Damus 调研文档

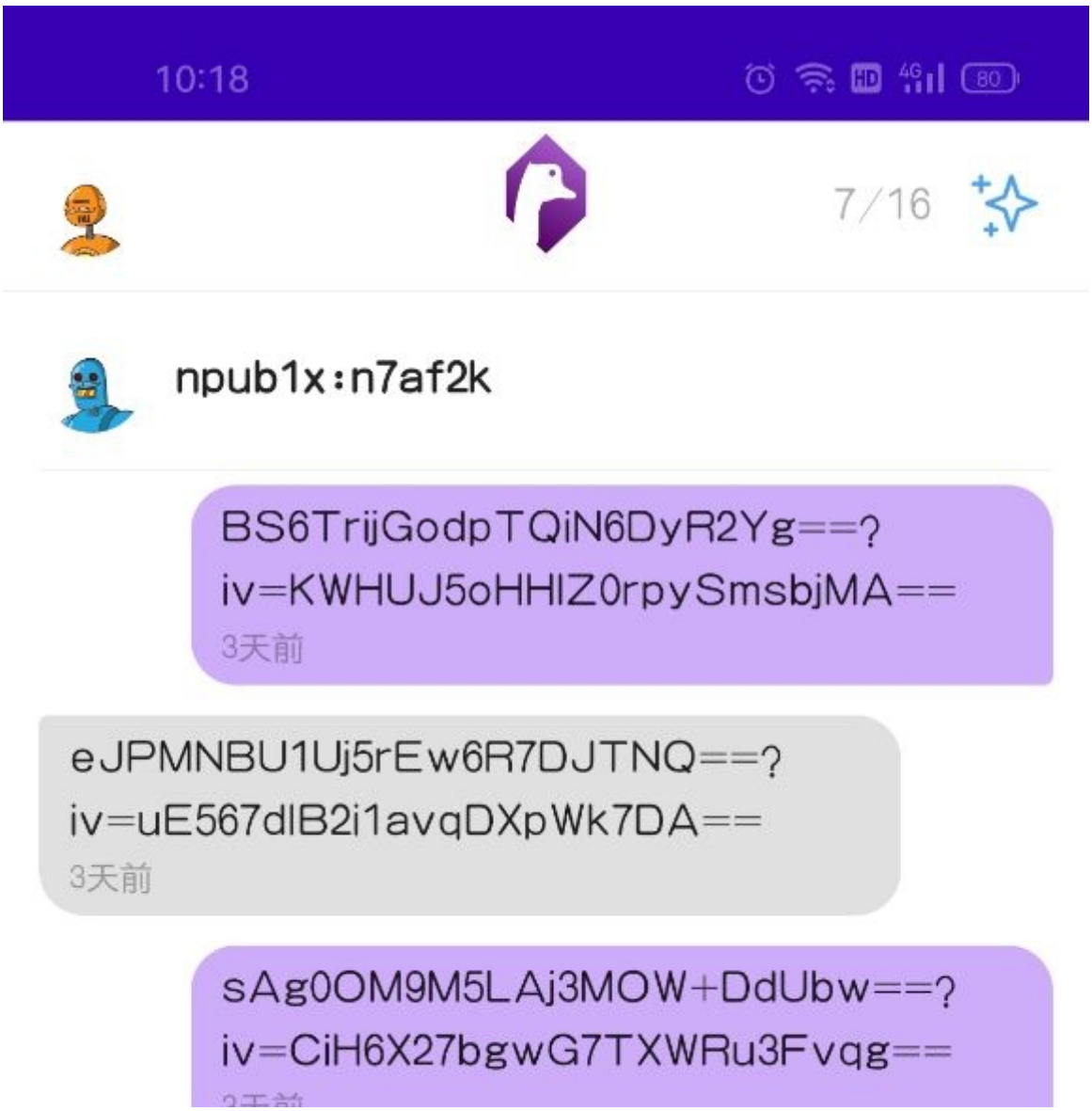
1. damus简介

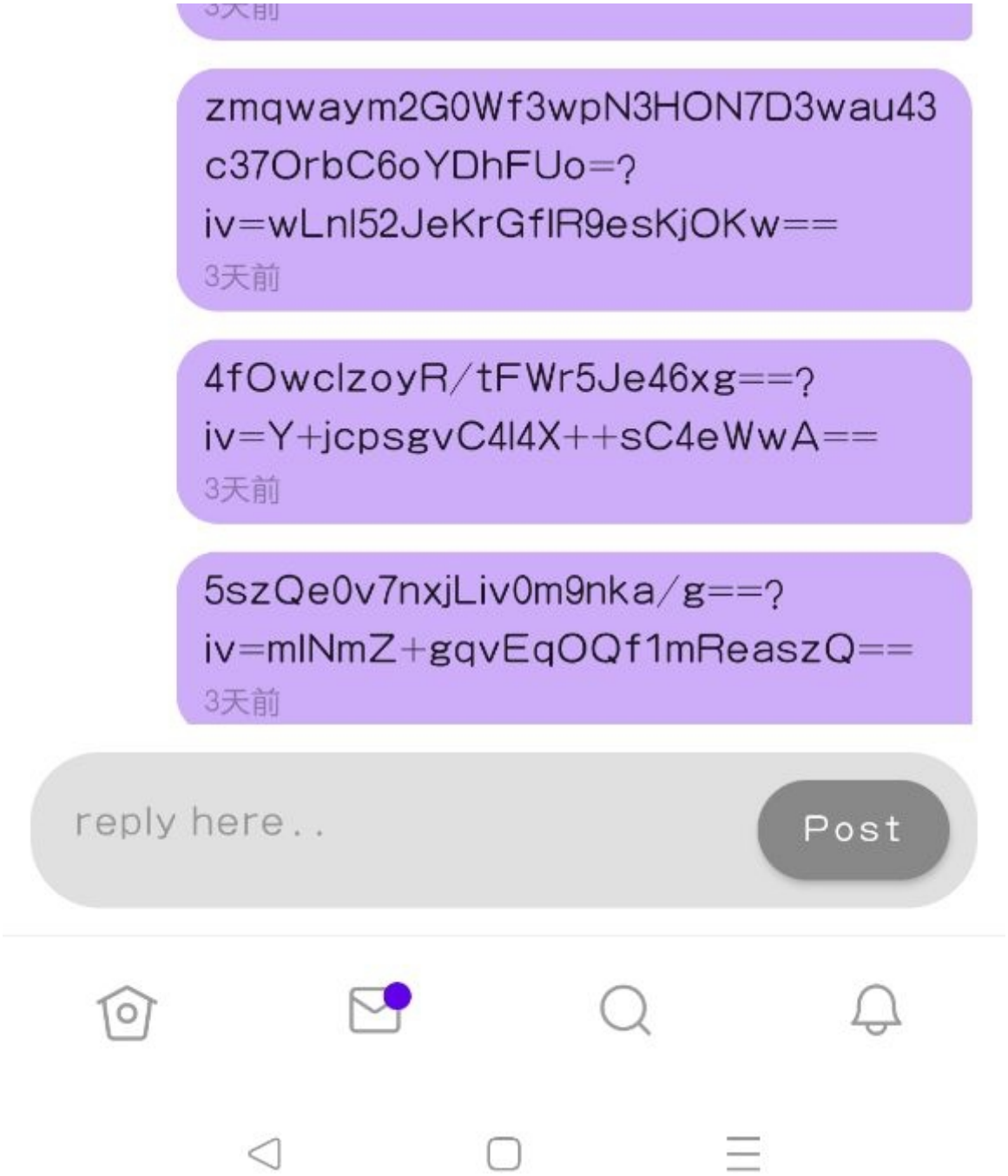
damus是一款去中心化的社交app（在ios上叫damus，安卓上叫Amethyst），类似于Twitter和微博，可以发布自己的贴文（Post），别人可以点赞，评论，相互关注后可以发私信等，但是所有的操作**不支持删除和撤回**。Damus是基于去中心化的协议**nostr**上的应用。

