# Nostrプロトコルの現行仕様まとめ

## イベントの送信

Nostrでは、**イベント（event）**と呼ばれるJSONオブジェクトが情報伝達の基本単位です[[1]](https://nips.nostr.com/1#:~:text=)。各イベントには以下のフィールドが含まれます[[1][2]](https://nips.nostr.com/1#:~:text=):

{  
 "id": "<イベントID（SHA-256ハッシュ）>",  
 "pubkey": "<発信者の公開鍵（32バイトの16進数）>",  
 "created\_at": <作成時刻のUNIXタイムスタンプ（秒）>,  
 "kind": <イベント種別を表す整数>,  
 "tags": [ ... タグ配列 ... ],  
 "content": "<任意の文字列コンテンツ>",  
 "sig": "<署名（64バイトの16進数）>"  
}

idはイベントの内容を規定の方法でシリアライズした後、そのバイト列にSHA-256を適用して得られる32バイトのハッシュ値です[[1]](https://nips.nostr.com/1#:~:text=)。シリアライズには、[0,<pubkey>,<created\_at>,<kind>,<tags>,<content>]という配列構造をUTF-8エンコードしたJSON文字列にしてハッシュ化する手順が用いられます[[3]](https://nips.nostr.com/1#:~:text=To%20obtain%20the%20,of%20the%20following%20structure)[[4]](https://nips.nostr.com/1#:~:text=)。生成されたid（ハッシュ値）はイベントの一意識別子となり、イベントオブジェクト内にも格納されます。クライアントはこのidに対応するイベントデータ全体を**秘密鍵**で署名し、その署名をsigフィールドに含めます[[2]](https://nips.nostr.com/1#:~:text=)。Nostrプロトコルでは暗号技術としてsecp256k1曲線上の**Schnorr署名**（BIP-340標準）を使用しており、各ユーザーが保持する鍵ペアでイベントに署名します[[5]](https://nips.nostr.com/1#:~:text=Events%20and%20signatures)。これによりイベントの作成者認証と改ざん検知が行われます。

イベントの送信（パブリッシュ）は、WebSocket接続上でクライアントからリレーへ**EVENTメッセージ**を送ることで行います[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)。このメッセージはJSON配列で、フォーマットは次のようになります[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns):

["EVENT", <イベントJSONオブジェクト>]

例えば、上記のようなイベントオブジェクトを組み立て、自身の秘密鍵で署名したら、以下のようなメッセージを送信します（<...>は省略されたハッシュ値や鍵を示します）:

["EVENT", {  
 "id": "9cf...8e",  
 "pubkey": "8d4...98",  
 "created\_at": 1690000000,  
 "kind": 1,  
 "tags": [],  
 "content": "Hello, Nostr!",  
 "sig": "c1f...abc"  
}]

リレーは受信したイベントメッセージに対し、内容検証とポリシーチェックを行います。まず、イベントのidが正しく計算されていること、sigがpubkeyに対応する公開鍵で検証できる正当な署名であることを確認します。不正な署名やid不一致のイベントはリレーで拒否されます。また各リレーは独自のポリシー（例えば過去の重複イベント拒否、投稿者のBAN、Rate Limit、時刻異常など）に基づきイベント受理可否を判断します[[7]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,26%20is%20less%20than%2030)。イベントが受理された場合、リレーはそのイベントを保存し、購読中のクライアントへ配信します。さらにNIP-20で定義された**OKメッセージ**によって、クライアントに対し当該イベントを保存したか否かの結果が返されます[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)。OKメッセージは次の形式で、第三要素がtrueなら受理成功、falseなら拒否を示し、第四要素に機械可読な理由コードと人間向けメッセージが含まれます[[9]](https://nips.nostr.com/1#:~:text=,true%2C)（例："duplicate: already have this event"や"blocked: you are banned from posting here"など[[10]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,off%20from%20the%20current%20time)）:

["OK", "<イベントID>", true, ""]

上記のように、イベント送信時にはクライアントがイベントオブジェクトを生成・署名し、リレーに送信、リレー側で署名検証・保存・配信し、必要に応じ確認応答を返す、という流れになります。

## イベントの受信

クライアントがリレーからイベントを**受信**するには、まずリレーに対して購読（サブスクリプション）のリクエストを送信します。購読は**REQメッセージ**を用いて行われ、WebSocket接続上で以下の形式のJSON配列を送ります[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns):

["REQ", <購読ID>, <フィルタオブジェクト>, <フィルタオブジェクト>, ...]

