发送公钥，显然不一定每次都需要这么做。也许只在第一条消息中附上公钥就够了。利用仅追加日志的结构，你可以回到日志开头去找到公钥。我们考虑的就是类似这样的做法。

00:34:55 • Andreas Dzialocha

我们也在探索另一种想法：不在每条消息里都附上签名，只在检查点附上，比如每一百条消息一次。这样会有一小段风险窗口，在这段时间里，你还没有验证签名或消息的真实性，而是等到检查点再验证，例如每十条、二十条消息设置一次检查点，诸如此类。

00:35:19 • Andreas Dzialocha

在我们的密码学方案中，这样还能再省下 64 字节。对于后量子密码学，这也很值得研究，因为签名可能有 2 KB 大。这些考虑在很多方面都很有意思，不过目前还只是一些想法。我觉得 tinySSB 团队在探索数据量究竟能压缩到多小这件事上，已经走得很远，甚至尝试通过——那个叫什么来着——无线电波，让分组无线电通信横跨太平洋。

00:35:59 • 主持人

好，谢谢。聊天框里又有一个新问题。

问答：浏览器能否作为独立副本节点

00:36:19 • Sam Andreae

我来读一下，然后试着回答。问题是：“你们尝试过在浏览器中运行 p2panda 吗？根据这些经验，浏览器能否使用 IndexedDB 和 core/auth/sync，作为一个独立的复制节点？在持久化和同步恢复方面，主要有哪些限制？”

00:36:46 • Sam Andreae

有的。p2panda 的一个早期版本其实非常侧重浏览器。我们当时采用客户端和节点的模式：客户端在浏览器里，与一个节点通信。常见的设计方式是把两者一起封装进一个应用，例如 Tauri 应用。我们当时还有一套设计取向更明确、基于对象的查询 API，对某些场景很好用，比如面向 Web 开发者的场景。

00:37:25 • Sam Andreae

不过，GNOME 社区反馈说，他们想要的其实是很不一样的架构。于是我们开始了长达几年的调整，逐渐走向这种事件溯源模式。当前形态的 p2panda 还没有实现浏览器支持，我们也还没有围绕如何把现在这套框架打包到浏览器里做很多实验。之前那一版，我们在那个技术上做过很多工作——我一时想不起名字了。

00:38:04 • Andreas Dzialocha

WebAssembly?

00:38:05 • Sam Andreae

对，也包括 WebAssembly。我们做过 Rust 绑定之类的工作，运行得不错。相关的一点是，我们目前也有面向很多其他语言的 FFI 绑定，不过那是稍微不同的话题。如果问这件事可不可行，我的回答是：当然可以。至于“持久化和同步恢复的主要限制是什么”，其实我不确定自己是否完全理解了这个问题。

00:38:47 • Andreas Dzialocha

也许是在问 IndexedDB 的限制，以及浏览器里能存多少数据。

00:38:53 • Sam Andreae

对。

00:38:54 • Andreas Dzialocha

这当然也取决于浏览器厂商。每种浏览器的限制都不同，所以很难给出一个确切答案。我记得有些浏览器允许每个作用域存储几个 GB 的数据。

00:39:13 • Andreas Dzialocha

到目前为止，包括以前的 p2panda 版本，我们的做法和建议都是：不要把太多工作和数据放进浏览器，原因很简单，使用体验可能会变慢、变卡。你当然可以巧妙地设计 Web 应用，比如使用 service worker，把工作从主循环中移出去。但这样一来，它就突然比传统 Web 开发复杂得多。有经验的开发者肯定能做出很好的东西，不过我们一直希望，做一个应用这件事能尽可能简单。

00:39:57 • Andreas Dzialocha

我们通常建议依靠代理。代理负责保存数据，它是一台在线、网页能够访问到的服务器。现在我们把它叫作 broker，也就是事件流代理。客户端可以连接 broker，从中消费数据。回到事件传递与事件处理分离的思路，浏览器中只需要放入编译成 WebAssembly 的事件处理层。它不负责同步，甚至不负责存储原始数据，只做处理，也许保存物化后的结果，而不保留那些数据包或操作本身。

