Bruce Beaulieu, Riley Daughton,

Matthew Gibbons, and Noah Holt

What are we doing?

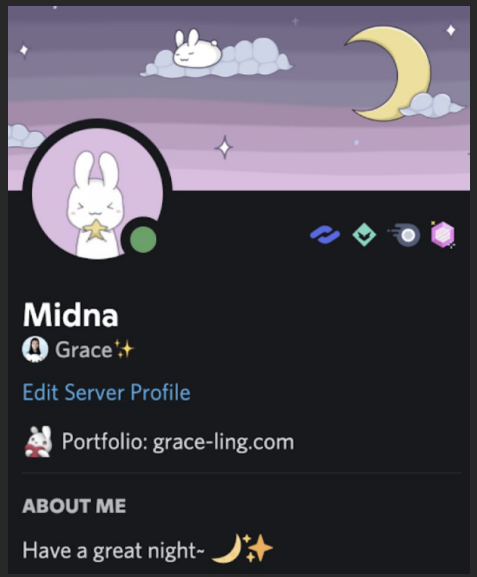
- We made a wyrm that functions as a honey pot.
  - The worm functionality is turned off for security and ethical reasons
- Created library wrapper to disguise bots to appear as users
  - Discord limits access of regular bots so we got around this security system in place
  - This inbuilt security disallowed our access to resources we wanted.
- Our bots then in servers, collect data on users in the shared servers.
  - Data such as username, photo, messages posted, connected accounts, screenshares and other activity and changes to user accounts.
- That data is passed to our database to be accessed through discord bot commands
  - We made several commands to look up specific users or search message logs or see changes made to an account.

Why are we doing it?

- Discord is easily becoming one of the largest social media/ messaging apps in the world, but its approach to user security is very disappointing, and is not as private as it leads on.

- A caution to be weary of what you decide to share on your profile, or say within public discord servers. It could be easily traced back to you as an individual.

- If Discord lacks basic security, imagine what they are collecting about all its users all the time, and what data could be collected by third at any point.



An example of a user profile from Discord Documentation

Is there a need for it?

1 0 1 0 1 0 1 1 1 0 0 0 1 1 1 0

- Security
  - Found flaws and loop arounds in discord API that we used to collect and see information
  - Accessing public endpoints for all information collected.
  - No data on discord is encrypted so easy to access other than their g-zip system on the socket
  - Many companies use their own API systems similar to this that could have similar vulnerabilities or workarounds to this.
  - Info mostly found on https://discord.com/developers/docs/intro
- Recent news
  - Spy.pet
    - Similar to ours except paying users to spread where we use bots
    - Guess we were 2 minutes to late.