<購読ID>（subscription\_id）はクライアントが任意に指定する識別子で、空でない最大64文字の文字列です[[11]](https://nips.nostr.com/1#:~:text=%60%3Csubscription_id%3E%60%20is%20an%20arbitrary%2C%20non,guaranteed%20to%20be%20globally%20unique)。このIDによって同一接続内の複数購読を区別します（各接続ごとにIDは独立に管理されます）[[11]](https://nips.nostr.com/1#:~:text=%60%3Csubscription_id%3E%60%20is%20an%20arbitrary%2C%20non,guaranteed%20to%20be%20globally%20unique)。REQメッセージには一つ以上の**フィルタ**を含めることができ、これによってクライアントが欲しいイベントの条件を指定します（フィルタの詳細は後述）[[12]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,used%20to%20stop%20previous%20subscriptions)[[13]](https://nips.nostr.com/1#:~:text=%60,can%20have%20the%20following%20attributes)。

リレーはREQを受け取ると、指定されたフィルタ条件にマッチする保存済みイベントを検索し、それらを**EVENTメッセージ**として送り返します[[14]](https://nips.nostr.com/1#:~:text=Upon%20receiving%20a%20,created%2C%20replacing%20the%20old%20one)。EVENTメッセージ（リレー→クライアント）は次の形式です[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other):

["EVENT", <購読ID>, <イベントJSONオブジェクト>]

ここで<購読ID>は、クライアントがREQで指定した購読識別子を指します[[15]](https://nips.nostr.com/1#:~:text=,messages%20received%20from)。リレーは各イベントに対してこのIDを付けて送り返すことで、どの購読に対するイベントかをクライアントが識別できるようにします。

最初のREQ送信後、リレーはまず**過去に保存されたイベント**の中からフィルタにマッチするものをすべて送信します。その初回バッチ送信が完了すると、リレーは**EOSEメッセージ**（End of Stored Events, NIP-15で定義）を送ります[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)。EOSEメッセージは["EOSE", <購読ID>]という形式で、指定購読について保存済みイベントの送信終了をクライアントに通知します[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)。クライアントはこれを受け取ることで、過去分のイベント受信が一段落し、以降は新着イベントのストリームになることを認識できます。

EOSE送信以降、リレーは**リアルタイムに新規受信したイベント**でフィルタにマッチするものを逐次EVENTメッセージで配信します[[14]](https://nips.nostr.com/1#:~:text=Upon%20receiving%20a%20,created%2C%20replacing%20the%20old%20one)。このサブスクリプションは持続し、クライアントが明示的に停止するか接続が閉じられるまで、条件に合致する新しいイベントを送り続けます[[14]](https://nips.nostr.com/1#:~:text=Upon%20receiving%20a%20,created%2C%20replacing%20the%20old%20one)。

クライアント側で購読を停止したい場合、**CLOSEメッセージ**をリレーに送ります[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)。CLOSEは["CLOSE", <購読ID>]という形式で、指定した購読IDのサブスクリプションを終了するリクエストです[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)。これを受け取ったリレーは、該当購読に対する今後のイベント送信を停止します。リレー側が何らかの理由でサーバー側から購読を打ち切る場合（例えばフィルタ内容が不正・非対応、負荷制限など）には、リレーからクライアントへ**CLOSEDメッセージ**が送信されます[[16]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,or%20other%20things%20to%20clients)。CLOSEDメッセージは["CLOSED", <購読ID>, <理由>]という形式で、購読IDに対応する購読が終了したことと、その理由（auth-requiredやunsupported等のプレフィックス付きメッセージ）を通知します[[17]](https://nips.nostr.com/1#:~:text=,are)。

このように、イベント受信の流れは (1)クライアントがREQで購読開始→(2)リレーが過去イベントEVENT群を返送→(3)区切りにEOSE送信→(4)新着イベントをリアルタイムEVENT送信→(5)必要に応じCLOSE/CLOSEDで購読終了、となります。

