Welcome to "Bringing players together: Building Cross Play in Destiny 2.

Today we're here to talk about what it took the Destiny 2 team to realize the cross-platform experience for players, the lessons we took away from the development process, and how they can be applicable to you if you're looking to bring cross-platform play to your game.

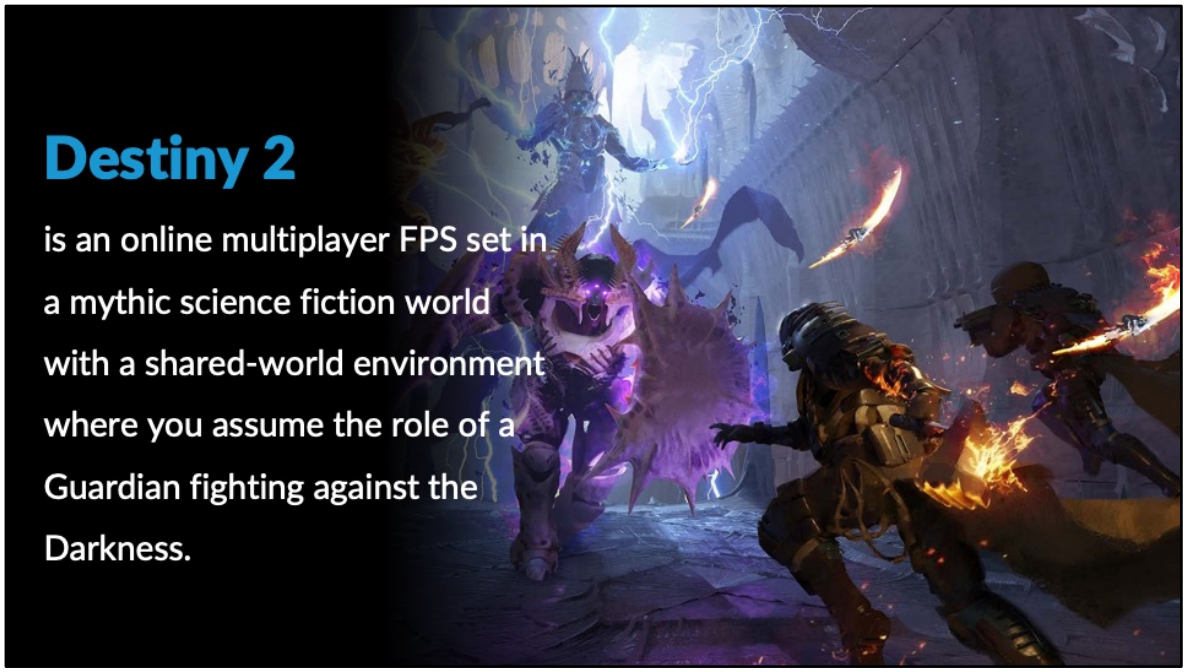
So let's dive into it!

A little about me. I'm Jon Chu, a Senior Technical Program Manager at Bungie in the Central Technology organization. As a Technical Program Manager, we function as the Producers for engineering-focused teams helping define roadmaps and product vision while driving the execution of our various projects. I've been at Bungie for 3 years now, having worked on various teams focused on building and improving our tools and engine, such as improving workflows for UI designers, updating the Destiny 2 engine for activities, and speeding release builds times. Currently, my focus is on leading initiatives that span across multiple teams around the studio, like most recently delivering Cross Play for Destiny 2.

That brings us to what we're talking about today! We'll be doing a deep dive on the production of Cross Play and the challenges and lessons we took away from building and releasing this feature to players.

I'll be focusing more on the production process, team organization, and some design considerations that come with building cross-platform. While I'll touch on some of the technical challenges that we faced, it'll be more of an overview. An explanation of the deep technical specifics is worthy of a whole other session.

We'll have time at the end for questions, and these slides with speaker notes will be uploaded to the Vault!



Quick show of hands, who here has played Destiny?

For anyone that hasn't played, Destiny 2 is an action MMO first-person shooter in which you play as a Guardian, imbued with the power of Light, protecting humanity. You take on challenges and activities all while looting and shooting your way to getting better gear and personalizing your character to be the ultimate monster-killing machine.

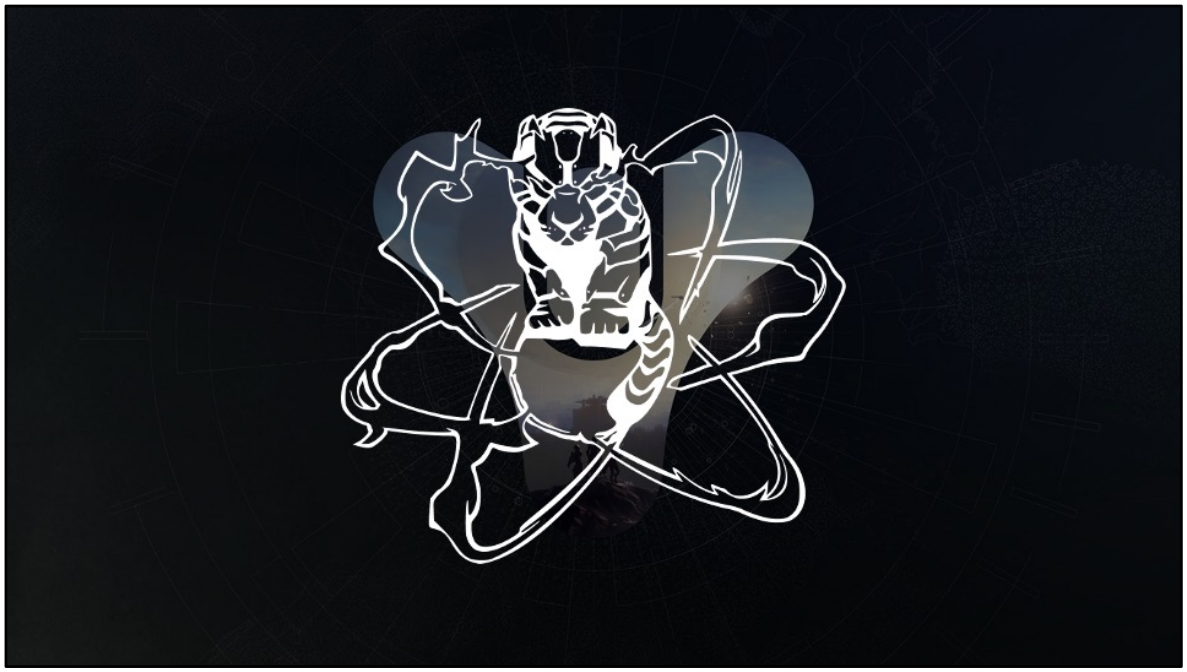
In terms of how we build, maintain and release Destiny 2,

We utilize a games-as-a-service model to constantly deliver updates and new content to players.

Our main content deliverables are yearly expansions and 4 seasonal releases during that year. Initially released in 2017, Destiny 2 is now in it's 5th year, and 16th Season of being a live game. Most recently in February, we released The Witch Queen along with Season of the Risen.

When it comes to platforms, Destiny 2 is currently available on 7 different platforms: PS4, PS5, Xbox One, Series One X and S, Stadia, Steam, and Microsoft PC Store.

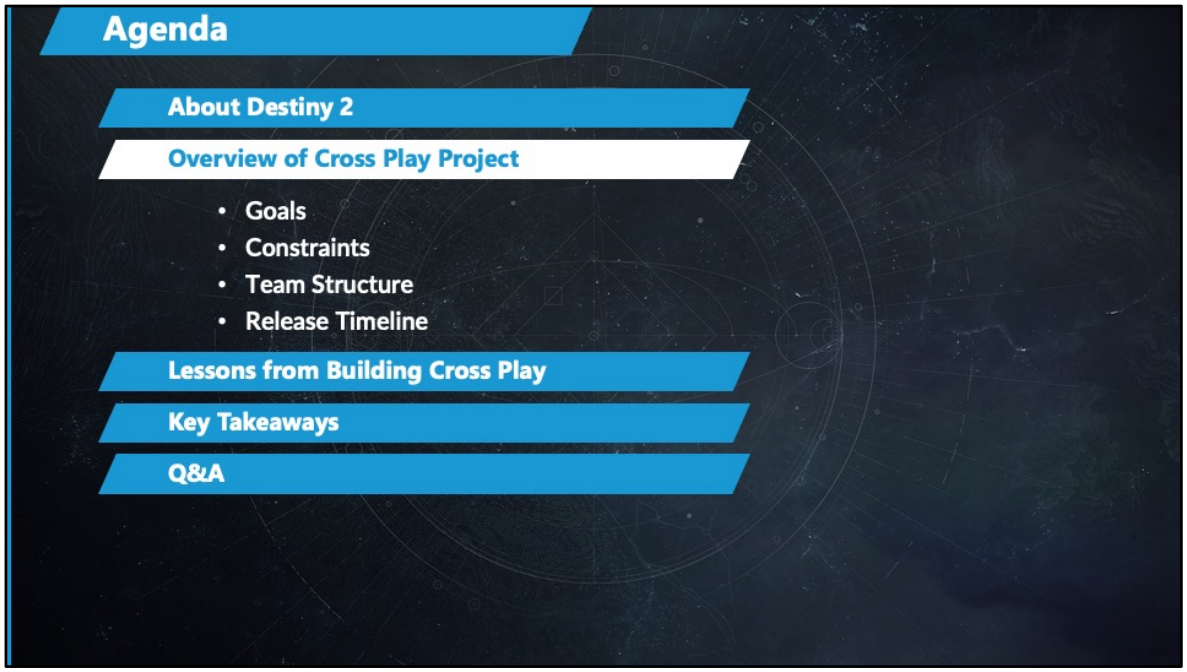
Lastly, Destiny 2 is built on our proprietary Tiger Engine.



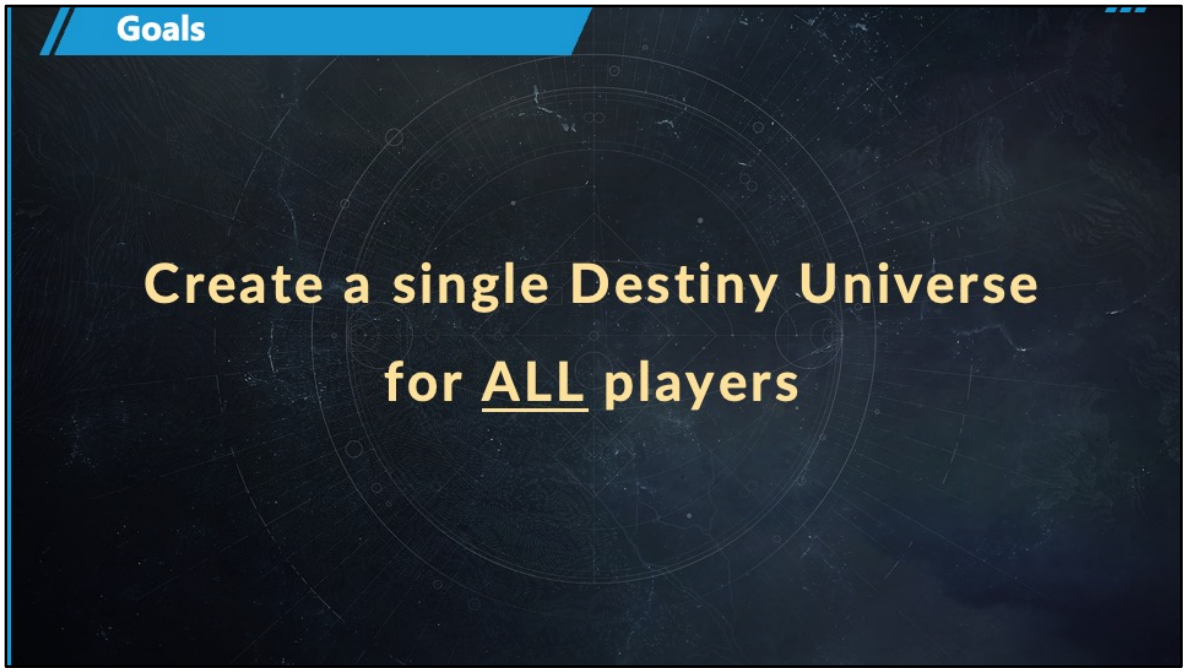
It’s a large legacy codebase built for the original Destiny, with some parts of the engine reaching back to the Halo days of Bungie.

There’s a great 2015 GDC talk by Chris Butcher that goes more in-depth about the history and development of the Tiger Engine if you're interested in learning more.

Needless to say, there are a lot of systems and code within our own engine that are foundational and hard to change, and we’ll discuss more to the interesting challenges this presented while building Cross Play, later in this talk.



Now that you know a little bit of background about Destiny 2, let's talk about the Cross Play project and how we set out to build this experience.



The biggest reason we wanted to build Cross Play was to deliver on creating a truly unified Destiny Universe.

By releasing across many platforms, players can play Destiny 2 from wherever they wanted, even other devices supported by Stadia.

We further built towards this goal with

our cross-progression system, Cross Save released in 2020. This allowed players to bring their Guardians and all the loot and gear they spent so much time amassing while playing on one platform to any platform they wanted to.

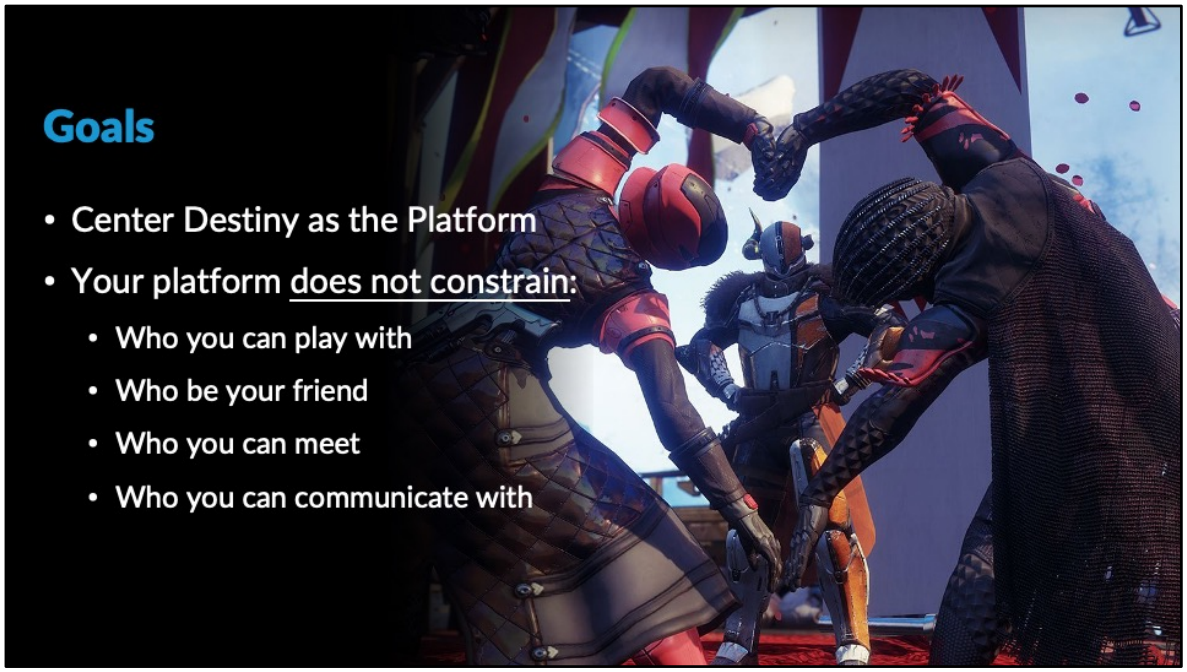
While with Cross Save, you could now hop between your PC and console and continue the story of your Guardian wherever you wanted, the one thing that was missing was that when players crossed platform boundaries, that meant leaving behind the friends you had on one platform.