1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 0

Company Implications

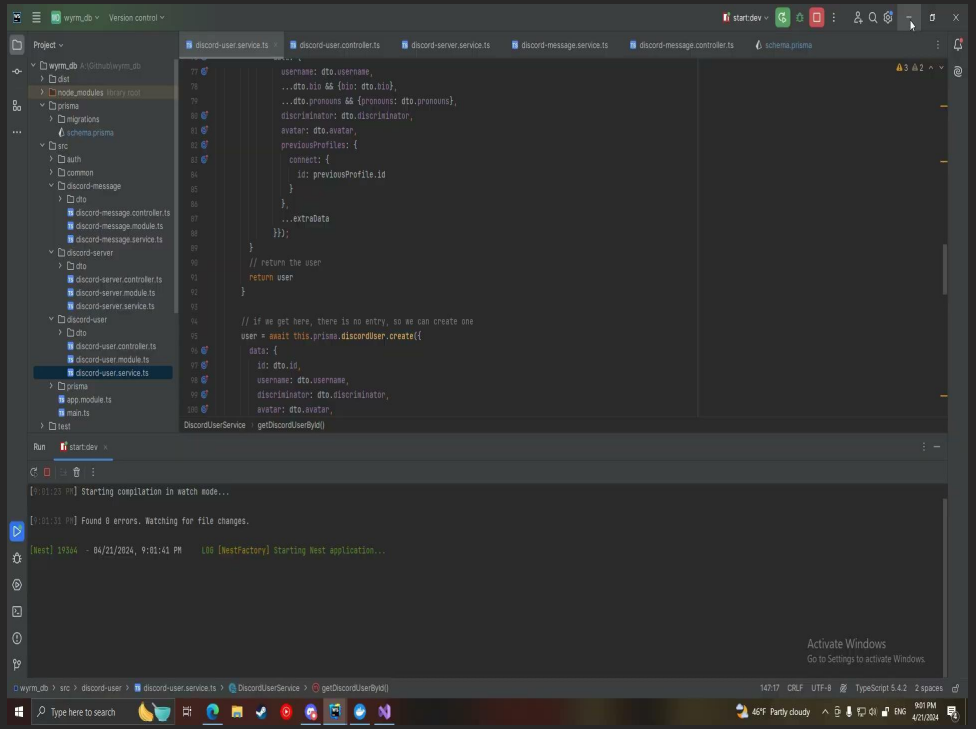
- Jason Citron, owner of Discord, has a history with selling user data previously. Before discord was OpenFeint. It was another social media gaming app made to connect players on mobile phone games, and was popular in early 2010's. Eventually he sold the company to GREE, and OpenFeint was hit with a lawsuit detailing that the company was gaining unauthorized access to users devices and collecting PI such as GPS data, Internet Browsing Data, and Social Media accounts.[1]

A Picture of Jason Citron. The Ceo of Discord

- Discord also has zero encryption when it comes to the messages sent between users, and its servers. It was stated earlier this year by Jason that this was done to easily work with law enforcement in cases of child endangerment.[2] But with his past, and with Tencent owning majority shares of Discord, I would not be surprised something is currently happening.

[1] Marshall, C. (2011, June 24). Gamers Say OpenFeint Sold Them Out. Courthouse News Service. Retrieved April 21, 2024, from https://www.courthousenews.com/gamers-say-openfeint-sold-them-out/

[2] Fung, B. (2024, January 31). Discord says it intentionally does not encrypt user messages. CNN. Retrieved April 22, 2024, from CNN


wyrm_db
Version control
start:dev
Project
discord-user.service.ts
discord-user.controller.ts
discord-server.service.ts
discord-message.service.ts
discord-message.controller.ts
schema.prisma
username: dto.username,
...dto.bio && {bio: dto.bio},
...dto.pronouns && {pronouns: dto.pronouns},
discriminator: dto.discriminator,
avatar: dto.avatar,
previousProfiles: {
connect: {
id: previousProfile.id
}
},
...extraData
});
}
// return the user
return user
}
// if we get here, there is no entry, so we can create one
user = await this.prisma.discordUser.create({
data: {
id: dto.id,
username: dto.username,
discriminator: dto.discriminator,
avatar: dto.avatar,
DiscordUserService
getDiscordUserById()
Run
start:dev
[9:01:23 PM] Starting compilation in watch mode...
[9:01:33 PM] Found 0 errors. Watching for file changes.
[Nest] 19304 - 04/21/2024, 9:01:41 PM LOG [NestFactory] Starting Nest application...
Activate Windows
Go to Settings to activate Windows.
Type here to search
46°F Partly cloudy
ENG
9:01 PM
4/21/2024