（1）匿名性：身份和登陆

注册：damus注册很简单，在ios上只需要填写昵称，安卓上连昵称都不用，就可以直接生成账号（基于公钥和私钥），注册时不需要手机，邮箱等私人信息。

登陆：登陆时可以输入公钥和私钥，基于私钥登陆，才是正常的登陆方式，可以解锁全部的功能；基于公钥登陆，可以看到该用户发布的贴文及回复，但是不能进行任何操作，不能发布贴文，可以看到私信会话，但是里面的内容是加密的





（2）基于公私钥的加密方式

damus注册后会生成一对公私钥，公钥相当于用户名，私钥相当于密码。

公钥：可以用于搜索某个用户，或者发布贴文or回复时，对发布的内容做签名，防止在网络传输过程中内容被篡改。

私钥：主要是用于登陆，以及相互关注的好友在私信时的内容加密，双方在私信时，会根据自己的私钥和对方的公钥，对聊天的内容进行加密和解密，实现点对点的私密通信。

（3）去中心化

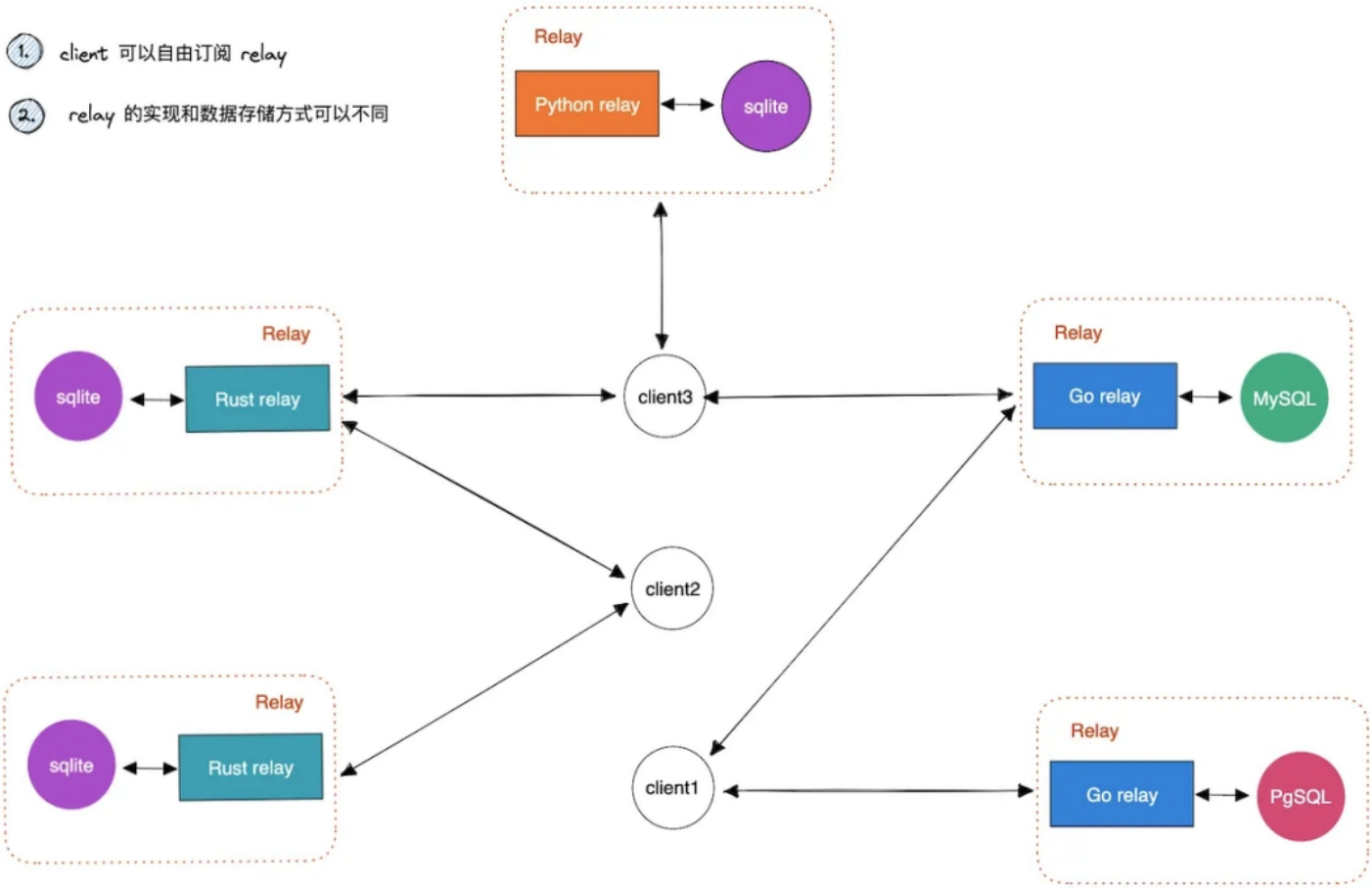
damus是基于nostr协议的去中心化app，**nostr协议没有用到区块链技术，不是基于链式的存储方式**，只是运用了公私钥的加密，将数据存储在分布式的中继节点上，每个中继节点并不互通，不需要同步数据，类似于数据库的分片。