People have different preferences for the platform they like to play Destiny 2 with. Just to demonstrate, of all the people here who do play Destiny 2, how many of you prefer playing on PC? … PlayStation? … Xbox? … Stadia?

I for one like playing on Xbox, and for the longest time, I didn't have access to a PC to play with friends who preferred Steam.

So that's where Cross Play comes in. Cross Play is the culmination of all these earlier efforts and now allowing you to play together with your friends from whatever platform you want. Now all players could exist in a single Destiny Universe, rather than segmented ones by platform.

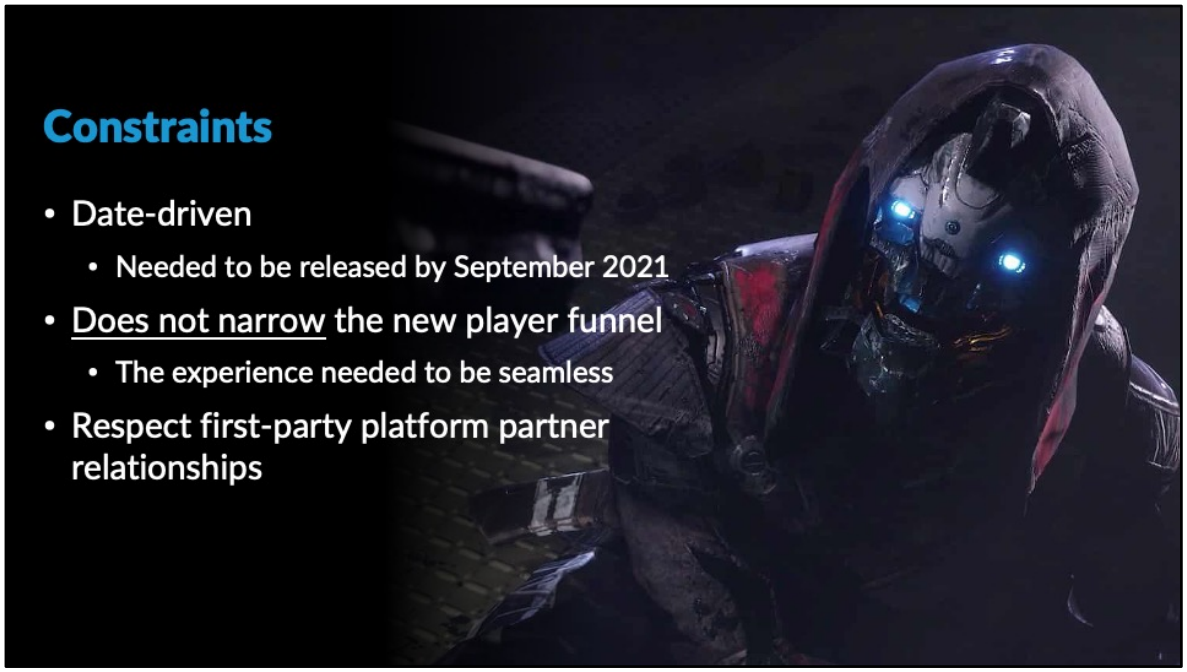
While this effort wasn't creating new content like a raid or secret exotic mission for players, we knew that doing this was the right thing to do and would be a delighter for them. So for this reason, we committed to making Cross Play happen.



When we first started pre-production of Cross Play, guided by our overarching vision of creating a single Destiny universe, we outlined some high-level goals for the project.

While Destiny is being released on multiple platforms, we wanted to use Cross Play as a chance to center Destiny as the platform for all your social interactions with players you meet while in-game.

Regardless of where you are playing from, you should not be limited in your Destiny experience, and everyone on any other platform is accessible to you. To be friends with, to communicate with, and more.



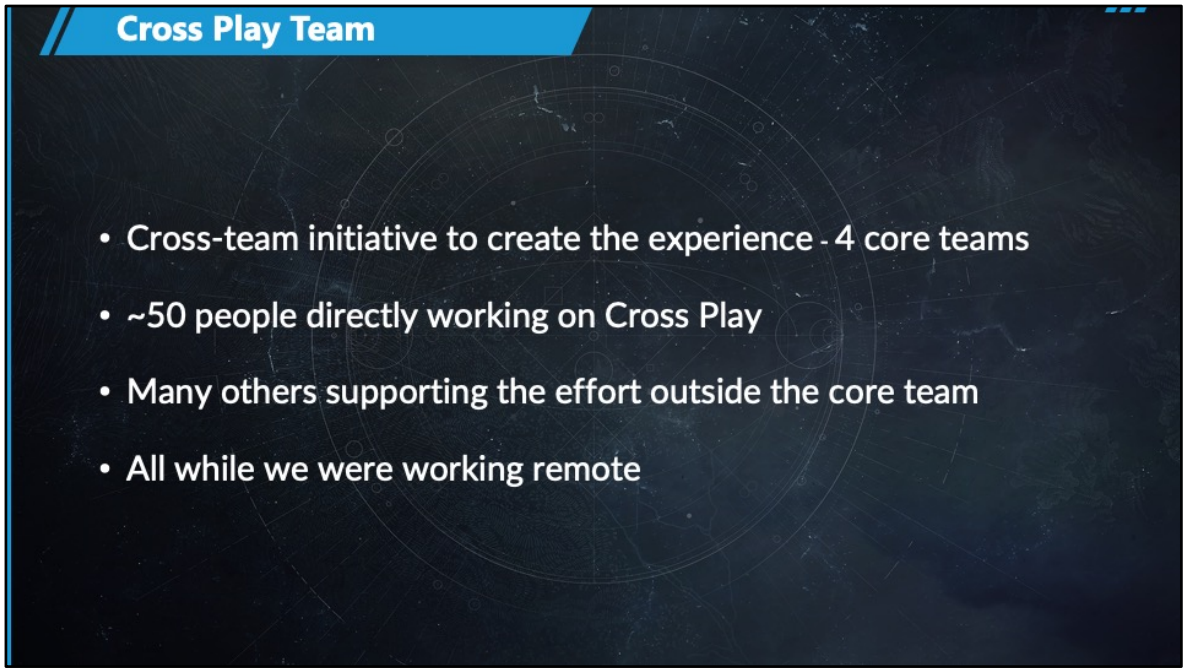
While we defined goals for the experience, we also made sure to articulate constraints that we could use as razors to help make decisions along the way.

First was that we were date-driven. We wanted to release Cross Play as soon as possible for the best player experience. So we targeted a release of September 2021. The pre-production planning for the project started in Summer 2020. This gave us a little over a year to plan, execute, and release Cross Play so we had to scope accordingly to meet this deadline.

Next, the experience needed to be seamless. Especially to new players. We did not want to over complicate the experience of getting a new player through character creation to having their boots on the ground in the first mission.

Lastly, knowing we would have to collaborate a lot with first-party platforms, like

Sony, Valve, Google, and Microsoft, we knew it was important to respect our partner relationships with each of the these companies. Each platform has their own unique cert requirements, and while building Cross Play, we should be doing our best to adhere to the spirit of each cert requirement.



Let's talk about the actual team that built Cross Play.
This effort was a cross-team initiative that spanned across the studio with 4 core teams working together to create the experience.
The core teams that were involved were :

- The Engine Client Team working on the lower-level engine code making changes to things like matchmaking and networking
- The UI-UX Team working on the UI players would be interacting with to manage all their new cross-platform friends and invites
- The Services Team tasked with standing up the many new game services needed
- And the Bungie.net Team that worked on the web and mobile experiences for Bungie.net and the Destiny 2 Companion App

In total there were about 50 people directly working on Cross Play.

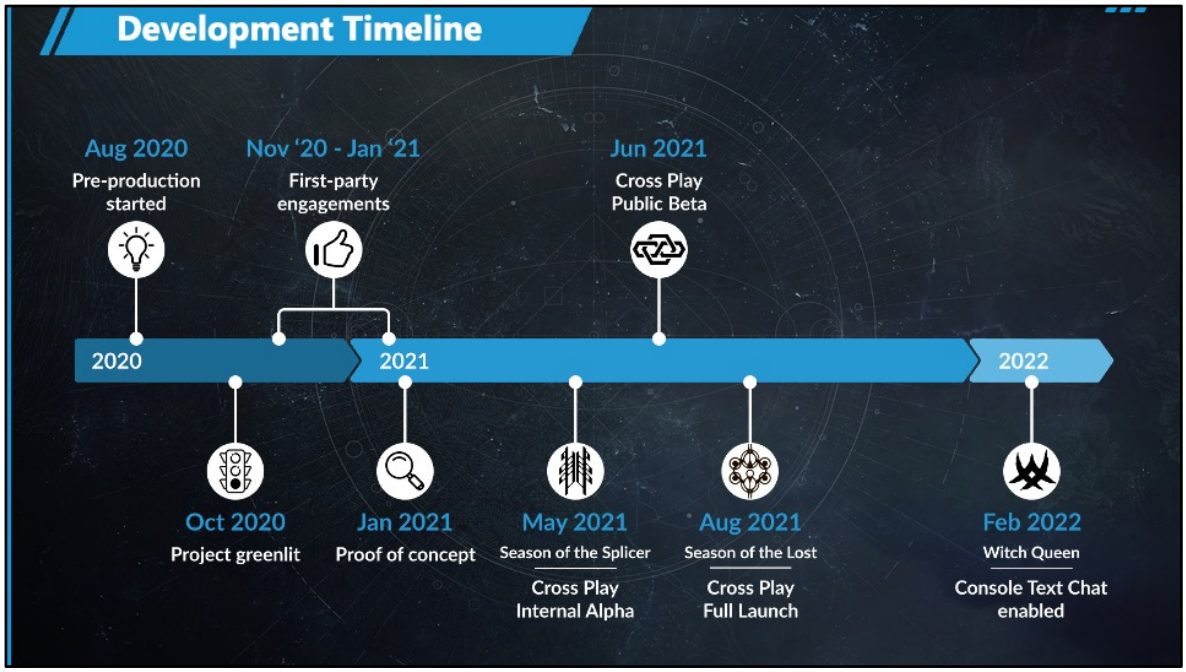
- This includes Engineers, Testers, Designers, Producers
- We also had our Technical Account Management deeply integrated on the team to handle first-party partner relationships

There were many other people throughout the studio supporting the effort outside the core team.

- Like the Publishing, Marketing, and Analytics teams helping create ways to get

players to engage with Cross Play
- The Player Support and Community teams being in tune with players feedback
- Also the Site Reliability and Network Operations teams making sure the Services we stood up kept running once we were live

And lastly this was all done while working remote!



In terms of the development timeline of Cross Play:

As mentioned before, we started preproduction planning during the summer of 2020.

From then, it was about a year of development to fully release Cross Play with the Season of the Lost in August 2021.

Along the way, there were a couple milestones and events that helped us along. We spent the first couple months working on major tech pieces of the experience while meeting with first-party partners to get approvals for our designs. That all led to us having our Proof of Concept by the start of 2021 in which we had two platforms joining together for the first time!

We also utilized live updates to the game to get early versions of Cross Play out to players.

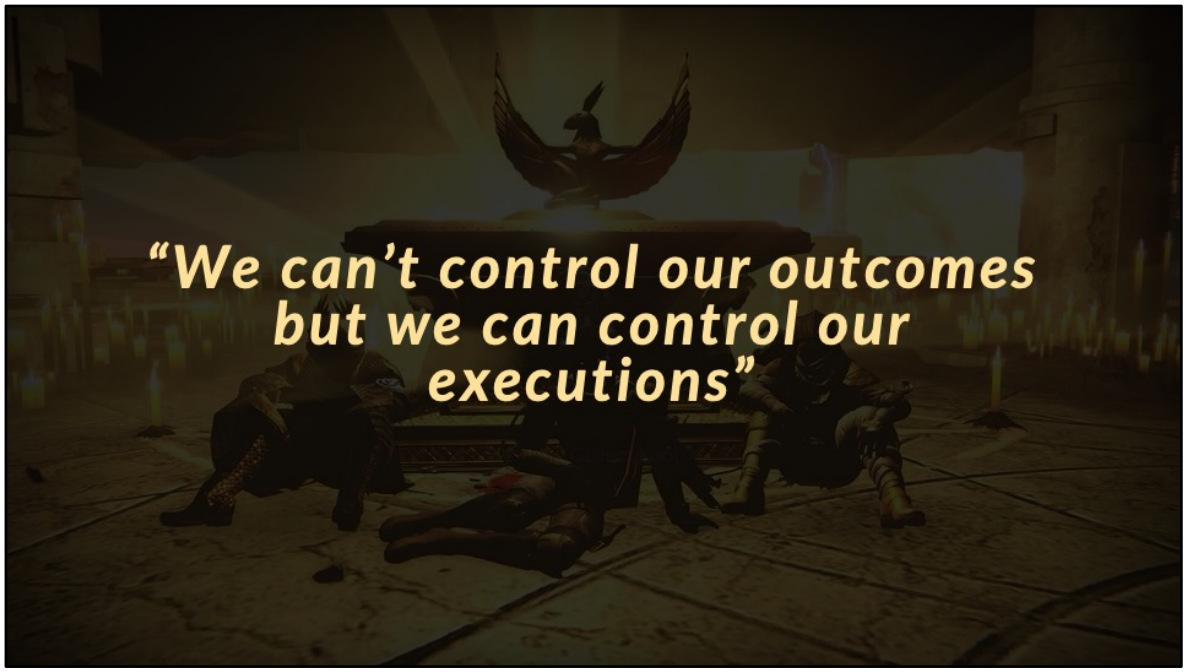
In season before we fully launched, we released an Alpha of Cross Play for Bungie employees to have some Cross Play features enabled for them, and later a Beta which we enabled Cross Play in a strike playlist for all players to try out.