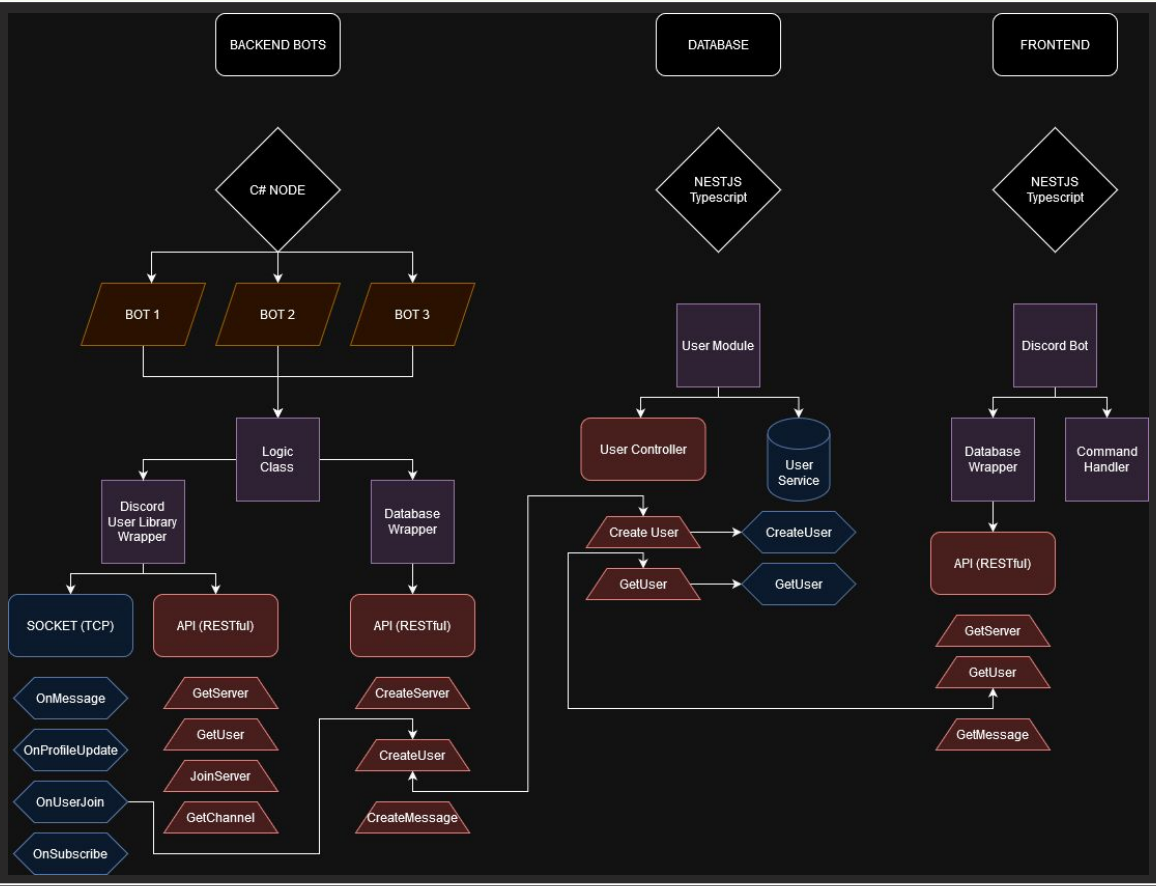
Tech Stack

shared database architecture

Backend
- C# multithreaded

Database
- Typescript
- Nest.Js
- Prisma ORM -> Postgresql
- Docker

Frontend
- Typescript
- Discord.js



Problem Solving and Bypassing

```
public string BuildXSuperProperties()
{
    var data = new
    {
        os = ToOs(),
        browser = ToBrowser(),
        device = "",
        system_locale = "en-US",
        browser_user_agent = ToString(),
        browser_version = ToVersion(),
        os_version = ToOsVersion(),
        referrer = "https://www.google.com/",
        referring_domain = "www.google.com",
        search_engine = "google",
        referrer_current = "",
        referring_domain_current = "",
        release_channel = "stable",
        client_build_number = 250306,
        client_event_source = null as Object
    };


    string propJson = System.Text.Json.JsonSerializer.Serialize(data, new JsonSerializerOptions { });
    propJson = Convert.ToBase64String(Encoding.UTF8.GetBytes(propJson));
    return propJson;
}
```

- Important Header to help hide the bot from discords detection and stopped flags
- Created from headers gathered from a get request to discord.com/register and the config the bot has for emulation of system.