## フィルターの使用方法

NostrのREQ購読要求では**フィルター**を指定して、受信するイベントを絞り込むことができます。フィルタは一つのJSONオブジェクトで、以下のようなキーを含みます[[18]](https://nips.nostr.com/1#:~:text=,ids)[[19]](https://nips.nostr.com/1#:~:text=,to%20this%20to%20pass):

* ids: イベントID（64文字の16進数）**のリスト**。このリスト内のいずれかに前方一致するIDを持つイベントだけが対象になります[[20]](https://nips.nostr.com/1#:~:text=Filter%20attributes%20containing%20lists%20%28,least%20one%20item%20in%20common)[[21]](https://nips.nostr.com/1#:~:text=The%20%60ids%60%2C%20%60authors%60%2C%20%60,character%20lowercase%20hex%20values)。
* authors: 発行者の公開鍵（64文字の16進数）**のリスト**。イベントのpubkeyがこのリスト内に含まれる場合にマッチします[[22]](https://nips.nostr.com/1#:~:text=,ids)[[20]](https://nips.nostr.com/1#:~:text=Filter%20attributes%20containing%20lists%20%28,least%20one%20item%20in%20common)。
* kinds: 種別番号（kind）**のリスト**。イベントのkindがリスト内のいずれかの値と等しい場合にマッチします[[23]](https://nips.nostr.com/1#:~:text=,ids)[[20]](https://nips.nostr.com/1#:~:text=Filter%20attributes%20containing%20lists%20%28,least%20one%20item%20in%20common)。
* #<letter>: 特定のタグの値**のリスト**。#eならイベント内の「eタグ」（他イベントへの参照ID）にリスト中のいずれかのIDが含まれるイベント、#pなら「pタグ」（他のユーザーへの参照公開鍵）にリスト中のいずれかの鍵が含まれるイベント、というようにタグごとに指定できます[[24]](https://nips.nostr.com/1#:~:text=,kind%20numbers)[[20]](https://nips.nostr.com/1#:~:text=Filter%20attributes%20containing%20lists%20%28,least%20one%20item%20in%20common)。#に続く一文字のアルファベットタグ（例:e,p）についてはリレーがインデックスを構築しているため、高速な参照が可能です[[25]](https://nips.nostr.com/1#:~:text=As%20a%20convention%2C%20all%20single,any%20given%20tag%20is%20indexed)[[26]](https://nips.nostr.com/1#:~:text=,bytes%20lowercase)。
* since: Unixタイムスタンプ（秒）を指定します。この時刻以上にcreated\_atが新しいイベントをマッチ対象とします[[27]](https://nips.nostr.com/1#:~:text=event%20ids%2C%20for%20,a%20list%20of%20pubkeys%2C%20etc)[[28]](https://nips.nostr.com/1#:~:text=lowercase%20hex%20values)。
* until: Unixタイムスタンプ（秒）を指定します。この時刻以下にcreated\_atが古いイベントをマッチ対象とします[[19]](https://nips.nostr.com/1#:~:text=,to%20this%20to%20pass)[[29]](https://nips.nostr.com/1#:~:text=lowercase%20hex%20values)。
* limit: 数値Nを指定すると、保存済みイベントの初回取得において最大N件までのイベントを返すようリレーに要求できます[[30]](https://nips.nostr.com/1#:~:text=,return%20in%20the%20initial%20query)[[31]](https://nips.nostr.com/1#:~:text=The%20,avoided%20to%20prevent%20overwhelming%20clients)。この制限は初回クエリのみに適用され、リアルタイム配信には影響しません[[31]](https://nips.nostr.com/1#:~:text=The%20,avoided%20to%20prevent%20overwhelming%20clients)。

**フィルタの論理**は、1つのフィルタオブジェクト内では指定した複数条件がすべて満たされるイベントのみが通過するAND条件として扱われます[[32]](https://nips.nostr.com/1#:~:text=All%20conditions%20of%20a%20filter,conditions)。例えば{"authors":["ALICE"], "kinds":[1]}というフィルタは「発行者がALICEでかつkindが1のイベント」にマッチします。一方、REQメッセージで複数のフィルタオブジェクトを配列で渡した場合、それらフィルタの間はOR条件として扱われ、いずれかのフィルタにマッチすればイベントが返されます[[32]](https://nips.nostr.com/1#:~:text=All%20conditions%20of%20a%20filter,conditions)。例えば["REQ","sub1", {"authors":["ALICE"]}, {"authors":["BOB"]}]とすれば、「ALICEによるイベントまたはBOBによるイベント」を購読できます。