These early releases helped us get Cross Play out there and incrementally build on the

experience. We'll talk a bit more about that later in the talk.

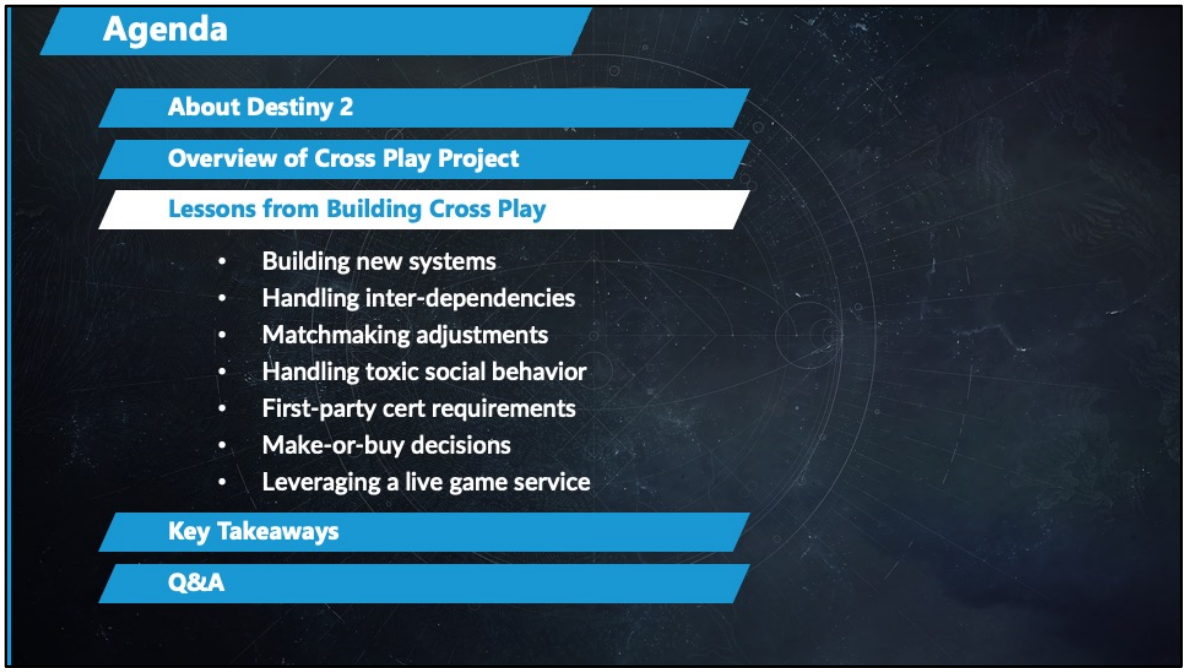
Even after the full release we've been continuing to add more features to Cross Play with things like extended text chat on consoles.

It was definitely a journey for the team to get Cross Play into players hands!



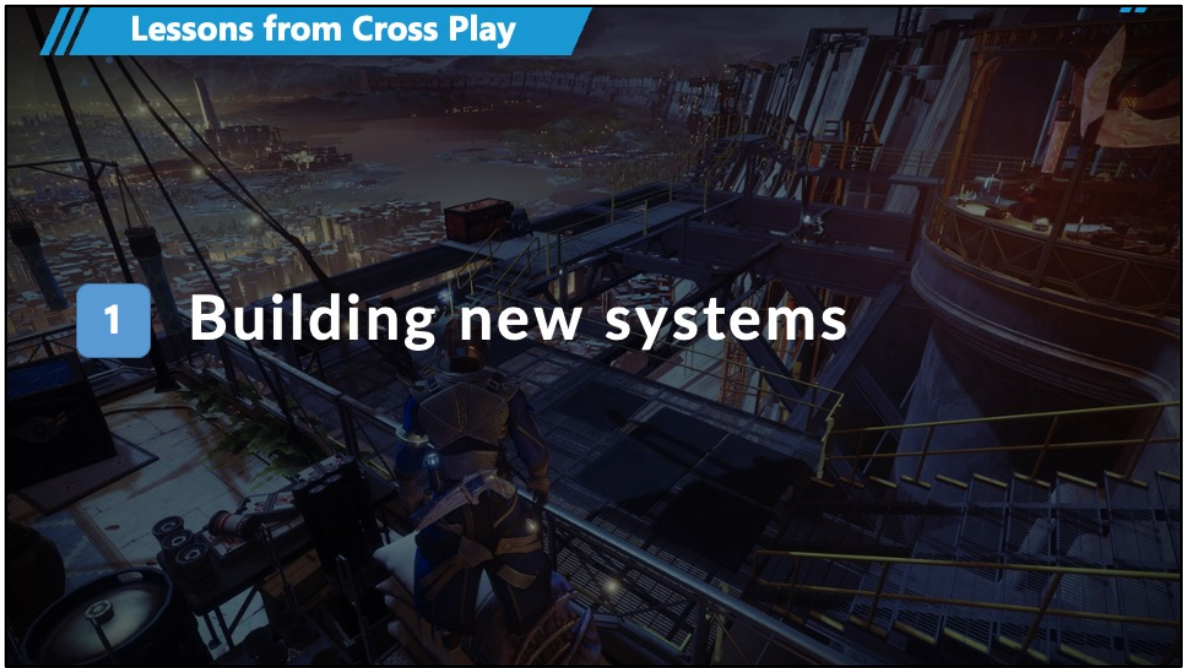
From the timeline, you can see we used many different milestones and releases to launch Cross Play. And one of my favorite quotes about the process we used comes from within the studio - ”We can’t control our outcomes, but we can control our executions”. We tried our very best to execute in the best way possible, to make the right decisions for the team, the experience, and the player. And that’s the main focus of what we wanted to share in this talk. In the end, that’s really all that you really can focus and work on because once it’s in the players’ hands you have no control over that.

But luckily, Cross Play was released to a lot of positive attention from our players, uniting them all together and allow them to play and engage with each other in a new way.

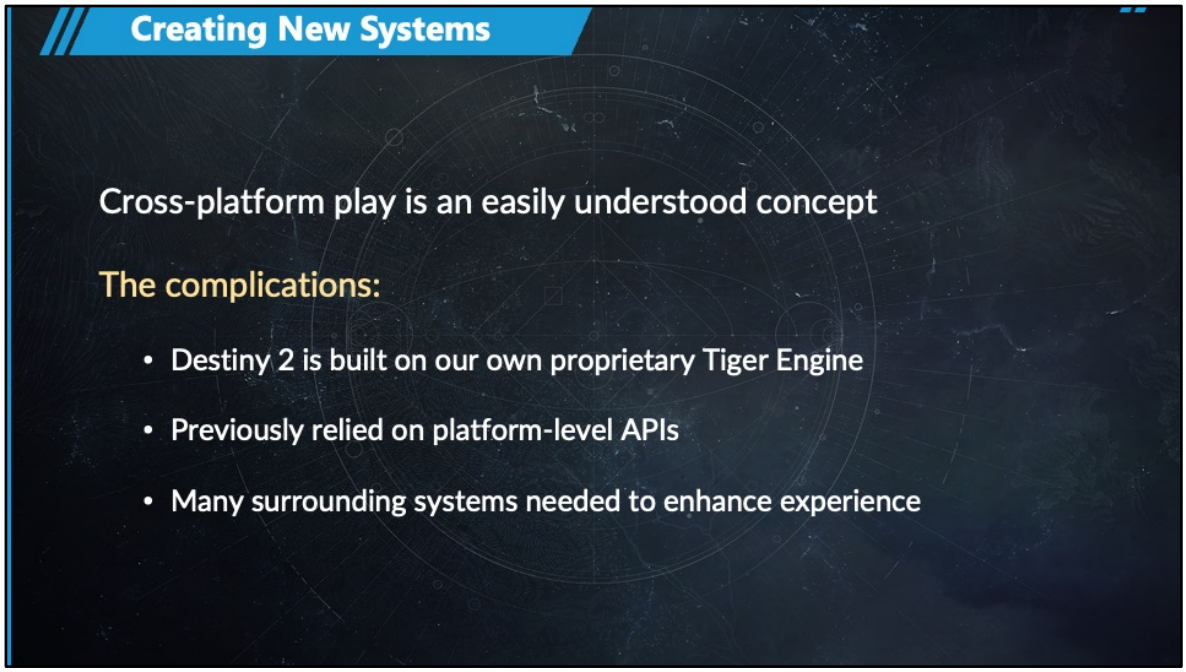


Now that we talked about the motivations and circumstances surrounding making Cross Play, we'll move onto the actual lessons we learned. At Bungie, we like the number 7, so this next section comes in that many parts.

Here's the 7 Lessons we learned in building Cross Play!



First up, building Cross Play in our own engine meant creating a lot of new systems from scratch.



Cross-platform play is an easily understood concept. It's allowing everyone on the various platforms you support to play together.

It's just networking around bits to all the different platforms. Right?

I mean sure, that's definitely how you get everyone in world together, but to really create a great Cross Play experience there's so much more to it.

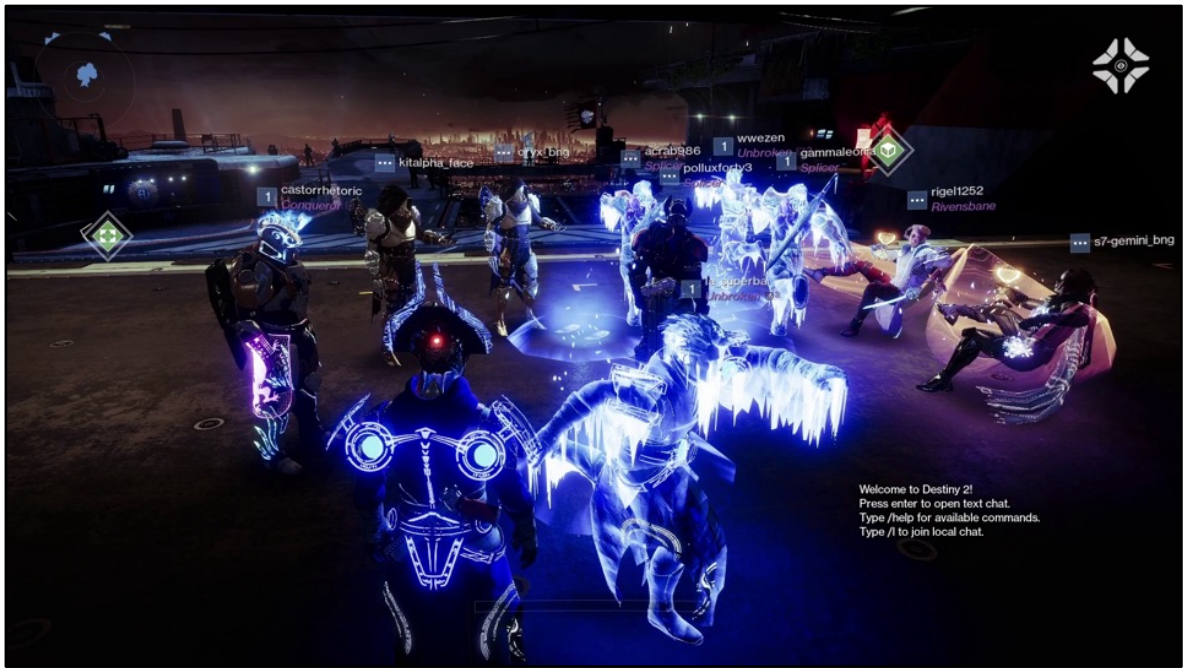
There's a couple of reasons that caused us to need to build so many new systems.

- Firstly, it's that we build Destiny in our own proprietary engine. All the new cross-plat functionality for the engine needed to be created from scratch.
- Next, many systems that we were using, were based on platform-level APIs, and all of these had to be recreated in-engine to support cross-platform. Unlike a game that uses a 3rd party engine like Unreal or Unity which has a lot of cross-plat built in, we'd had to build a lot of platform equivalent systems and services.
- Also, many surrounding systems are needed to enhance Cross Play like how the

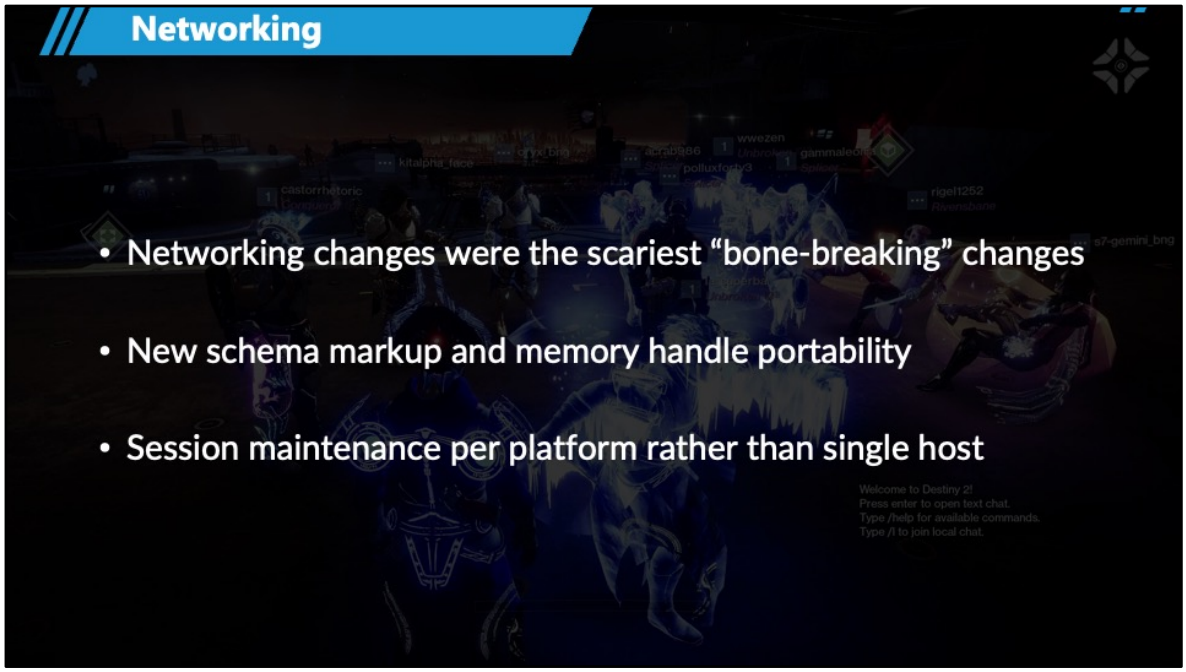
player now identifies themselves across various platforms.