2. nostr协议简介

Nostr是一个去中心化的分布式的网络协议，nostr网络中有两个角色client（客户端）和relay（中继服务器），用户通过client和中继服务器relay进行交互，数据在relay中存储，client和relay的交互方式采用websocket保持长链接，通过json格式传递数据。

每个relay（中继服务器）之间不需要同步数据，这一点与区块链节点有着本质区别，所以每个relay存储的数据库类型可以不同（mysql，pgsql，sqllite等都可以），不同的客户端在和不同的中继器交互时，只要传输的数据符合nostr协议约定的json格式就可以，与上层具体的应用（damus，astral等）无关。

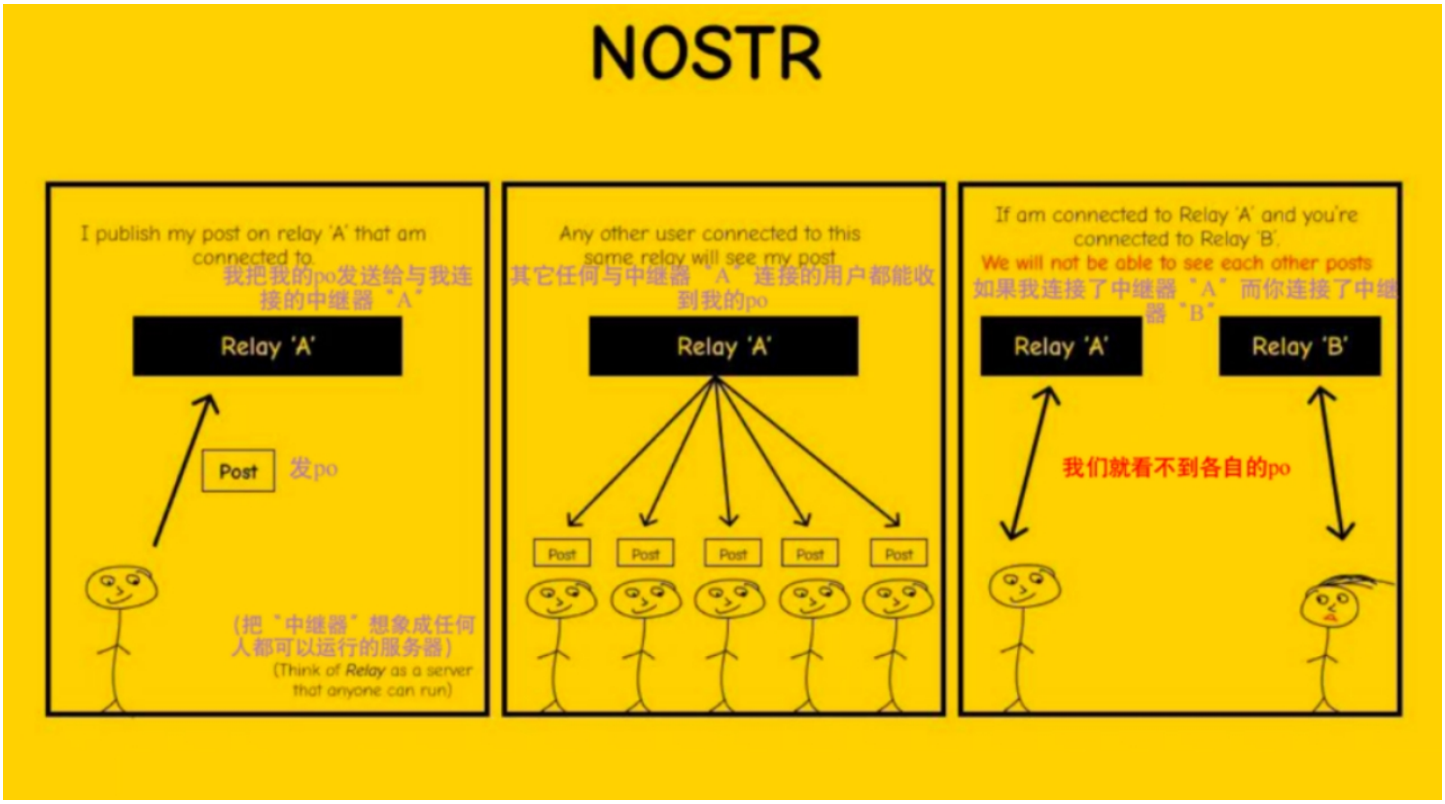
虽然 nostr 协议本身并不是建立在区块链之上，但大多数Nostr应用都支持通过比特币闪电网络进行支付。