フィルタを活用することで、クライアントは必要な種類・範囲のイベントだけを効率よく取得できます。例えば、特定ユーザーの最新投稿のみ取得したり、ある話題（特定タグを含むイベント）のみ購読したり、一定期間のイベントログを遡って取得する、といった指定が可能です。フィルタ構文は基本的にNIP-01で規定されており[[13]](https://nips.nostr.com/1#:~:text=%60,can%20have%20the%20following%20attributes)、NIP-12などでタグフィルタの利用方法が補足されています。またフィルタキーはリレー実装によって拡張されうるものの、上記が標準的な項目です。

## 認証の仕様（署名と公開鍵の扱い）

**鍵ペアと署名:** Nostrではユーザーごとに秘密鍵と公開鍵からなる鍵ペアを持ち、自身の公開鍵（32バイトの16進表現）がそのままユーザー識別子となります[[33]](https://nips.nostr.com/1#:~:text=%22pubkey%22%3A%20%3C32,key%20of%20the%20event%20creator)。各イベントは秘密鍵によるデジタル署名sigが付与されており、対応する公開鍵pubkeyによって検証可能です[[2]](https://nips.nostr.com/1#:~:text=)。署名アルゴリズムにはsecp256k1曲線上のSchnorr署名（64バイト署名）が採用されており[[5]](https://nips.nostr.com/1#:~:text=Events%20and%20signatures)、署名の対象となるハッシュは前述のイベントシリアライズデータ（すなわちイベントIDと同一のSHA-256ハッシュ）です[[2]](https://nips.nostr.com/1#:~:text=)。リレーサーバーは受信したイベントについて、この署名検証とid再計算による検証を行い、**改ざんや成りすましがないことを確認した上で保存・配信**します。不正な署名や無効なイベントは破棄され、OKメッセージで失敗理由（例：「invalid: ...無効なイベント」等）が通知される場合があります[[7]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,26%20is%20less%20than%2030)。Nostrのこの仕組みにより、各クライアントは**公開鍵をユーザーの身元証明**として信頼し、リレーや第三者に依存しない検証性を実現しています。

**アイデンティティと署名管理:** Nostrネットワーク上ではユーザー登録やパスワード認証といった概念はなく、公開鍵がユーザーIDとして機能し、対応する秘密鍵の所有がそのユーザー性を担保するシンプルな設計です。クライアントはユーザーの秘密鍵を用いてイベント署名を行いますが、秘密鍵の保護や管理は各クライアント実装に委ねられています。例えば、**NIP-07**ではWebブラウザにおける署名用APIが定義されており、window.nostrオブジェクトを通じて公開鍵の取得 (getPublicKey())、イベントへの署名要求 (signEvent()) などを行うことができます[[34]](https://stacker.news/items/119948#:~:text=NIP,or%20by%20the%20Blockcore%20wallet)。これはブラウザ拡張（例: Albyやnos2x等）によって提供され、Webクライアントがユーザーの秘密鍵を直接扱わずに安全に署名操作を委託できる仕組みです[[34]](https://stacker.news/items/119948#:~:text=NIP,or%20by%20the%20Blockcore%20wallet)。このようにクライアント側ではNIP-07互換の署名インターフェースや、あるいはデスクトップ/モバイルアプリでの安全なキーストレージによって、鍵と署名の扱いが実装されています。