Problem Solving and Bypassing Pt.2

1 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0

- User does not get the all same events as a discord bot
- They Mitigate events from servers that are important to our collection
- Sending a OP 37 event code with the server id we can subscribe to events that we normally should not see.

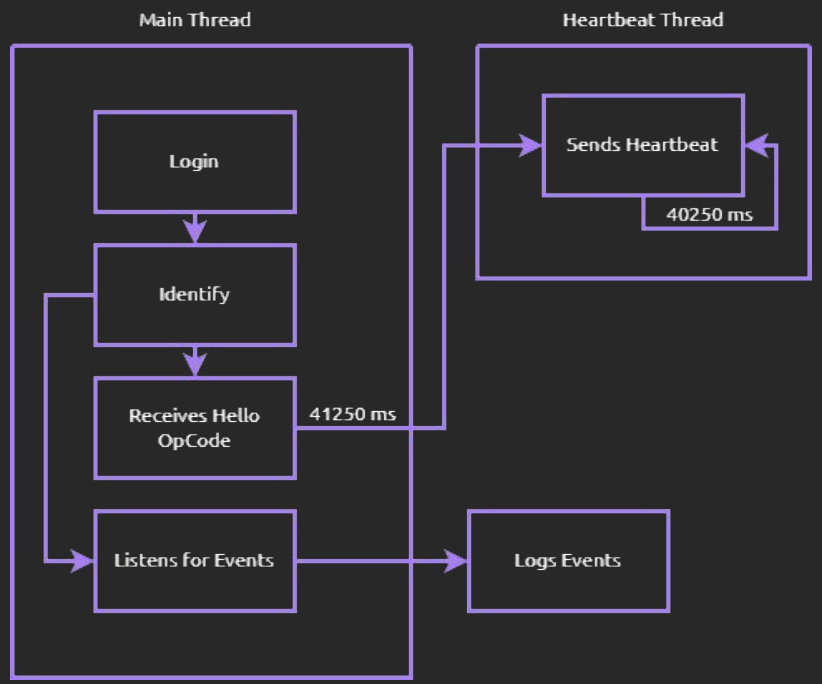
```
private async void ExpandEvents(string serverId, string channelId)
{
    var payload = new ArraySegment<byte>(
        Encoding.UTF8.GetBytes("{\"op\":37,\"d\":{\"subscriptions\":{\"" + serverId.ToString()
        + "\":{\"typing\":true,\"activities\":true,\"threads\":true,\"channels\":{\""
        + channelId.ToString() + "\":[[0,99]]}}}}}"));

    // send the OP payload
    await client.SendAsync(payload, WebSocketMessageType.Text, true, CancellationToken.None);

    var payload2 = new ArraySegment<byte>(
        Encoding.UTF8.GetBytes("{\"op\":36,\"d\":{\"guild_id\":\"" + serverId.ToString() + "\"}}"));
    // send the message
    await client.SendAsync(payload2, WebSocketMessageType.Text, true, CancellationToken.None);
}
```

1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 1

Discord Gateway Socket

1 0 1 0 1 0 1 1 1 0 0 0 0 1 1 1 0

- Login: Initializes the socket connection
- Identify: Gathers (spoofs) client data, such as BrowserUserAgent or Os and OsVersion
- Hello opcode contains heartbeat interval
- Heartbeat thread initializes, sends heartbeat response at interval ±1000 ms
- Listens for socket events
- Logs those events
- Contains ‘delegates’ (subscribable events)