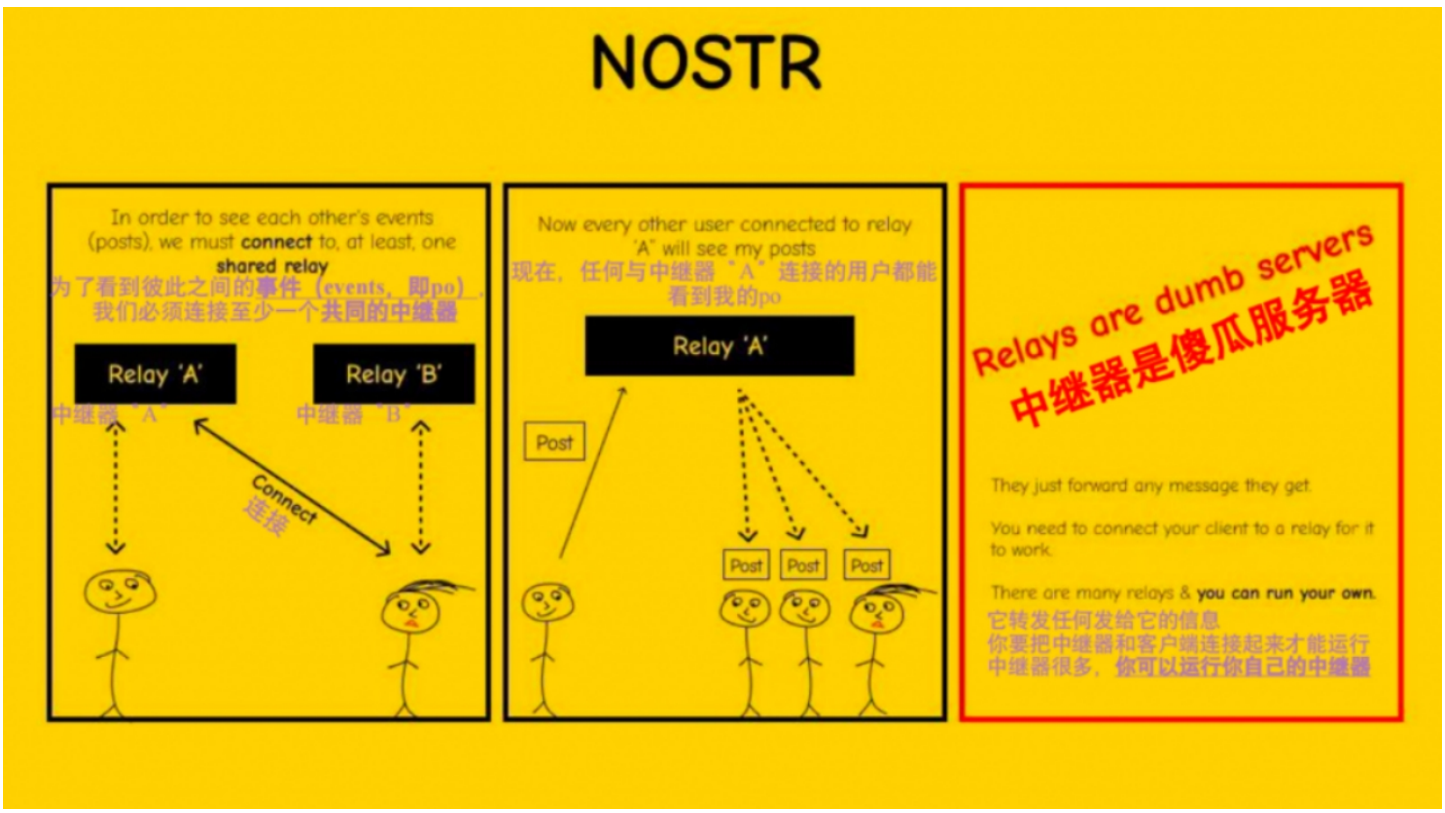


（1）client和relay之间的连接

客户端client用于签名、验证信息，由用户运行。中继服务器relay可以抓取、存储任何与它连接的客户端client的信息，并且转发给其他客户端，任何人都可以在互联网中部署自己的中继服务器。

两个用户，要想搜索到对方，或者看到对方发布的内容，**必须两个用户连接了至少一个相同的中继服务器地址**。





客户端允许用户与他们想要的**任何数量中继端相连**，用户还可以选择是否想要从自己所连接的中继端中读取、写入信息等等。这就意味着，我们可以连接某个中继端来检索内容，但是可以选择不在那里进行事件发布，或者反过来也成立。

在damus应用中，用户登陆后，可以看到默认就连接了十几个中继服务器，点开查看列表，可以删除其中的某些中继端，可以对某个中继端操作为只上传，或者只下载；还可以手动输入自己想要连接

的某个中继服务器URL地址。

client 可以将消息发布到任意的 relay，由于消息都经过了签名，relay 无法篡改这些消息，所以也不需要关心 relay 是否是可信的。在 nostr 中，如果一个 中继服务器 relay 把你封禁了，你也可以转发到其他的 relay。即使你是一个发布垃圾信息的人，也可以创建自己的 relay，然后发布消息，至于这些消息会不会被其他的 relay 和 client 抵制就是另外一回事了。

（2）client和relay之间的数据交互格式

每个人使用的终端是 client，client 会和 中继服务器 relay 之间进行交互，使用 websocket + json 。

- 从客户端client到中继端relay支持三种操作：

  1）EVENT：发布event，发布消息，修改个人简介等

  2）REQ：搜索数据，订阅relay的新消息，比如你关注的用户发布了新的贴文，client就可以通过这个req的请求获取到

  3）CLOSE：关闭在Req操作中的订阅

- 从中继端 relay 到客户端 client 支持两种操作：

  1) EVENT: 返回用户订阅的 event 信息

  2) NOTICE：返回可读的信息，这些信息的内容可以由 relay 自行决定

其中最重要的就是 EVENT 这个结构，用户有任何的数据新增或者修改，都是通过 event 来发布，为了防止 event 被篡改，发出的 event 都需要使用公钥签名，client 在收到 event 时会去验证这些签名。**所有需要被中继服务器 relay 存储记录下来的数据，都必须是 event 结构**。