**リレーへの認証:** 基本的にオープンなNostrリレーでは上記の鍵署名機構のみでユーザー認証は十分ですが、特定のリレーでは**アクセス制限や利用者認証**を要求する場合があります。そのための拡張仕様として**NIP-42: クライアントのリレー認証**があります[[35]](https://nips.nostr.com/42#:~:text=New%20client)。NIP-42では、新たにAUTHというメッセージを定義し、クライアントとリレーがチャレンジ・レスポンス方式で認証を行います[[36]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form)。リレーが認証を必要とするとき、まずクライアントに対し["AUTH", <ランダムなチャレンジ文字列>]というメッセージを送ります[[36]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form)。クライアントは自分の秘密鍵でこのチャレンジを署名した**認証用イベント**（kind:22242の一時イベント）を作成し、["AUTH", <署名済みイベントJSON>]形式でリレーに返答します[[37]](https://nips.nostr.com/42#:~:text=When%20sent%20by%20relays%20the,message%20has%20the%20following%20form)[[38]](https://nips.nostr.com/42#:~:text=The%20signed%20event%20is%20an,Example)。このイベントには少なくともチャレンジ文字列（"challenge"タグ）とリレーURL（"relay"タグ）が含まれ、リレー側は署名とタグ内容を検証して正当なユーザーであることを確認します[[38]](https://nips.nostr.com/42#:~:text=The%20signed%20event%20is%20an,Example)[[39]](https://nips.nostr.com/42#:~:text=To%20verify%20,must%20ensure)。認証が成功すれば、リレーはOKメッセージで承認を返し、以後その接続については認証済みユーザーとして制限解除された操作（イベント投稿や購読）が可能となります[[40]](https://nips.nostr.com/42#:~:text=Clients%20MAY%20provide%20signed%20events,all%20pubkeys%20as%20authenticated%20accordingly)[[41]](https://nips.nostr.com/42#:~:text=%2A%20%60%22auth,or%20is%20exceeding%20its%20authorization)。例えば、有料リレーやプライベートリレーではこの仕組みにより**事前に登録・承認された公開鍵のみ**に利用を限定するといった運用がなされています。「auth-required: ...」や「restricted: ...」といったメッセージプレフィックスがOK/CLOSEDで返された場合、NIP-42認証が求められていることを意味します[[42]](https://nips.nostr.com/42#:~:text=%60OK%60%20and%20%60CLOSED%60%20machine)。

まとめると、Nostrの認証は主に**暗号学的な署名によるイベント単位の認証**であり、ユーザーは秘密鍵を使って自らの発言に責任を持つ形になっています。追加のリレー認証が必要な場合でも、公開鍵署名を用いたプロトコルレベルの方式で実現されており、ユーザー名・パスワード等の従来型認証は用いない点が特徴です。

## リレーサーバーとの接続と管理（WebSocketベース）

Nostrクライアントとリレーサーバーは**WebSocket**を介して通信します[[43]](https://nips.nostr.com/1#:~:text=Relays%20expose%20a%20websocket%20endpoint,of%20connections%20from%20specific%20IP%2Fclient%2Fetc)。リレーはWebSocketのエンドポイント（例: wss://relay.example.com)を公開し、クライアントはそこへ接続して双方向通信を行います[[43]](https://nips.nostr.com/1#:~:text=Relays%20expose%20a%20websocket%20endpoint,of%20connections%20from%20specific%20IP%2Fclient%2Fetc)。クライアントは**1つのリレーにつき1本のWebSocket接続**を開き、必要なすべての購読とイベント送信をその接続で実施することが推奨されています[[43]](https://nips.nostr.com/1#:~:text=Relays%20expose%20a%20websocket%20endpoint,of%20connections%20from%20specific%20IP%2Fclient%2Fetc)。リレー側では同一クライアントから多数の接続が張られるのを制限する場合もあります[[43]](https://nips.nostr.com/1#:~:text=Relays%20expose%20a%20websocket%20endpoint,of%20connections%20from%20specific%20IP%2Fclient%2Fetc)。

**接続確立とメッセージ交換:** 接続が確立すると、以降は前述した形式のJSONメッセージ（EVENT, REQ, CLOSEなど）をテキストフレームでやり取りします[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)。クライアントからリレーへ送信可能なメッセージはNIP-01で3種類定義されています[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns):

* ["EVENT", <イベント>] – イベントの送信（公開）。クライアントが新規イベントを投稿する際に使用します[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)。
* ["REQ", <購読ID>, <フィルタ>, ...] – イベント購読の開始要求。指定したIDとフィルタ条件で過去および将来のイベント送信をリクエストします[[12]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,used%20to%20stop%20previous%20subscriptions)。
* ["CLOSE", <購読ID>] – 購読の停止要求。先に発行した購読IDに対応する購読を終了させます[[12]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,used%20to%20stop%20previous%20subscriptions)。