1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 1 0

Discord Gateway Socket - Events

OpCode + **Event Name** + **Message**

Integer String JSON

```
case OPCODES.PRESENCE_UPDATE:
                    // handle presence update
                    PresenceUpdate presenceUpdateData = message.Data.ToObject<PresenceUpdate>();
                    Logger.LogDetails($"Since: {presenceUpdateData.Since}", "HandleMessage (OpCode -> PresenceUpdate)");
                    Logger.LogDetails($"Status: {presenceUpdateData.Status}", "HandleMessage (OpCode -> PresenceUpdate)");
                    Logger.LogDetails($"AFK: {presenceUpdateData.Afk}", "HandleMessage (OpCode -> PresenceUpdate)");
                    if (presenceUpdateData.Activities != null) {
                        foreach (Activity activity in presenceUpdateData.Activities) {
                            Logger.LogDetails($"Activity Name: {activity.Name}, Type: {activity.Type}", "HandleMessage (OpCode -> PresenceUpdate)");
                        }
                    }
                    break;
```

Discord Gateway Socket - OpCodes/Types

Bot Event OpCodes

DISPATCH, HEARTBEAT, HEARTBEAT_ACK, HELLO, IDENTIFY, INVALID_SESSION, PRESENCE_UPDATE, RECONNECT, REQUEST_GUILD_MEMBERS, RESUME, and VOICE_STATE_UPDATE

Bot Dispatch Types

APPLICATION_COMMAND_CREATE, APPLICATION_COMMAND_DELETE, APPLICATION_COMMAND_UPDATE, AUTO_MODEATION_ACTION_EXECUTION, AUTO_MODERATION_RULE_CREATE, AUTO_MODERATION_RULE_DELETE, AUTO_MODERATION_RULE_UPDATE, CHANNEL_CREATE, CHANNEL_DELETE, CHANNEL_PINS_ACK, CHANNEL_PINS_UPDATE, CHANNEL_RECIPIENT_ADD, CHANNEL_RECIPIENT_REMOVE, CHANNEL_UPDATE, ENTITLEMENT_CREATE, ENTITLEMENT_DELETE, ENTITLEMENT_UPDATE, GUILD_AUDIT_LOG_ENTRY_CREATE, GUILD_BAN_ADD, GUILD_BAN_REMOVE, GUILD_CREATE, GUILD_DELETE, GUILD_EMOJIS_UPDATE, GUILD_INTERACTIONS_UPDATE, GUILD_JOIN_REQUEST_DELETE, GUILD_MEMBER_ADD, GUILD_MEMBER_REMOVE, GUILD_MEMBER_LIST_UPDATE, GUILD_MEMBERS_CHUNK, GUILD_ROLE_CREATE, GUILD_ROLE_DELETE, GUILD_ROLE_UPDATE, GUILD_SCHEDULED_EVENT_CREATE, GUILD_SCHEDULED_EVENT_DELETE, GUILD_SCHEDULED_EVENT_UPDATE, GUILD_SCHEDULED_EVENT_USER_ADD, GUILD_SCHEDULED_EVENT_USER_REMOVE, GUILD_STICKETS_UPDATE, GUILD_SYNC, GUILD_UPDATE, INTEGRATION_CREATE, INTEGRATION_DELETE, INTEGRATION_UPDATE, INTERACTION_CREATE, INVITE_CREATE, INVITE_DELETE, MESSAGE_ACK, MESSAGE_CREATE, MESSAGE_DELETE, MESSAGE_DELETE_BULK, MESSAGE_REACTION_ADD, MESSAGE_REACTION_REMOVE, MESSAGE_REACTION_REMOVE_ALL, MESSAGE_REACTION_REMOVE_EMOJI, MESSAGE_UPDATE, PRESENCE_REPLACE, PRESENCE_UPDATE, READY, RESUMED, STAGE_INSTANCE_CREATE, STAGE_INSTANCE_DELETE, STAGE_INSTANCE_UPDATE, THREAD_CREATE, THREAD_DELETE, THREAD_LIST_SYNC, THREAD_MEMBER_UPDATE, THREAD_MEMBERS_UPDATE, THREAD_UPDATE, TYPING_START, USER_SETTINGS_UPDATE, USER_UPDATE, VOICE_CHANNEL_STATUS_UPDATE, VOICE_SERVER_UPDATE, VOICE_STATE_UPDATE, and WEBHOOKS_UPDATE