nostr协议中，EVENT 有不同的类型，表示不同的作用，后续还在不断扩展。

kind	description
0	Metadata – 设置用户的元数据，比如用户名，头像等等
1	Text – 发布的消息，类似 twitter 的推文
2	Recommend Relay – 消息发布者推荐的一些中继服务器地址
3	Contacts – 当前用户的关注者列表
4	Encrypted Direct Messages – 加密消息
5	Event Deletion – 删除消息
7	Reaction – 点赞或者转发
40～44	Channel – 公共频道或者群聊
22242	Client Authentication – 中继服务器 relay 和 客户端 client 之间的鉴权

用户在发布消息时，都需要指定以上的一种类型，relay 会按照用户发送的类型来处理消息。

例如我们发布一条贴文，需要定义类型 kind为1的event，其Json结构如下

```
{
"id": "c011...4c43",
"pubkey": "dec1...4fb3",
"created_at": 1671551112,
"kind": 1,
"tags": [],
"content": "good morning!",
"sig": "e1dc...5f1"
}
```

（3）nostr网络的数据同步

在 nostr 中，有一个很重要的问题需要解决，由于 中继服务器 relay 之间不同步数据，那么数据要怎么在整个网络中同步呢？nostr 的设计很巧妙，每个 relay 之间不同步消息，同步消息的机制由 client 来实现。client 通过发送协议类型为 2和3 类型的 event事件（2–推荐的中继服务器URL列表；3–用户的关注者列表），可以把自己知道的用户及相关的 relay 地址传播到其他的 relay 中。这样其他的用户就可以通过这些信息找到目标用户的 relay，拉取到目标用户的消息，从而完成信息的传播。

这样做的好处在于 **每个 relay 不需要像区块链节点那样存储全量的数据**。nostr 的这种数据存储方式类似于数据库分片，让每个 relay 的存储压力不那么大。当然这样做也有坏处，如果用户的消息只在某一个 relay 上，如果这个 relay 出现问题，就有可能导致数据永久丢失。

3. 和我们的产品（Gold Spade）技术对比

	damus	Gold Spade
功能	类似于twitter，可以发帖，回复，关注，私信等	主要是IM即时通讯，私聊，群聊
账号体系	基于公钥和私钥	基于 sessionAccount 唯一标识
公私钥的作用	• 用户账号只和一对公私钥绑定，公私钥遗失，则账号丢弃 • client 通过公钥可以在关联的多个中继服务器中，搜索到该账号的贴文，回复，以及私信内容 • 公开的内容（贴文，回复等）需要用公钥做签名及验证 • 私信内容，需要基于自己的私钥和对方的公钥进行加解密	• 用户与公私钥不是一一绑定的关系 1. 用户第一次登陆时生成新的公私钥对，存储在客户端； 2. 当用户多设备登陆时，需要通过已经登陆的设备扫描授权 3. 已登陆的设备A，在授权新设备B登陆时，会在双方生成临时的DH公私钥 4. 设备A通过扫描确认，将自己存储的私钥，通过临时生成的DH

		公私钥对加密后，传递给新设备B，B通过临时的DH公私钥解密后，得到设备A的私钥，此时同一个账号的不同客户端就可以实现数据的正常同步（加密解密） 5. 如果新设备B登陆时，选择以新账号登陆时，会重新生成新的公私钥对，该用户的所有聊天记录会被清除，群关系也会被清除，但是好友关系会保留。 • 用户的公私钥主要应用于和好友聊天时，通过自己的私钥和对方的公钥，对聊天信息进行加密解密
数据的存储	• 去中心化存储，数据在client关联的多个不同的中继服务器relay中存储 • 两个用户能搜索到对方，或者查看到对方的贴文，必须这两个用户的client有关联到相同的中继服务器 • 每个client最好要关联多个中继服务器，避免因为单个中继服务器的故障或关闭，导致数据丢失	• 中心化存储，数据在我们自己的服务器存储