So let’s do a run down of all the various systems made for Cross Play!

Starting with Networking.



Just to get all these Guardians into the tower together, core networking tech had to be changed to be enable sending of content across various platforms.



The networking changes we made were the biggest and scariest of “bone-breaking” changes since it is one of the most fundamental and underlying systems in the Destiny 2 engine.

Since we had shipped separately for the various platforms, we made optimizations for each, and these didn’t all play nicely when combining them for Cross Play.

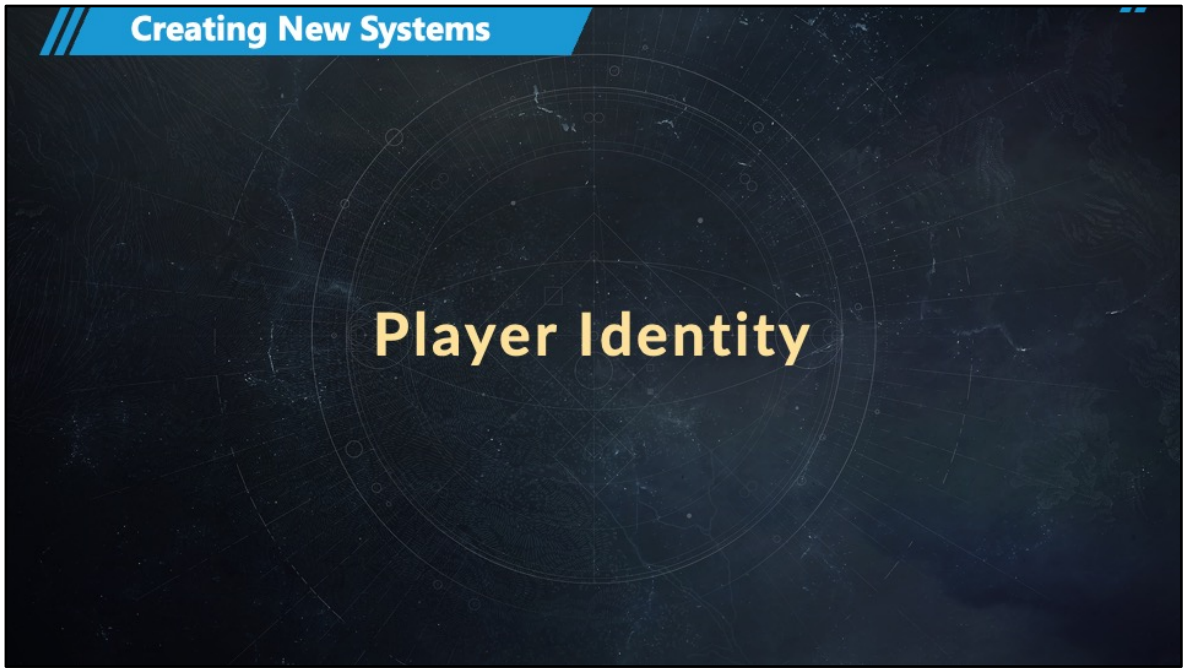
For example, our package layout is not always the same across platforms, so we can't just network memory handles from one platform to another. We ended up creating a mapping layer to communicate over the network with consistent handles that could then be mapped to the appropriate handles for a given platform

Also, the way we handle online sessions for fireteams also had to be changed since we no longer could we rely on a single session hosted by a single player. We now had to maintain a separate online sessions for each platform.

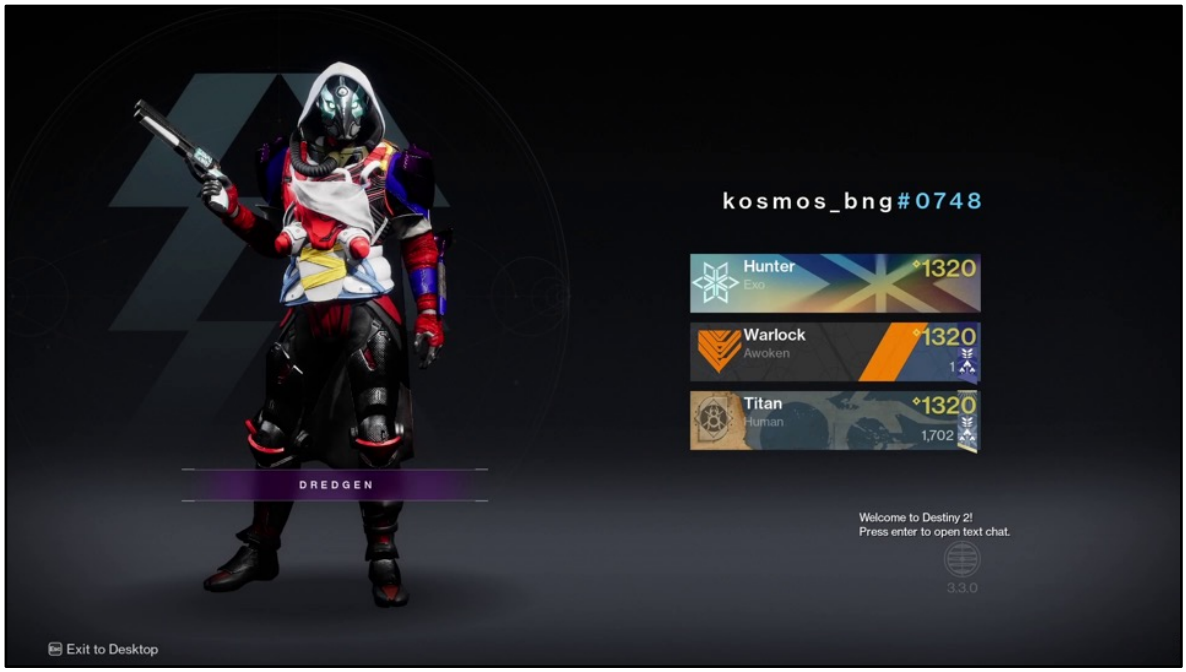
If you are interested in learning more about Destiny 2 networking there is a 2015 GDC talk by Justin Truman for you to watch.

There’s a lot more the technical specifics of what we did with networking. But

needless to say, these were big changes we needed to get different platforms talking to each other and were the earliest pieces we took on.



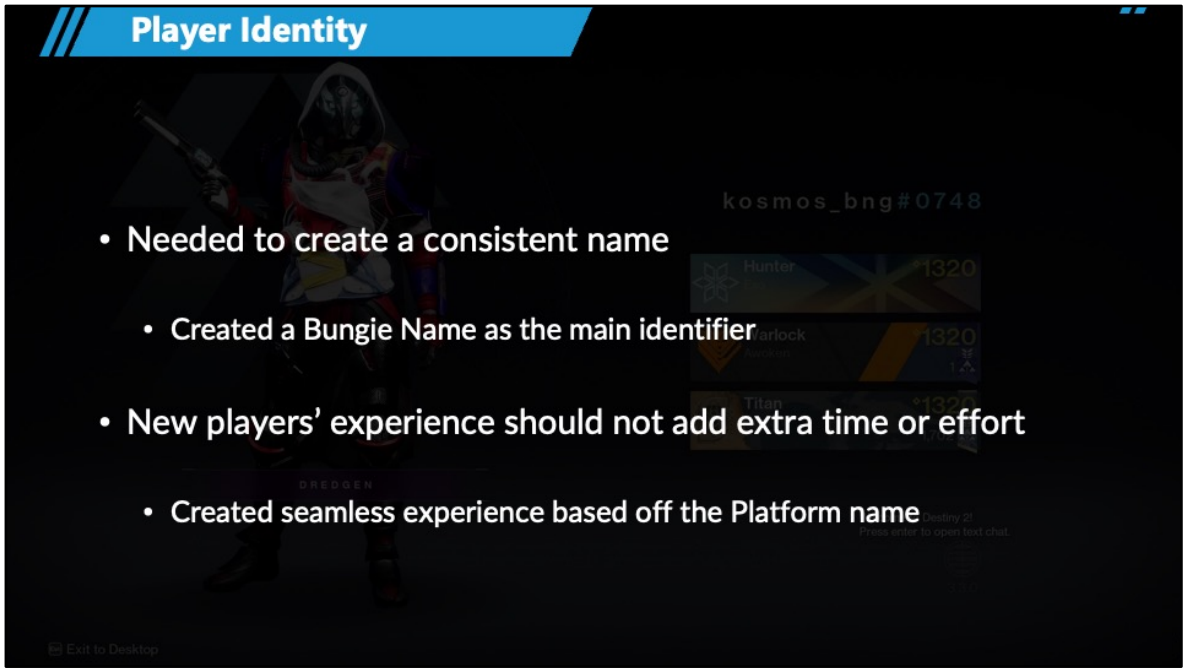
Next, let’s talk about Player Identity in a Cross Play world.



Players spend many hours playing Destiny 2, grinding through raids and dungeons, facing off in competitive matches, and more, trying to get the god roll of their favorite shotgun, or getting the right mods set up for their armor. The Guardian you play as, is an extension of yourself. And while what weapons you equip or getting your armor to look just right with shaders is important. Equally as important is your name, and how you identify yourself to other players.

Usually, you use your platform name like gamertag or Steam name (among other names for that) to identify yourself and by extension, your character in online games. However, with Cross Play, that's no longer sufficient. Say on PlayStation, you go by Oryx. One day, you hop over to PC and play on Steam, and there your account is Savathun none of your friends would recognize you by that name. Using platform names was no longer sufficient.

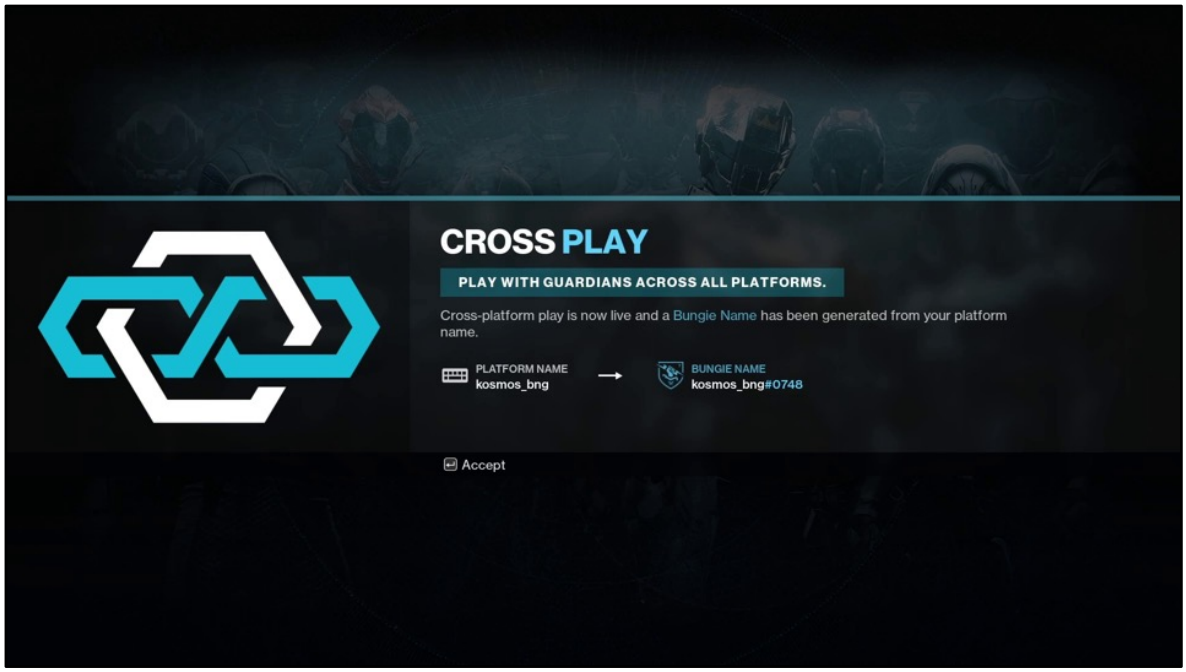
With Cross Play it was important that we deliver a consistent experience where someone is easily identified no matter what platform they are playing.



We needed to create a consistent name that identifies a player that is also unique so other players could find them.

To that end, we created the concept of a Bungie Name for a player. This consists of Player Name and 4-digit Unique Identifier. For example, Oryx#1234. Doing this allows multiple players to have the same name (in this case Oryx) but also allows people to search for a specific person with the 4-digits.

Guided by our earlier stated goals of the project of not narrowing the new player funnel. For new players logging on once Cross Play was enabled, the experience of setting a Bungie name should not add extra time or effort. We created a seamless experience that seeds the Player Name of a Bungie Name from the Platform name that a player first logged into Cross Play with.

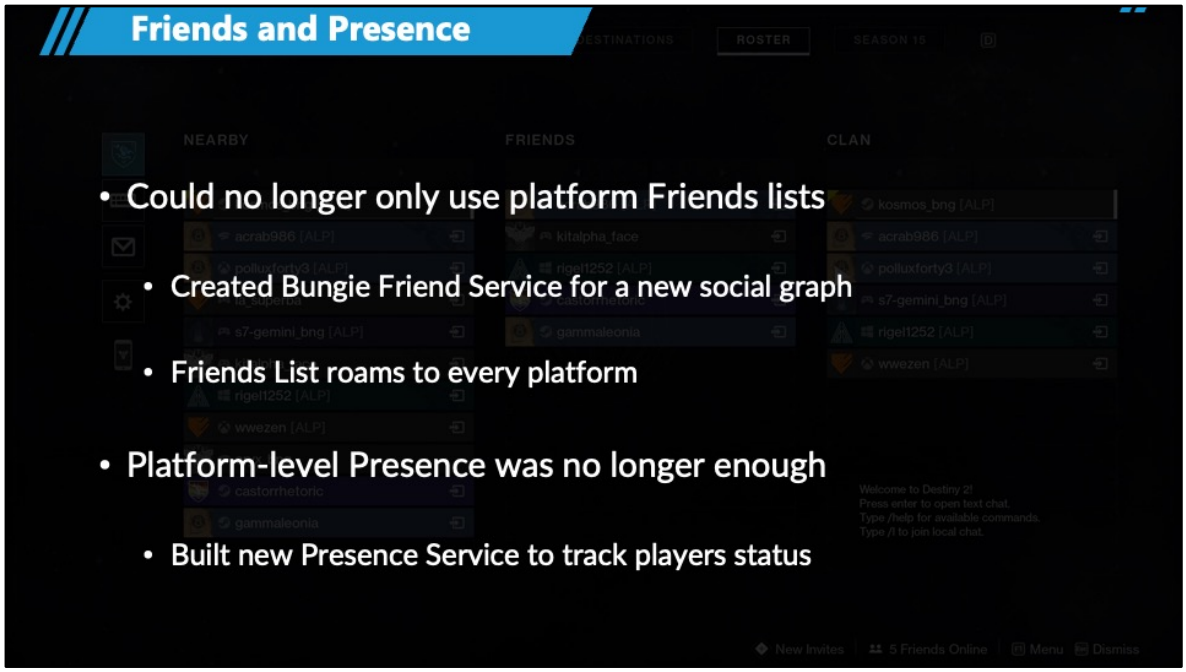


This is the screen players were met with the first time they logged in after Cross Play was enabled. It tells the player that their Bungie name was set from their platform name (in this case they were signing in for the first time using Steam and the profile name kosmos_bng. That seeds into the Bungie Name, while also the unique identifier of 0748 was appended to the end.