一方、リレーからクライアントへ送られるメッセージは以下の5種類があります[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other):

* ["EVENT", <購読ID>, <イベント>] – REQに対するイベント配信。[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)
* ["OK", <イベントID>, <true/false>, <メッセージ>] – イベント受理/拒否通知（NIP-20）。投稿受信時の結果を伝えます[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)。
* ["EOSE", <購読ID>] – 保存済みイベント送信終了通知（NIP-15）[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)。
* ["CLOSED", <購読ID>, <メッセージ>] – 購読の強制終了通知。リレー側で購読を打ち切った際に送信[[16]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,or%20other%20things%20to%20clients)。
* ["NOTICE", <テキスト>] – 汎用のテキスト通知メッセージ。エラーや情報をユーザーに伝えるためのものですが、使用規則は特に定められていません[[44]](https://nips.nostr.com/1#:~:text=This%20NIP%20defines%20no%20rules,should%20be%20sent%20or%20treated)。

クライアントはこれらのメッセージを受信することで、リレー上のイベントストリームや自身の投稿結果、エラー通知などを把握します。プロトコル上、リレーはクライアントからのREQに対してのみイベントを送り（購読IDで対応付け）[[15]](https://nips.nostr.com/1#:~:text=,messages%20received%20from)、EVENT投稿に対してOKやNOTICEで応答します[[9]](https://nips.nostr.com/1#:~:text=,true%2C)。したがって、クライアントから明示的な要求を出さない限りリレーから勝手にイベントがプッシュされることはなく、すべて**クライアント主導**でデータフローを管理できます。

**クライアントとリレーの役割:** Nostrにおけるリレーサーバーは基本的に**シンプルなストア＆フォワード**の役割を担います。リレーは受け取ったイベントを保存し、購読しているクライアントに配信するだけで、イベント内容の検閲や改変は行いません（ただし各リレー運営者のポリシーで投稿を拒否したり、保存期間を制限したりする場合はあります）。一方クライアントは、複数のリレーからイベントを収集・統合してタイムラインを構築したり、自分の投稿を複数リレーへ一斉送信して冗長化するなど、**リレーを組み合わせて利用するロジック**を担います。例えばあるユーザーAの投稿を読む場合、クライアントはユーザーAが利用しているリレーすべてに購読要求を送り、その全結果をマージして重複除去する、といった処理を行います。こうした分散モデルの下で、リレーは信頼性のあるデータ配信に徹し、クライアントは複数ソースからのデータ統合と検証を行う構成です。

**リレー管理と拡張情報:** クライアントはユーザーやネットワーク状況に応じて適切なリレーに接続する必要があります。ユーザーがどのリレーを主に使用しているかといった情報は、プロトコル上メタデータとして共有することも可能です。例えば**NIP-02（Contact List）**では、ユーザーがフォローしている公開鍵（kind:3イベントのpタグ）とともに、その各ユーザーに対応する「推奨リレーURL」をリストに含める仕様になっており[[45]](https://nostr-nips.com/nip-02#:~:text=Each%20tag%20entry%20should%20contain,anything%20and%20should%20be%20ignored)、コンタクトリストを公開することで「自分や自分のフォロー先は主にこのリレーに投稿している」という情報を他クライアントと共有できます[[46]](https://nostr-nips.com/nip-02#:~:text=Relay%20sharing)。また**NIP-65（Relay List Metadata）**では、ユーザー自身が利用中のリレー一覧（種別10002のイベント）を公開し、読み書き可否フラグとともに周知する機能も定義されています[[47]](https://nostr-nips.com/nip-65#:~:text=Defines%20a%20replaceable%20event%20using,The%20event)。クライアントはこれらのデータを参照し、接続すべきリレーのリストアップや、不要なリレー接続の削減を行うことができます。さらに、リレー自体も**NIP-11**によってメタ情報（リレー名や説明、対応NIP、利用ポリシー等）をHTTPエンドポイントで提供する仕様があり、クライアントは必要に応じてリレーの/または/.well-known/nostr.jsonへ問い合わせることで情報を得られます。