00:40:52 • Andreas Dzialocha

这样可以减少网页或浏览器承担的工作，我觉得这是处理大型应用的一种办法。浏览器客户端还会保存一个状态向量，记录自己已经见过什么，类似同步协议的做法。它把自己的状态告诉 broker，broker 就能把缺少的操作以流的形式发送给网页。这样，多个网页即使处在不同状态，也能分别从 broker 获取最新数据。这是我们目前正在考虑的模式。

00:41:31 • Andreas Dzialocha

不过我也能想象，将来有人在浏览器中用 WebSocket 或 WebRTC 实现事件传递，让浏览器真正参与点对点同步和事件传递。只是，要让所有访问网页的人都获得良好体验，我觉得会更难。我们的朋友 Hyper Hyper Space 在这方面已经走了很远，但我记得他们后来也一度停下了，因为要突破浏览器沙箱的限制，并在移动设备上做好体验，实在很困难。当然，技术已经变了，智能手机也快了很多。谁知道呢？也许现在值得重新尝试。

问答：多个应用共用本机 P2P 服务

00:42:32 • Sam Andreae

我看到另一个问题，可以来回答。问题是：“有没有考虑过，让同一台设备上的多个本地优先应用共享一个本地节点或网络基础层，而不是每个应用都运行自己的发现、gossip、存储和身份栈？这能成为本地优先应用的共享基础设施吗？还是会让原本独立的应用之间产生太多耦合？”

00:43:00 • Sam Andreae

这个想法其实就是我在一张幻灯片中介绍过的内容：我们与 GNOME 社区合作的系统服务。你列出的发现、gossip、存储和身份，正是我们在讨论哪些层可以放进这个基础层。我们讨论了很多。在这个项目里，我们主要遵循 GNOME 社区的想法，或者说他们当下准备好去做什么，因为究竟哪些东西可以共享，还有很多问题。

00:43:48 • Sam Andreae

我们最初的做法是共享最少的一部分，在这里就是通过支持节点完成同步。项目中会有一个所有应用都与之通信的共享服务，也就是本地运行的支持节点。它可以代你与其他支持节点同步，每个应用再从这个节点取出自己关心的数据。这样，多个应用就会共享同一套基础设施、同一个同步层。

00:44:31 • Sam Andreae

共享身份服务是另一个我们讨论过很多的问题。我们想做，也非常感兴趣，GNOME 社区也是如此。不过它会带来更多关于隐私，以及在 GTK 环境中怎样设计用户体验的问题。我觉得这很有意思，也很有可能实现。但它同样很依赖具体的使用情境，其中有很多细节需要权衡。我认为让这些系统得到共享，是很值得做的。

00:45:28 • 主持人

好，谢谢。还有其他问题吗？大家有没有想到什么？

问答：社区应用与用户体验

00:46:17 • 主持人

我再问一个：把 p2panda 用到真实的社区项目中时，你们遇到过哪些特别有意思的情况？比如 Toolkitty 这样的使用场景。

00:46:58 • Sam Andreae

你是说，一些有意思的……

00:47:01 • 主持人

对。

00:47:02 • Sam Andreae

使用场景？

00:47:04 • 主持人

对。

00:47:05 • Sam Andreae

对。Toolkitty 是一个很有意思的场景。它会把 P2P 系统的并发特性带来的用户体验问题摆到面前，因为你在处理对某个资源的使用请求。这时，冲突已经不是文本文档里的“我在这里打了一个 A，你删掉了这个字符，来解决一下”了。它变得更贴近人的生活：“我想借你的车。我怎么知道请求已经被接受，而且没有别人抢先借走？”这些问题非常具体，而向潜在用户解释这些情况，也是一种挑战。