The player name of kosmos_bng is now what is displayed above the players head in game, and in the roster, wherever they play. It's their consistent identity.

Next is Friends and Presence. The concept of Friends and who the people you can play with was completely changed.

Now, your roster could have people from any and all the platforms Destiny is available on, and that brought with it many new challenges.

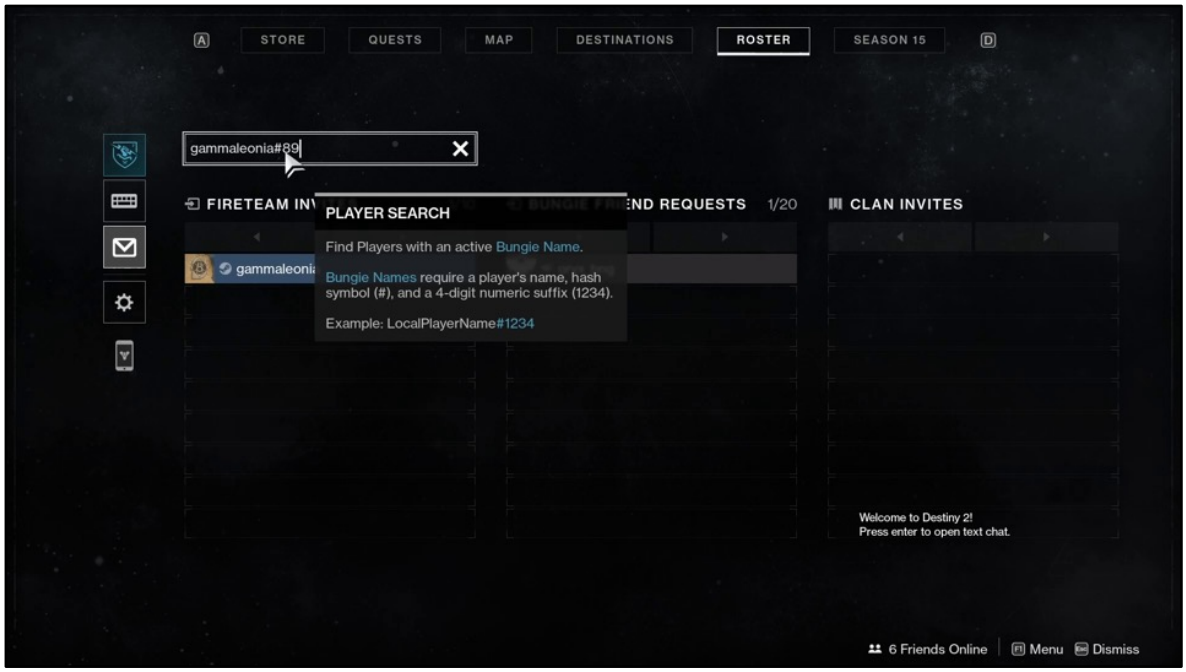


Destiny 2 is an MMO with many communities of players that are fostered by deep social mechanics. Most activities are meant to be played with others. Even if you play solo, we match you up with other players like with strikes or free roam around destinations. Destiny 2 is meant to be played with others.

Now that players could interact with anyone, that meant we could no longer just use platform Friends lists. Without changes, you would not have access to friends you made on other platforms. Since you could make and have friends anywhere now, we were exposing players to a whole new social network of Destiny players.

We created the Bungie Friend Service to handle a new social graph for players to engage with and enable them to become Friends with anyone. When a player “friends” someone in game, whatever platform they log in with, they will appear on the Bungie Friends List.

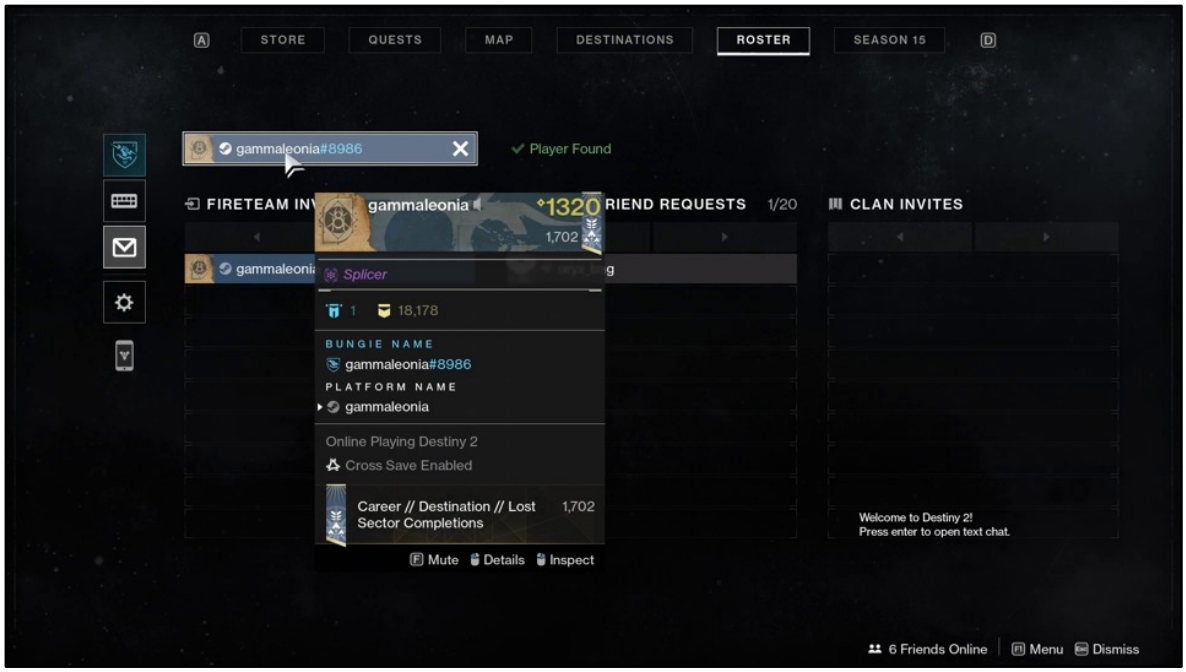
Additionally, platform-level presence was no longer enough to keep track of when someone was playing Destiny. So we stood up a new Presence Service to track players online and offline status wherever they were playing



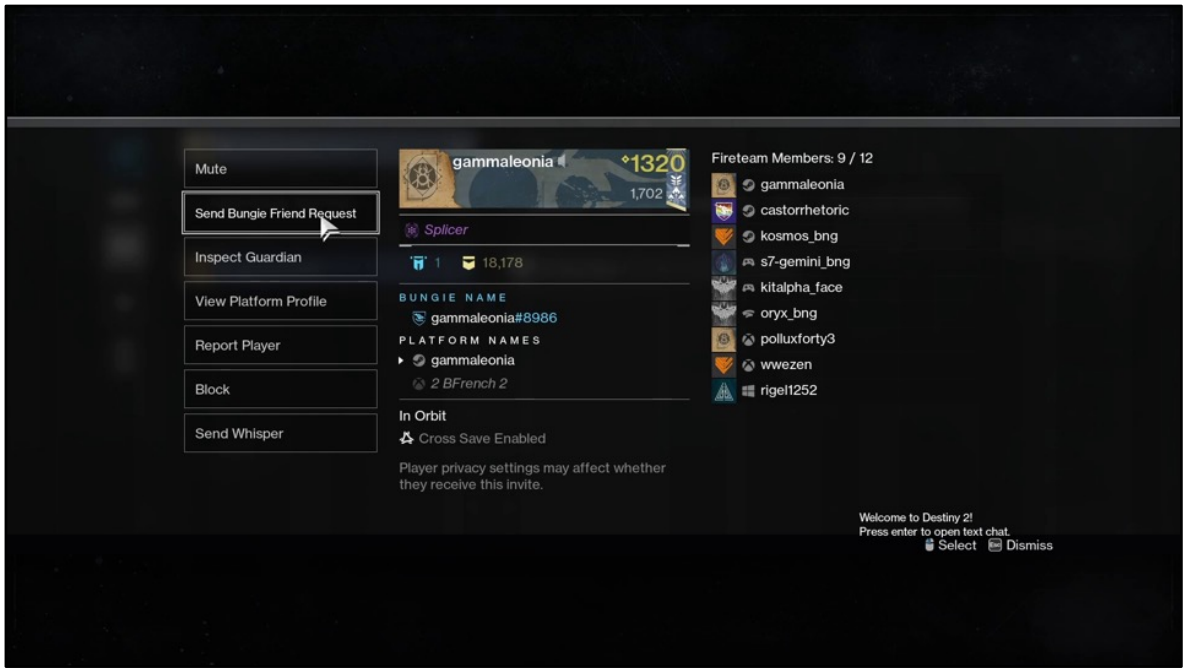
With this concept of a new Bungie Social Network, we needed to make sure we had all the interactions people have come to expect with social networks was also present.

First off is being able to find people. To that end we had to build searching. We kept it to finding others by their Bungie Name.

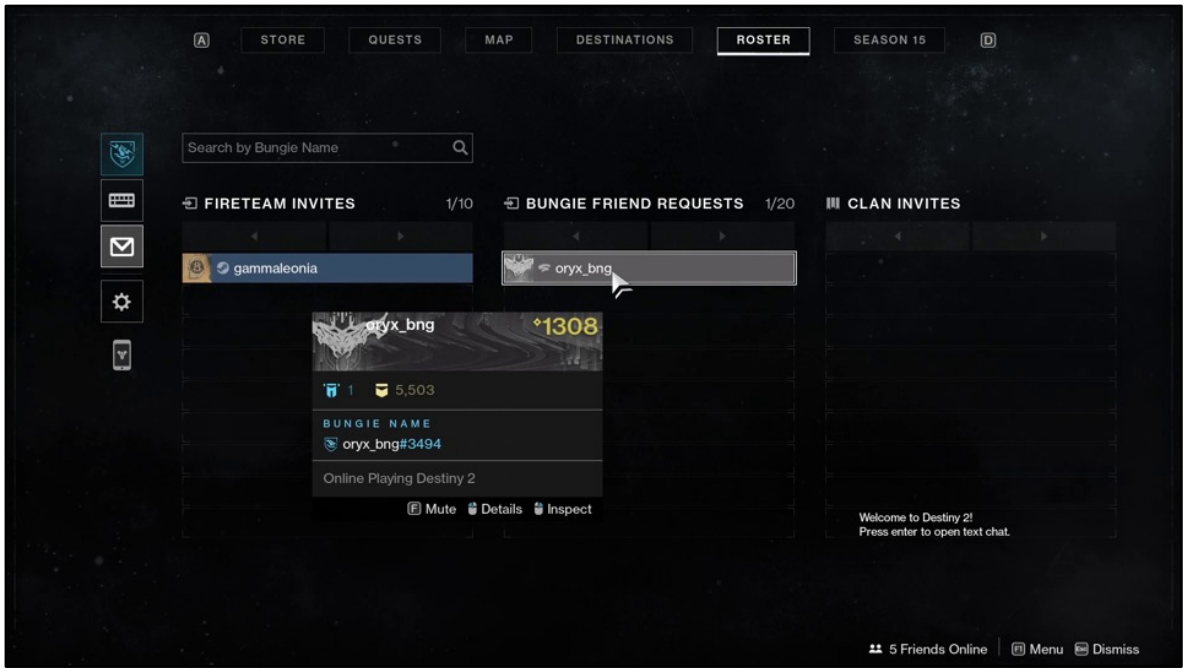
This is where the 4-digit unique identifier at the end of the name is so important to ensure people are explicitly searching for the right person.



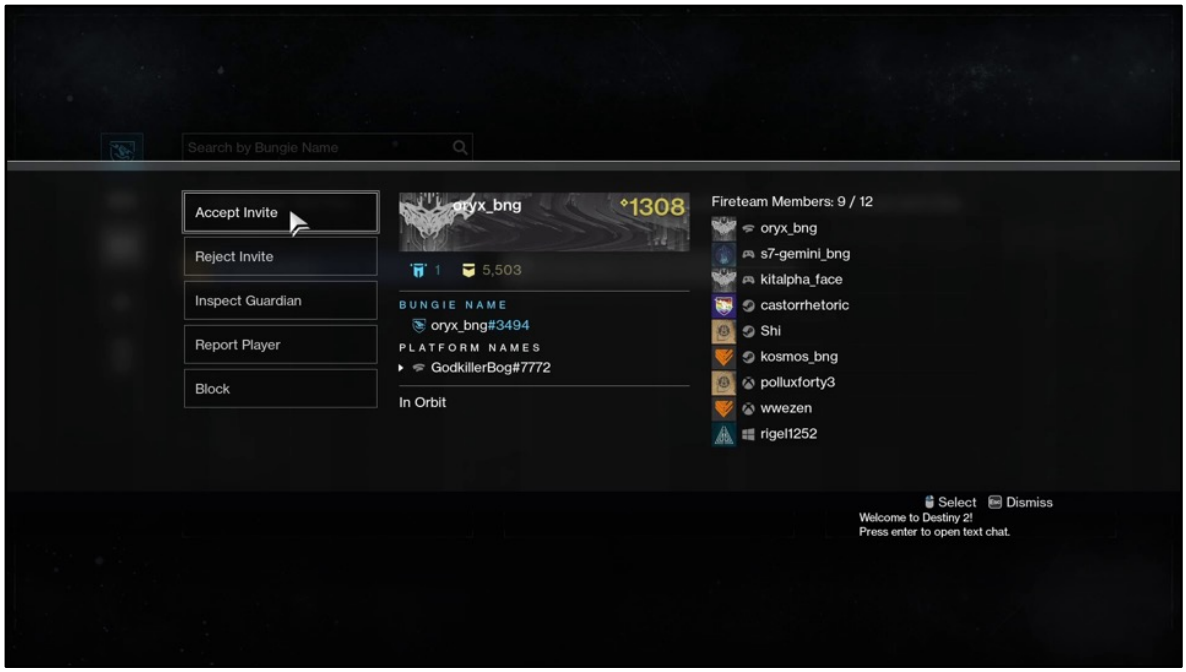
Upon finding that person you are able to inspect them



and send a Friend Request. .