最後に、WebSocket接続の維持や再接続の管理はクライアント側の実装に委ねられています。ネットワーク不調等で切断した場合は再接続し直し、必要なREQを再送して購読を復元する必要があります。Nostrプロトコル自体はステートレスでシンプルなため、再接続後にどのイベントを取得済みかといった管理もクライアント側で追跡します。適切なフィルタ（例: sinceに直前のタイムスタンプを指定）を用いることで再取得の重複を防ぐことができます。このようにしてクライアントは永続的にイベントストリームを受信し続け、リレーとの接続を管理します。

以上、Nostrのイベント送受信、フィルタ、署名認証、リレー接続について現行仕様をまとめました。詳細な仕様については各NIPドキュメント（例えば基本仕様は**NIP-01**[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)、連合IDとリレー共有は**NIP-02**[[45]](https://nostr-nips.com/nip-02#:~:text=Each%20tag%20entry%20should%20contain,anything%20and%20should%20be%20ignored)[[46]](https://nostr-nips.com/nip-02#:~:text=Relay%20sharing)、ブラウザ連携は**NIP-07**[[34]](https://stacker.news/items/119948#:~:text=NIP,or%20by%20the%20Blockcore%20wallet)、認証拡張は**NIP-42**[[48]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form)等）を参照してください。Nostrはシンプルな設計ながら、NIPによる拡張でソーシャル機能やセキュリティを強化しています。プロトコルの各所で公開鍵暗号を駆使することで、信頼できる分散型ソーシャルメディア基盤を実現している点が技術的な特徴です。

**Sources:**

* NIP-01: *Basic protocol flow description*[[1][2]](https://nips.nostr.com/1#:~:text=)[[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns)[[18]](https://nips.nostr.com/1#:~:text=,ids)[[19]](https://nips.nostr.com/1#:~:text=,to%20this%20to%20pass)[[32]](https://nips.nostr.com/1#:~:text=All%20conditions%20of%20a%20filter,conditions)[[7]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,26%20is%20less%20than%2030)他
* NIP-02: *Contact List and Petnames*[[45]](https://nostr-nips.com/nip-02#:~:text=Each%20tag%20entry%20should%20contain,anything%20and%20should%20be%20ignored)[[46]](https://nostr-nips.com/nip-02#:~:text=Relay%20sharing)
* NIP-07: *window.nostr capability for web browsers*[[34]](https://stacker.news/items/119948#:~:text=NIP,or%20by%20the%20Blockcore%20wallet)
* NIP-42: *Authentication of clients to relays*[[48]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form)[[38]](https://nips.nostr.com/42#:~:text=The%20signed%20event%20is%20an,Example)
* *Nostr Protocol NIPs (official repository)*[[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other)[[17]](https://nips.nostr.com/1#:~:text=,are)