00:48:03 · Sam Andreae

大家已经习惯了即时响应：我按下“预订”，马上收到回复，预订就确认了。但这里实际上需要等待，等某个人收到你的消息，再把回复传回来。在那个项目中，我们对用户体验、界面设计，以及如何在 p2panda 上实现它们，都学到了很多。Andreas 和 Glyph，你们有没有特别喜欢的、富有挑战的使用场景？

00:48:57 · Andreas Dzialocha

对我来说，也许是一个更整体的问题，可以回到刚才关于守护进程、共同基础层或服务的讨论。这里有一个应用如何从零开始的问题：每个应用都要重新引入一套身份管理方式，告诉你怎样管理自己的身份、信任谁来交换数据。做一个新应用，这些事情就要重新发明一遍，而且每个应用的做法还略有不同。如果每次都变，用户很难接受和习惯。不过我认为，我们现在确实需要不断尝试不同做法，从中学习，因为最佳方式还没有被找到，也许根本不会有最佳方式。

00:49:57 · Andreas Dzialocha

无论如何，对我来说最大的挑战是：找出哪些共同模式可以统一起来，让应用直接依靠它们，不必一遍又一遍地重复做同样的事情。这可能是一个更底层、更整体的问题。

00:50:33 · 主持人

非常感谢。

00:50:37 · Sam Andreae

谢谢邀请我们。也谢谢大家来参加，提出这么好的问题。

00:50:46 · 主持人

还有一个问题。

00:50:50 · Sam Andreae

哦，好。

00:50:50 · 主持人

另一个问题。

问答：AI 工具与 UI/UX

00:50:52 · Sam Andreae

“你们有没有尝试过让 Claude 和 Codex 协作，来改进 UI/UX？”这个我可以回答：没有，我们还没有试过。

00:51:04 · Andreas Dzialocha

也补充一下，我们自己并不是 UI 或 UX 开发者。我们对此很感兴趣，会和设计师、UI/UX 开发者交流，也自己试过一些概念。对我们来说，保持与这个领域的联系很重要，否则就是闭着眼睛做开发了。通常，我们说到 UI 或 UX 开发时，实际是其他团队在做；他们会把学到的东西反馈给我们，和我们讨论发现。比如我知道 Dash Chat 会用 Claude 做前端，但我其实不知道他们在这方面用得怎么样。

00:52:02 · 观众

谢谢。

00:52:13 · 主持人

还有问题吗？聊天框里有一条新的消息。哦，那好像只是给另一位参与者的回复。

UX 补充：熟悉操作与网络健康

00:52:46 · Andreas Dzialocha

关于界面，我还有一个很普通但依然有用的建议：只要可能，就沿用大家已经熟悉的东西。删除按钮就应该还是删除按钮，哪怕在 P2P 系统中，这件事可能会变得有些奇怪。如果没法沿用大家熟悉的方式，就应该把差异非常明显地表现出来。如果你的系统有某种特别之处，就让它在界面中清楚可见。我们做 Toolkitty 时，尽量让操作直观：请求一个资源，等待期间显示一个小小的“待处理”状态，这些都是大家熟悉的东西。

00:53:41 · Andreas Dzialocha

不过，我们还考虑了“网络健康度”：我与其他节点同步了多少，交换了多少数据？我们觉得这是一个新概念，普通应用里没有，普通应用可能只是告诉你有没有离线。既然这是新的东西，我们就想到了 Tamagotchi，也就是电子宠物的做法：有一个小小的生物，你需要让它保持开心。你经常同步、不要总是离线、与别人交换数据，并且能帮其他人传递数据，它就会开心。这就是一个我们在界面里特别突出呈现的东西。

00:54:30 · Andreas Dzialocha

我相信，在身份管理、私钥管理这些方面，还有更多可以探索的东西，也有很多想法。大致就是这两类：尽可能沿用大家熟悉的东西；如果不能沿用的话……