On the receiving side, you need to be able to view incoming Friend Requests or see your pending ones.

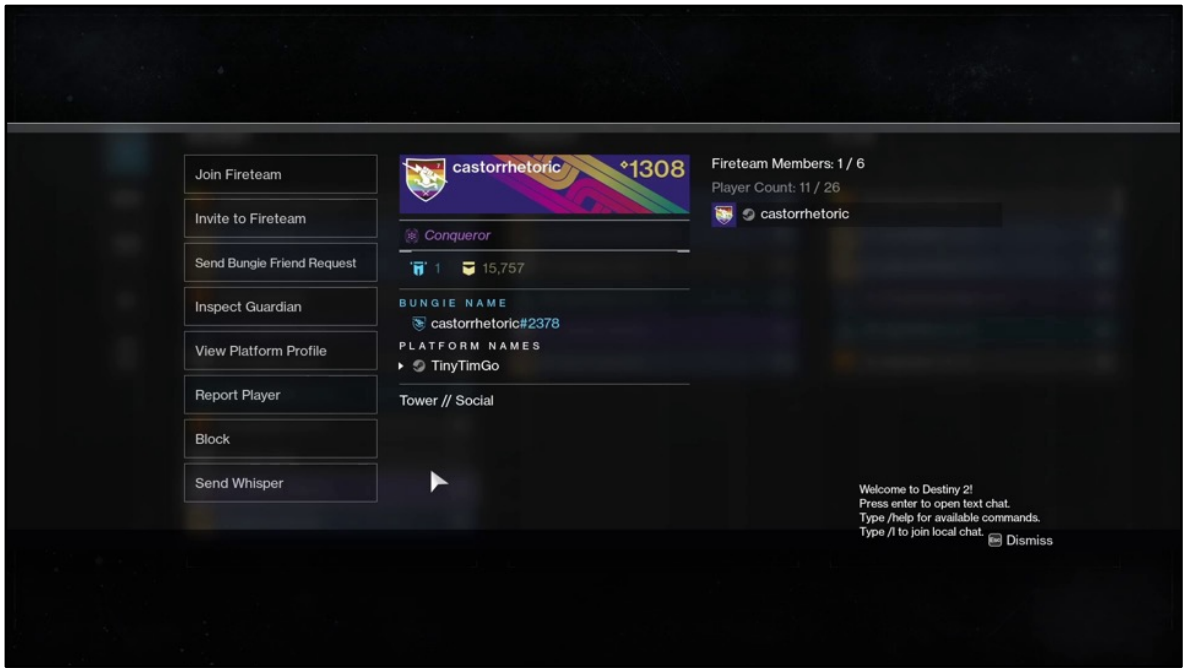


And finally be able to action on them by Accepting or Rejecting requests that come your way.

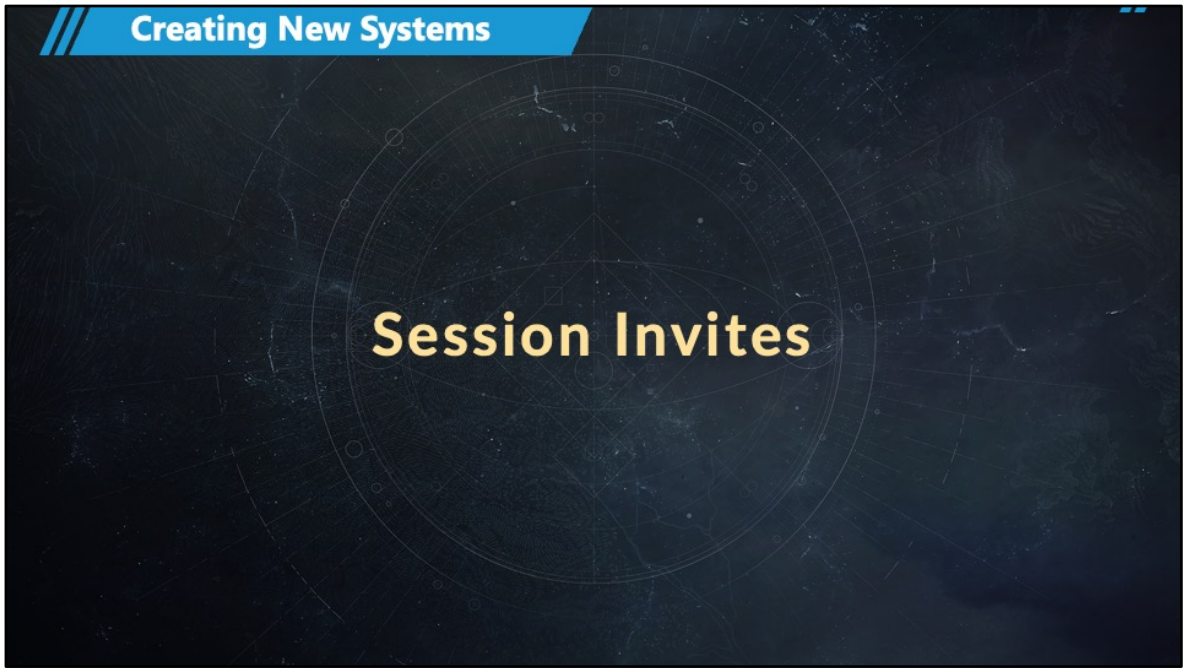
Also, other common social controls like Blocking and displaying your presence as offline or online had to be respected as well with this new Friends system.

The changes in identity and the new concept of Bungie Friends resulted in a major overhaul of our roster system. As the major UI screen that players utilize to interact with others, a lot needed to be done to account for cross-platform.

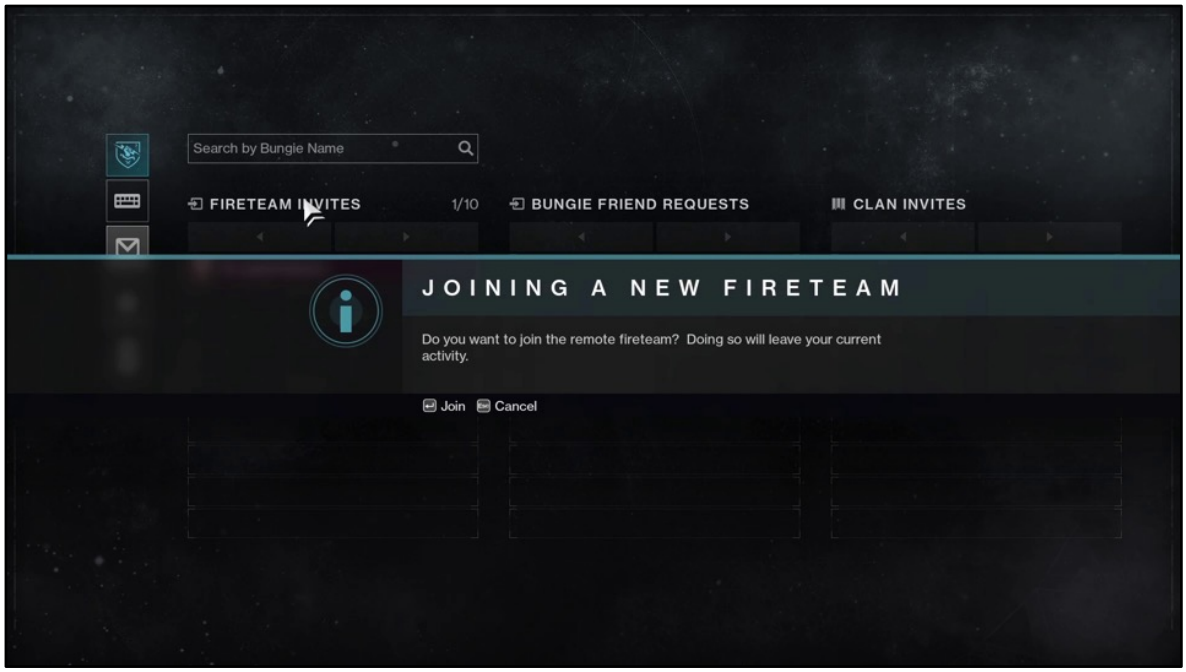
More information needed to be given to the player like what platform a friend or Nearby person was playing on. Now we show a little icon next to each person's name to give that indication.



Additionally, to help other players with identifying you, player details screens were updated to show all your linked screen names from other platforms. In this case this player is playing from Steam with the profile name TinyTimGo, but their Bungie Name is something different. This makes it easier for people to recognize who you are.

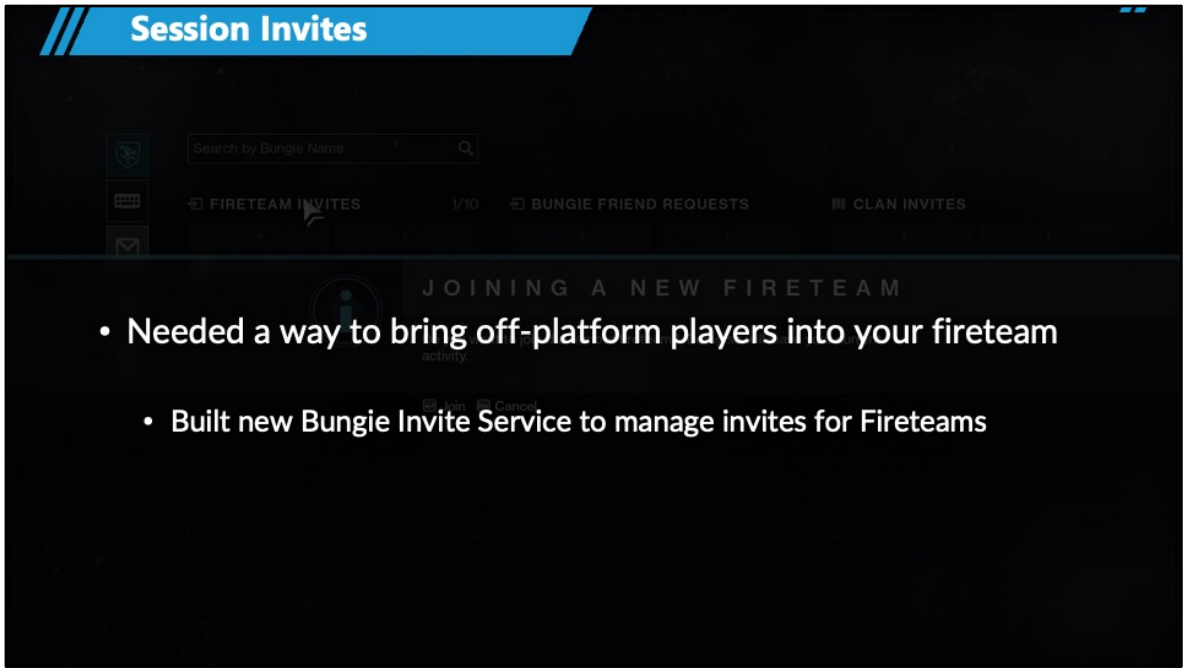


Let’s move onto the changes to Session or Fireteam Invites.



In Destiny 2, Fireteams are a group of two to six of players you play through various activities with. It's your squad!

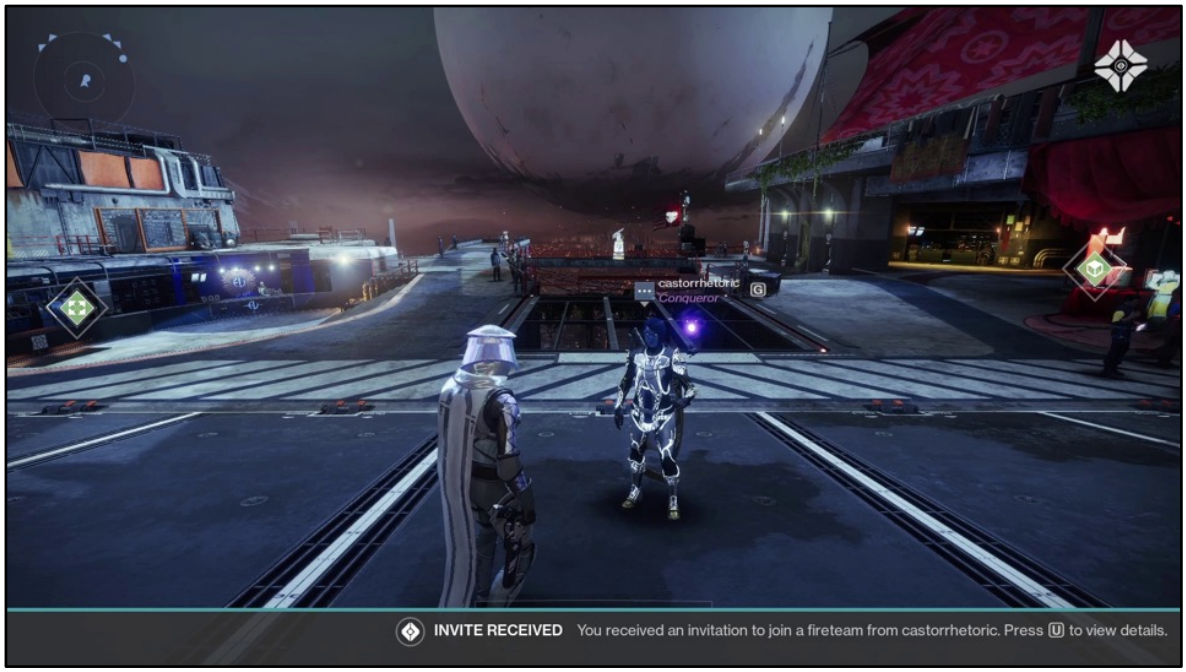
Fireteams are a core social and gameplay component of the game. As that's how many players engage with Destiny 2 content, by hopping online and fireteaming up with their friends to take on a raid or Trials match.