[[1]](https://nips.nostr.com/1#:~:text=) [[2]](https://nips.nostr.com/1#:~:text=) [[3]](https://nips.nostr.com/1#:~:text=To%20obtain%20the%20,of%20the%20following%20structure) [[4]](https://nips.nostr.com/1#:~:text=) [[5]](https://nips.nostr.com/1#:~:text=Events%20and%20signatures) [[6]](https://nips.nostr.com/1#:~:text=Clients%20can%20send%203%20types,according%20to%20the%20following%20patterns) [[7]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,26%20is%20less%20than%2030) [[8]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,readable%20error%20messages%20or%20other) [[9]](https://nips.nostr.com/1#:~:text=,true%2C) [[10]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,off%20from%20the%20current%20time) [[11]](https://nips.nostr.com/1#:~:text=%60%3Csubscription_id%3E%60%20is%20an%20arbitrary%2C%20non,guaranteed%20to%20be%20globally%20unique) [[12]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,used%20to%20stop%20previous%20subscriptions) [[13]](https://nips.nostr.com/1#:~:text=%60,can%20have%20the%20following%20attributes) [[14]](https://nips.nostr.com/1#:~:text=Upon%20receiving%20a%20,created%2C%20replacing%20the%20old%20one) [[15]](https://nips.nostr.com/1#:~:text=,messages%20received%20from) [[16]](https://nips.nostr.com/1#:~:text=%2A%20%60%5B,or%20other%20things%20to%20clients) [[17]](https://nips.nostr.com/1#:~:text=,are) [[18]](https://nips.nostr.com/1#:~:text=,ids) [[19]](https://nips.nostr.com/1#:~:text=,to%20this%20to%20pass) [[20]](https://nips.nostr.com/1#:~:text=Filter%20attributes%20containing%20lists%20%28,least%20one%20item%20in%20common) [[21]](https://nips.nostr.com/1#:~:text=The%20%60ids%60%2C%20%60authors%60%2C%20%60,character%20lowercase%20hex%20values) [[22]](https://nips.nostr.com/1#:~:text=,ids) [[23]](https://nips.nostr.com/1#:~:text=,ids) [[24]](https://nips.nostr.com/1#:~:text=,kind%20numbers) [[25]](https://nips.nostr.com/1#:~:text=As%20a%20convention%2C%20all%20single,any%20given%20tag%20is%20indexed) [[26]](https://nips.nostr.com/1#:~:text=,bytes%20lowercase) [[27]](https://nips.nostr.com/1#:~:text=event%20ids%2C%20for%20,a%20list%20of%20pubkeys%2C%20etc) [[28]](https://nips.nostr.com/1#:~:text=lowercase%20hex%20values) [[29]](https://nips.nostr.com/1#:~:text=lowercase%20hex%20values) [[30]](https://nips.nostr.com/1#:~:text=,return%20in%20the%20initial%20query) [[31]](https://nips.nostr.com/1#:~:text=The%20,avoided%20to%20prevent%20overwhelming%20clients) [[32]](https://nips.nostr.com/1#:~:text=All%20conditions%20of%20a%20filter,conditions) [[33]](https://nips.nostr.com/1#:~:text=%22pubkey%22%3A%20%3C32,key%20of%20the%20event%20creator) [[43]](https://nips.nostr.com/1#:~:text=Relays%20expose%20a%20websocket%20endpoint,of%20connections%20from%20specific%20IP%2Fclient%2Fetc) [[44]](https://nips.nostr.com/1#:~:text=This%20NIP%20defines%20no%20rules,should%20be%20sent%20or%20treated) NIP1 - NIP-01 - Basic protocol flow description

<https://nips.nostr.com/1>

[[34]](https://stacker.news/items/119948#:~:text=NIP,or%20by%20the%20Blockcore%20wallet) ChatGPT summaries of all Nostr Improvement Proposals \ stacker news ~nostr

<https://stacker.news/items/119948>

[[35]](https://nips.nostr.com/42#:~:text=New%20client) [[36]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form) [[37]](https://nips.nostr.com/42#:~:text=When%20sent%20by%20relays%20the,message%20has%20the%20following%20form) [[38]](https://nips.nostr.com/42#:~:text=The%20signed%20event%20is%20an,Example) [[39]](https://nips.nostr.com/42#:~:text=To%20verify%20,must%20ensure) [[40]](https://nips.nostr.com/42#:~:text=Clients%20MAY%20provide%20signed%20events,all%20pubkeys%20as%20authenticated%20accordingly) [[41]](https://nips.nostr.com/42#:~:text=%2A%20%60%22auth,or%20is%20exceeding%20its%20authorization) [[42]](https://nips.nostr.com/42#:~:text=%60OK%60%20and%20%60CLOSED%60%20machine) [[48]](https://nips.nostr.com/42#:~:text=This%20NIP%20defines%20a%20new,message%20has%20the%20following%20form) NIP42 - NIP-42 - Authentication of clients to relays

<https://nips.nostr.com/42>

[[45]](https://nostr-nips.com/nip-02#:~:text=Each%20tag%20entry%20should%20contain,anything%20and%20should%20be%20ignored) [[46]](https://nostr-nips.com/nip-02#:~:text=Relay%20sharing) Nostr | NIP-02

<https://nostr-nips.com/nip-02>

[[47]](https://nostr-nips.com/nip-65#:~:text=Defines%20a%20replaceable%20event%20using,The%20event) Nostr | NIP-65

<https://nostr-nips.com/nip-65>