Database

- Structure of DB
  - Prisma ORM (Object relational mapping)
    - Copy down method
    - One-to-many or many-to-many relations
  - Each endpoint has one relations between front end and back end respectively

```
model DiscordUser {
  id                String                @id /
  username          String
  discriminator     String
  avatar            String?
  bio               String?
  pronouns          String?
  // ...
  // ...
  previousProfiles  DiscordUserSaves[]
  guildMember       DiscordGuildMember[]
  discordMessages   DiscordMessage[]     @rela
  mentionedIn       DiscordMessage[]     @rela
  connectedAccounts ConnectedAccount[] // conn
  stream            Stream[] // Streams the us
  activity          Activity[]

  @@map("DiscordUsers")
}
```

REST API

- We created a 3 tier model for our API request.
- The Three parts is The Class, The Base, and The Endpoint

- The class is where we define our response data, by giving types to each item in it

```
export interface User {
    id:              string;
    username:        string;
    discriminator:   string;
    avatar:          string;
    bio?: string;
    pronouns?: string;
    previousProfiles: PreviousProfile[];
    guildMember: GuildMember[];
    discordMessages: Message[];
    connectedAccounts: ConnectedAccounts[];
}
```

```
async GetRequest(endpoint: string, params: string) : Promise<ResponseData> {
        const response = await fetch(`${this.baseurl}${endpoint}/${params}`)
        let data = await response.json()
        return {data: data, code: response.status}
    }
```

- The Base is the general request that we can call. This is an example of a Get Request.

- The endpoint is the path in the url we follow, and give it the proper parameters to call for the data we want to receive. We put all the endpoint calls of a specific type in its own file for organization of possible calls, and

```
const baseurl = "http://localhost:3000"
const endpoint = "/user"
const backendClient = new request.BackendClient((baseurl))

//Returns user data for a specific userid
export async function GetUserData(params: string) {
    return await backendClient.GetRequest(endpoint, params)
}
```

Legal/ToS/Ethical Violations

General Data Protection Regulation (GDPR)

Article 6 of GDPR: Lawfulness of Processing
All personal data processing must occur under one of six lawful bases:
A. Consent
B. Contract
C. Legal Obligation
D. Vital Interests
E. Public Task
F. Legitimate Interests

Article 17 of GDPR: Right to Erasure ("Right to Be Forgotten")
People can request that data controllers erase their personal data under certain circumstances, including:
- If it's no longer needed for the reasons it was collect
- If they've withdrawn consent, and there's no other lawful basis for processing it
- If it's being processed unlawfully

[1] Bateman, R. (2023, July 01). Articles of the GDPR. Retrieved April 21, 2024, from https://www.termsfeed.com/blog/gdpr-articles/

Legal/ToS/Ethical Violations

Discord's Developer Terms of Service

Don't use the services to do harm to Discord. Among other things, this includes trying to gain access to or attacking our systems, scraping our services without our written consent, …

Don't use the services to do anything else that's illegal. This includes using the services to plan or commit any crime or do anything else that is illegal.