Much like Friends lists, we were relying on the platforms for Session Invites. Before, you would go into the roster and find your platform Friends list, and send people invites to join your fireteam. That would use the platform APIs to send the receiving player a notification. Like on Xbox, if you sent an invite, on the Xbox UI, the receiver would see a toast notification that they could accept and it would open up Destiny and have their character hop into your fireteam.

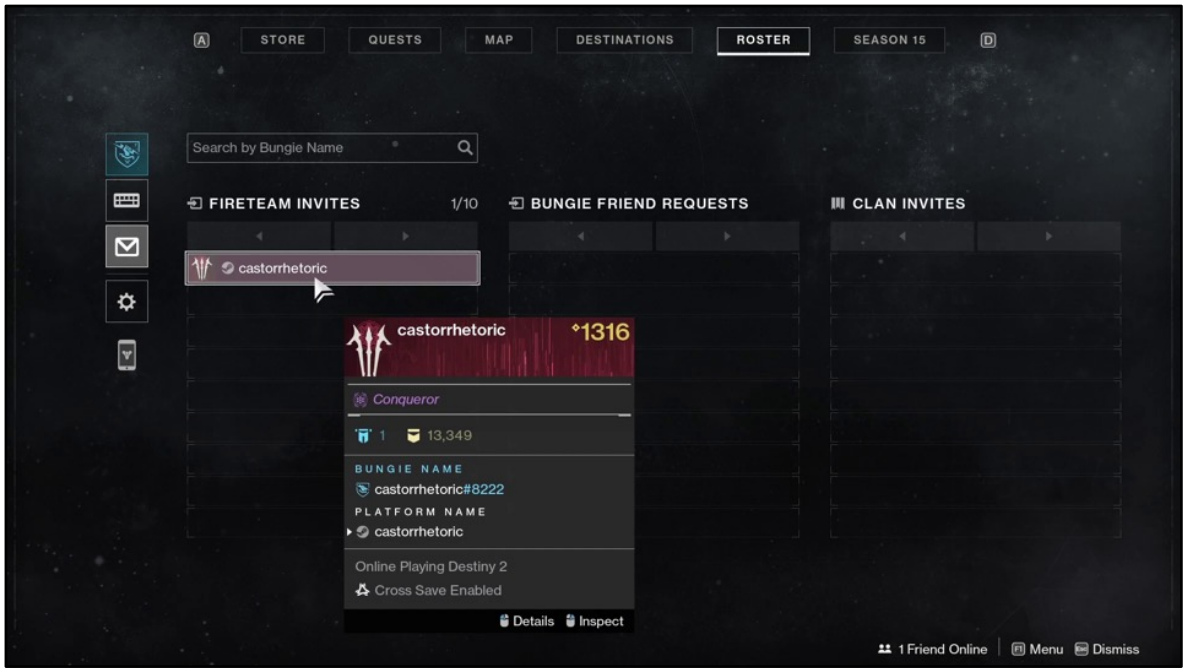
But now, we needed a way to bring off-platform players into your fireteam. There wasn't a way to call into other platforms' invite APIs from a different platform.

So we had to stand up another service from scratch to manage invites for Fireteams.

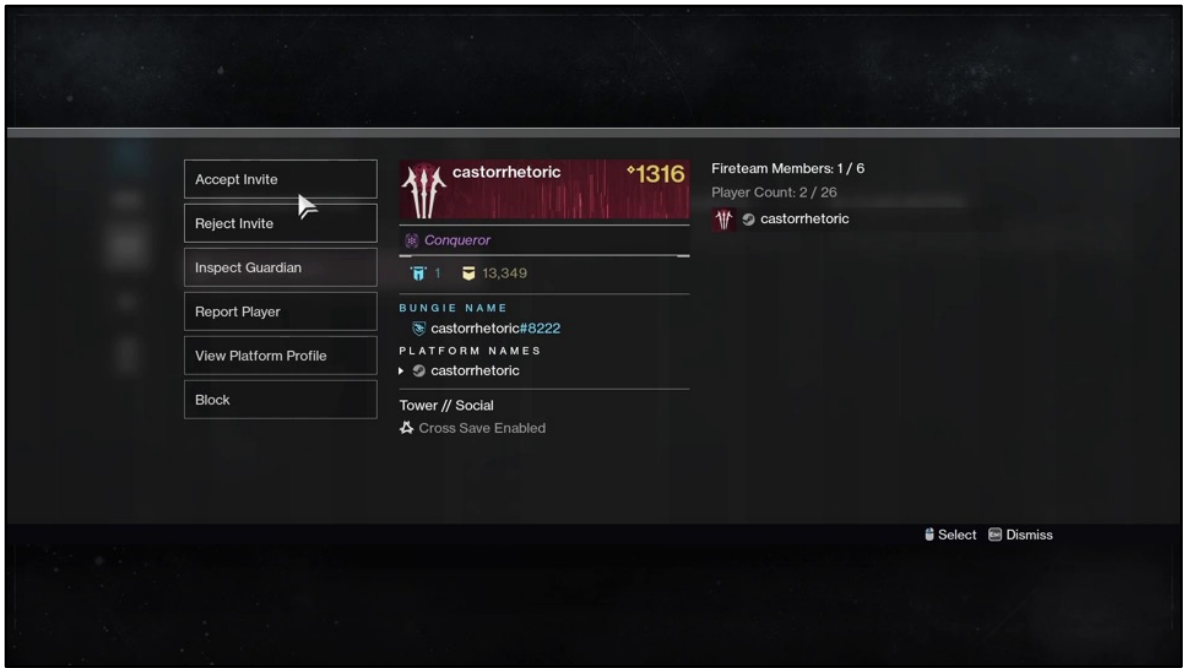


We created the Invite Service to allow players to create and join Fireteams with whoever they wanted.

We leveraged our own notification system to show players had invited them to a Fireteam, as you can see on the bottom of the screen.



Additionally, new UI had to be shown to view and manage pending invites. So we created a view for that (which Friend Requests also go through too).



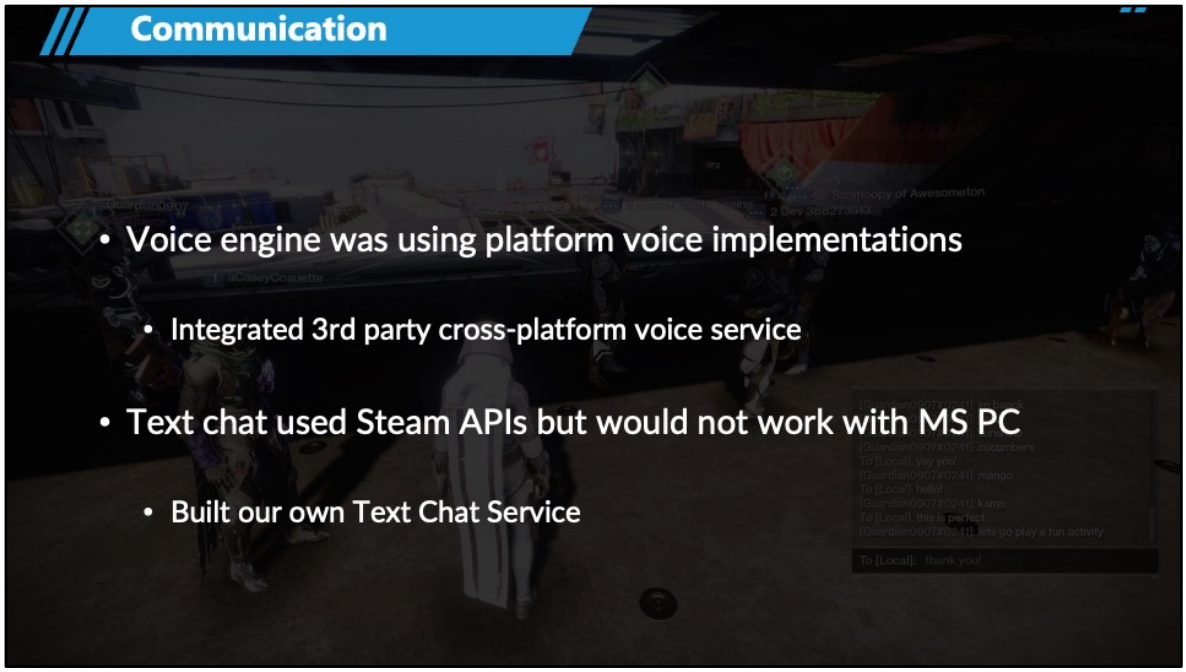
From here you can see stored invites (which do expire after some time), and action on them, like Accept, Reject, or more.

Accepting the invite pulls you into the Fireteam.

While fireteams act as the group of players you play with in various activities of Destiny 2, to really engage with each other, you need to be able to communicate.

In Destiny 2 players are able to voice chat.

And also text chat.



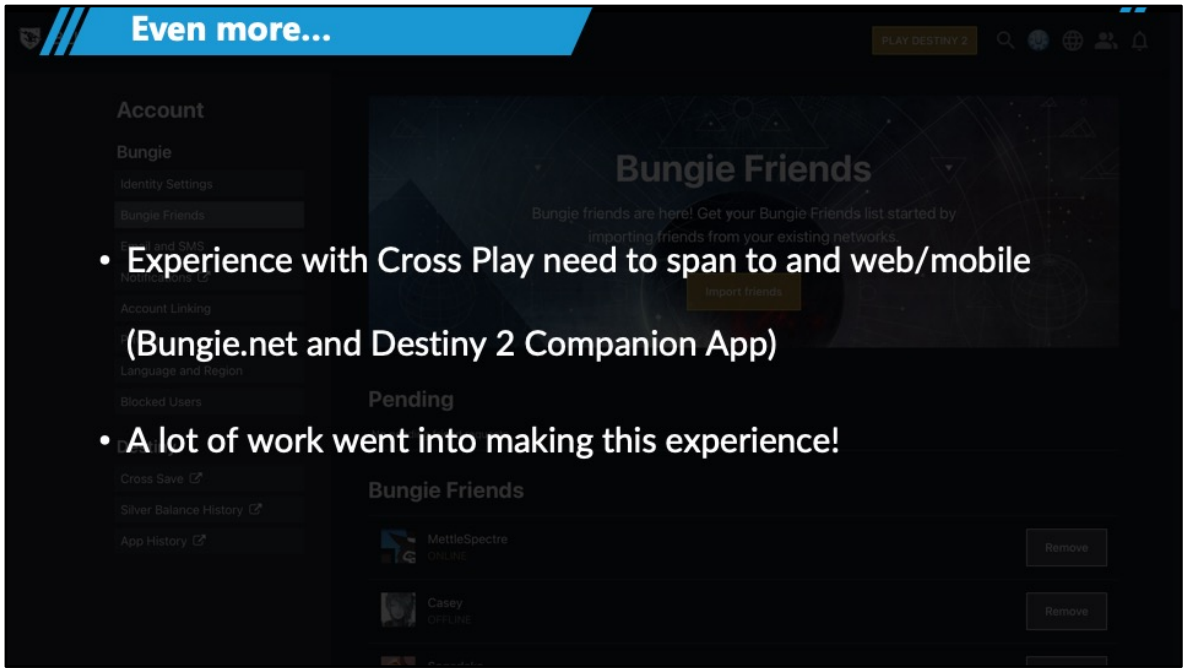
Like many other systems, our voice engine was previously using platform voice implementations. Sony's APIs for PlayStation players to talk to each other, Microsoft ones for Xbox players, and so on. But now with fireteams that had players across all platforms, they wouldn't be able to talk to each other.

What emerged was the need to have a solution that would encode and decode audio in a way that was portable across all platforms. So we turned to using a 3rd party voice service to handle cross-platform voice.

On the text side, previously only PC players would text chat with each other, and the only PC platform we shipped on was Steam, so it made sense to use Steam APIs. However, timed with the launch of Cross Play was also the release of Destiny 2 on the Microsoft PC Store. Now with another PC platform, our existing solution was not viable if we wanted PC players (at the very least) to be able to communicate.

Additionally, guided by our goal of not letting platform limit who you could communicate with, we wanted to expand text capabilities to console players as well. To that end, we built our own Text Chat Service to send messages to players.

We’ll talk more about these decisions to build our own Text Chat and integrate a 3rd party voice solution later in the talk.



So all that we covered with these systems describe what we did in just the game client. But there's even more to the Cross Play experience.

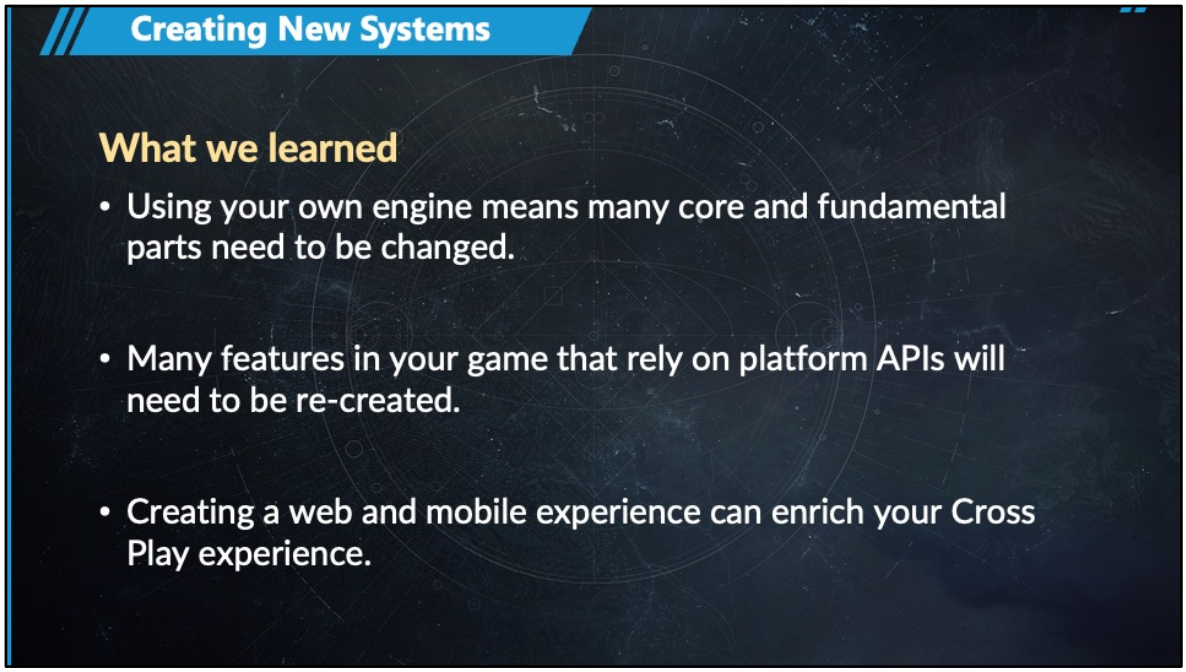
When building Cross Play, we knew that parts of the experience needed to be accessible from the web and mobile sites. In our case that was Bungie.net and the Destiny 2 Companion App respectively.