Ethical Implications

"The indiscriminate collection and monetization of personal data can lead to real harm, affecting people's social lives, personal relationships, and even their mental health." - Stack Diary

[1] Discord. (2024, April 15). Discord's Terms of Service. Retrieved April 21, 2024, from https://discord.com/terms
[2] Ivanovs, A. (2024, April 16). Spy.pet is harvesting your Discord history with no ability to opt-out. Retrieved April 21, 2024, from https://stackdiary.com/spy-pet-is-harvesting-your-discord-history-with-no-ability-to-opt-out/

Future Work

1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0

Database Optimizations

- Database should be able to handle a massive increase in users
- Set up Nodes to support multiple backend Wyrms to connect to the same database
- Store user tokens in database

Overall Security

- Authentication for front end access
- Authentication for backend endpoint access

Other Front Ends

- Web interface
- Realtime updates
- Authentication and security
- Alternative to discord bot commands

Refine Gateway Socket

- Add more supported events
- Add more tables to database

1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 1

Discord Gateway Socket - OpCodes/Types

1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0

Bot Event OpCodes

DISPATCH, HEARTBEAT, HEARTBEAT_ACK, HELLO, IDENTIFY, INVALID_SESSION, PRESENCE_UPDATE, RECONNECT, REQUEST_GUILD_MEMBERS, RESUME, and VOICE_STATE_UPDATE

Bot Dispatch Types

APPLICATION_COMMAND_CREATE, APPLICATION_COMMAND_DELETE, APPLICATION_COMMAND_UPDATE, AUTO_MODEATION_ACTION_EXECUTION, AUTO_MODERATION_RULE_CREATE, AUTO_MODERATION_RULE_DELETE, AUTO_MODERATION_RULE_UPDATE, CHANNEL_CREATE, CHANNEL_DELETE, CHANNEL_PINS_ACK, CHANNEL_PINS_UPDATE, CHANNEL_RECIPIENT_ADD, CHANNEL_RECIPIENT_REMOVE, CHANNEL_UPDATE, ENTITLEMENT_CREATE, ENTITLEMENT_DELETE, ENTITLEMENT_UPDATE, GUILD_AUDIT_LOG_ENTRY_CREATE, GUILD_BAN_ADD, GUILD_BAN_REMOVE, GUILD_CREATE, GUILD_DELETE, GUILD_EMOJIS_UPDATE, GUILD_INTERACTIONS_UPDATE, GUILD_JOIN_REQUEST_DELETE, GUILD_MEMBER_ADD, GUILD_MEMBER_REMOVE, GUILD_MEMBER_LIST_UPDATE, GUILD_MEMBERS_CHUNK, GUILD_ROLE_CREATE, GUILD_ROLE_DELETE, GUILD_ROLE_UPDATE, GUILD_SCHEDULED_EVENT_CREATE, GUILD_SCHEDULED_EVENT_DELETE, GUILD_SCHEDULED_EVENT_UPDATE, GUILD_SCHEDULED_EVENT_USER_ADD, GUILD_SCHEDULED_EVENT_USER_REMOVE, GUILD_STICKETS_UPDATE, GUILD_SYNC, GUILD_UPDATE, INTEGRATION_CREATE, INTEGRATION_DELETE, INTEGRATION_UPDATE, INTERACTION_CREATE, INVITE_CREATE, INVITE_DELETE, MESSAGE_ACK, MESSAGE_CREATE, MESSAGE_DELETE, MESSAGE_DELETE_BULK, MESSAGE_REACTION_ADD, MESSAGE_REACTION_REMOVE, MESSAGE_REACTION_REMOVE_ALL, MESSAGE_REACTION_REMOVE_EMOJI, MESSAGE_UPDATE, PRESENCE_REPLACE, PRESENCE_UPDATE, READY, RESUMED, STAGE_INSTANCE_CREATE, STAGE_INSTANCE_DELETE, STAGE_INSTANCE_UPDATE, THREAD_CREATE, THREAD_DELETE, THREAD_LIST_SYNC, THREAD_MEMBER_UPDATE, THREAD_MEMBERS_UPDATE, THREAD_UPDATE, TYPING_START, USER_SETTINGS_UPDATE, USER_UPDATE, VOICE_CHANNEL_STATUS_UPDATE, VOICE_SERVER_UPDATE, VOICE_STATE_UPDATE, and WEBHOOKS_UPDATE

1 0 1 0 0 1 0 1 1 0 1 1 1