Having out-of-game extensions of Cross Play offered deeper ways to manage and engage with the experience. Like it was only through Bungie.net that players could pull all their platform lists and batch send requests to seed their Bungie Friends lists.

Additionally, iteration on web and mobile is much faster compared to shipping new updates to the game client. Offering these power user features for Cross Play in this way allowed us to iterate and polish these features more more quickly.

Also, a peek behind the curtain, having the web experience helped with workarounds like with a bug we had at launch where player searching in-game to find people to add wasn't working correctly, so we were able to route players to the web and mobile sites to use search and find who they wanted until we got a fix out the next day.

All up, a lot of work went into making this experience



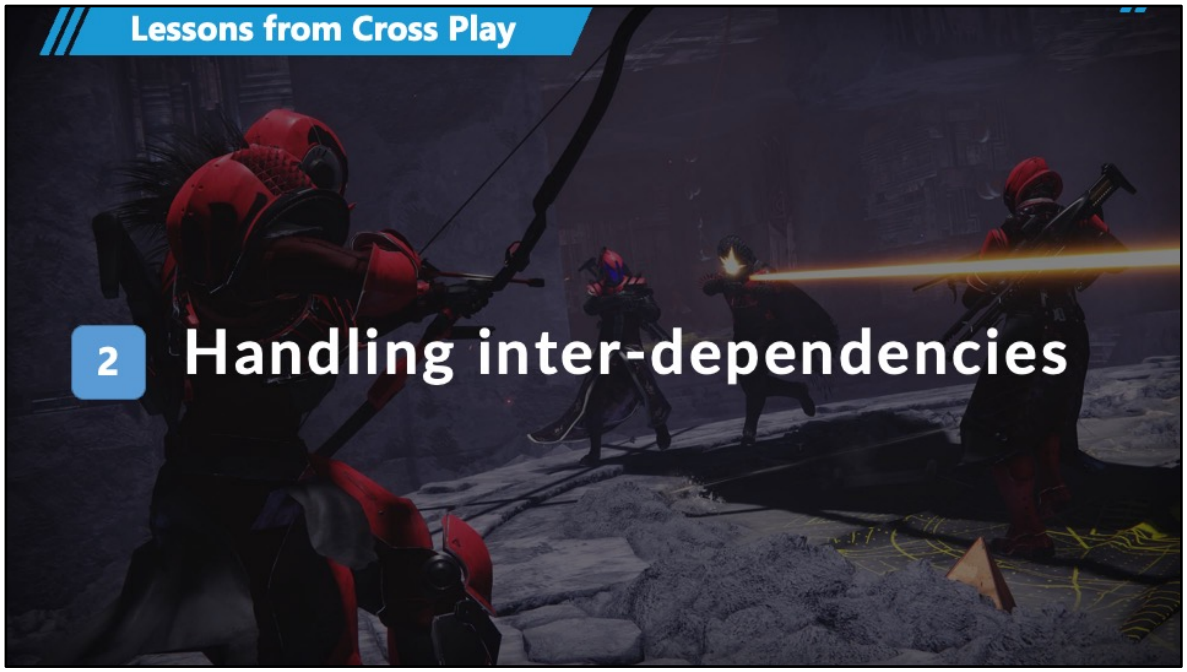
So in summary, here's the things we learned in having to build the many new systems for and surrounding Cross Play.

First, when you dealing with your own engine, many core and fundamental parts of the engine will need to be changed. From things like changes in underlying networking to revamping UI, all this means is that when planning to build your own cross-platform tech, scope and extra time need to be accounted for.

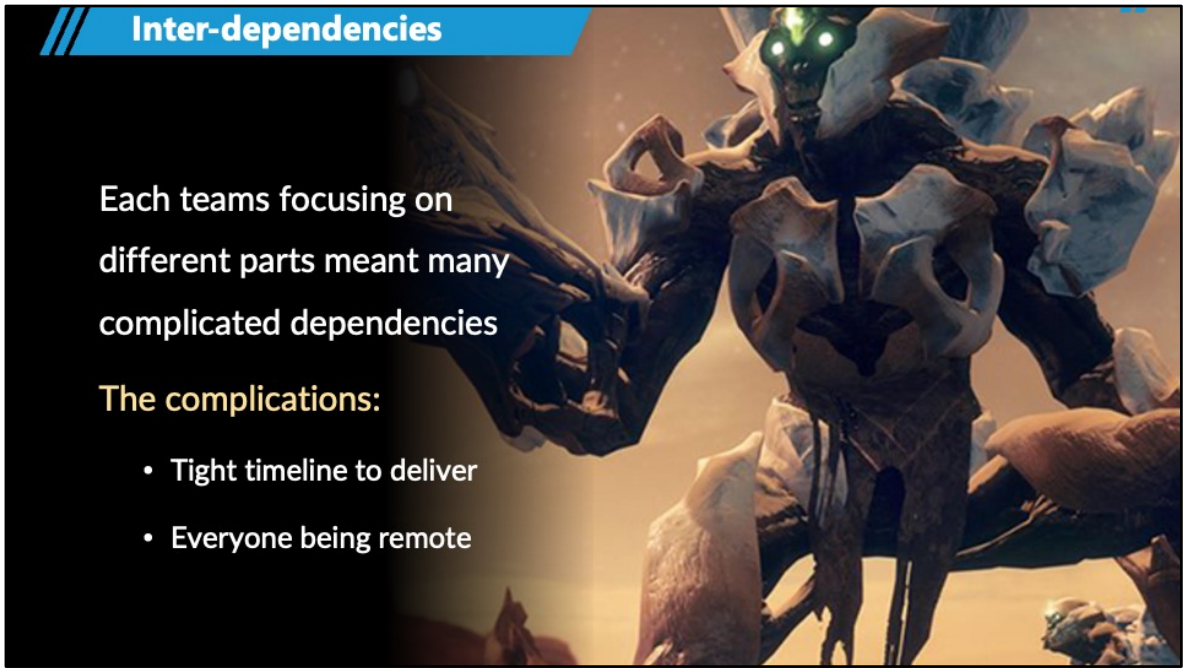
Second, when building for cross-platform, many of your game's pre-existing systems probably rely on platform APIs/features, like friends lists, session invites, voice chat… all of those. Most of these features will need to be re-created from scratch. Your mileage may vary on this because with 3rd party engines like Unreal or Unity, a lot of cross-platform portability of services and systems is already built in. But it's important to take stock of what systems you rely on platform tech that will need to be moved to cross-plat.

And third, extending your cross-platform experience outside of your game client to wherever players engage with your game can very valuable. This can make it easier to give players more fine-grain control over the many new systems you introduce.

Having web and mobile accounted for from the beginning makes for a truly cohesive experience.



Now that you have an idea of the systems that were involved with making Cross Play, it gives you a better sense of the scale of this large cross-team initiative. And with projects of this kind of scope, comes many complicated dependencies between all the teams working together that need to be wrangled.



Having four different teams focusing on separate parts of the experience meant there were many intertwined dependencies.

To make things more complicated:

- We were working under a tight timeline to deliver by September 2021. This made it all the more important to ensure all teams were not blocked and could make progress.
- Also, with everyone being remote, come complications to being in sync. “Ad-hoc hallway and desk conversations” did not happen as organically which smooth out questions and issues.

So how did we approach this issue?

We were fortunate to have so many people working on this feature. But with more people also comes the challenge of making sure everyone was building towards the same goal. We emphasized team alignment and providing as many avenues as possible to identify blockers and open issues for anyone on the team was encountering.

We set out to create as many opportunities as possible to ensure we were all building towards the same thing. And it's a bit cliché for a producer to bias this way, but especially in a remote work from home world, the solution came in the form of syncs and meetings.

What we found was that having recurring syncs that pared down attendees by feature areas (like everyone working on Communication and chat), as well as ones by discipline (like Test and Engineering) allowed us to focus conversations on specific topics (like open issues and questions), or upcoming integrations of certain features.

One example of how this benefited us was by having a weekly sync to discuss specifically Friends and Presence. By focusing in on how the Bungie Friends Service would interact with the UI and everything else in the client, people were able to bring up their questions. We realized we needed to figure out how Blocking worked in a cross-play world, and this ambiguity was being felt by most people working on this system. We were able to quickly address this issue and have engineers and designers in the same room to discuss what we needed to do for the experience.=

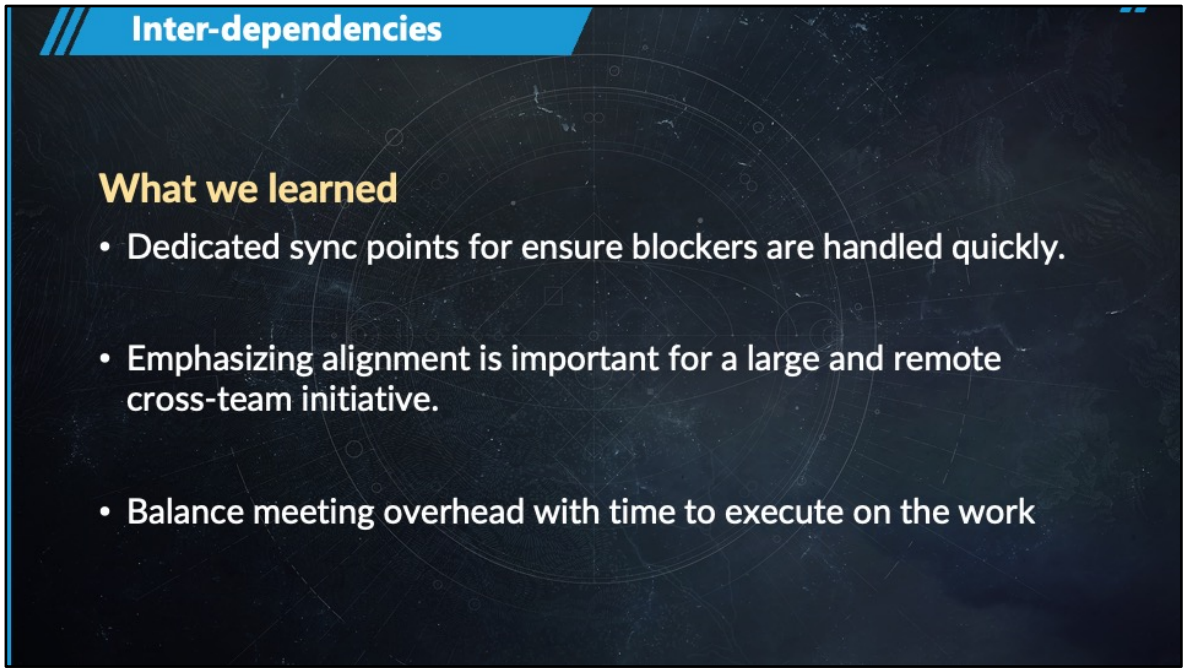
This was followed up on in weekly Engineering syncs for the team to discuss the

feasibility of certain technical solutions, like how we could ferry all the right data around.

We also used recurring rituals like meetings and where everyone on the core team was invited. This ensured everyone was present and able to understand progress holistically.

We also had core-team wide weekly playtests that allowed the whole team to see how the experience was coming together. It was definitely a morale boost seeing how features kept getting more polished with each week.

From all these points above it's pretty clear that the approach we took was giving team members lots of exposure to each other to build cohesion empower them to identify and mitigate issues.



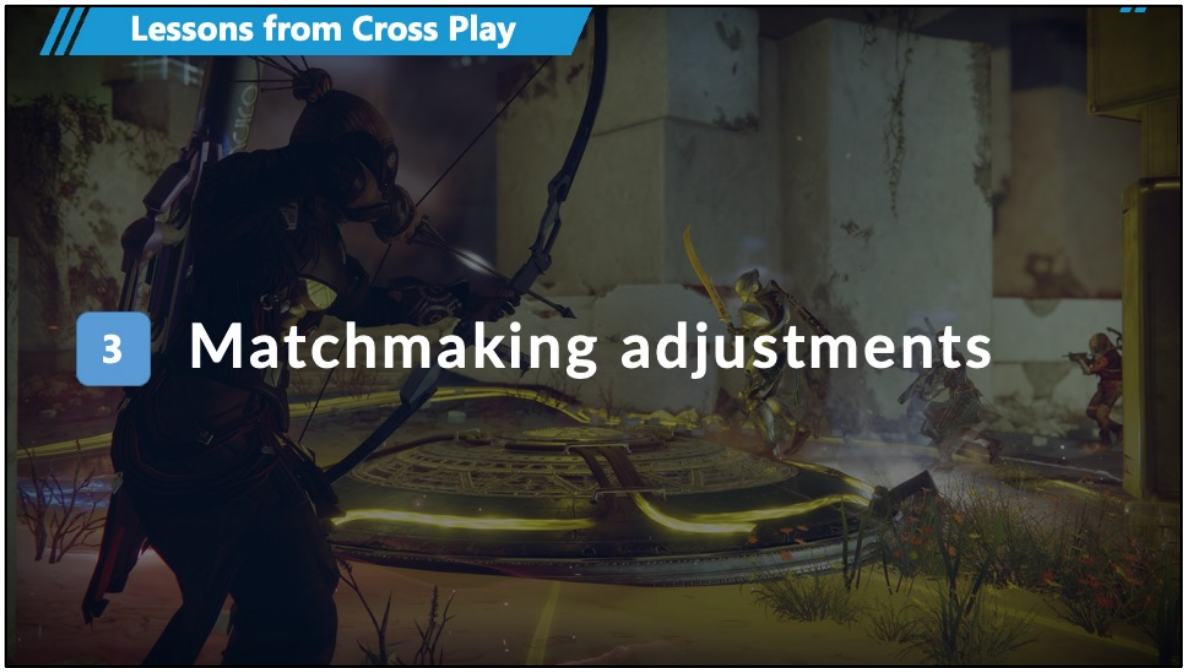
So to sum it all up:

Creating dedicated sync points ensures blockers are escalated at a cross-team level and handled quickly. Having syncs that slice your team in multiple different ways like by working group of feature area, or discipline, allows for different opportunities to discuss issues at different scales and depths.

Next, when dealing with a large team that might also be remote, it's important to make sure everyone is aligned. Offering as many opportunities for the team to "see" each other helped make it easier for people across the team to proactively approach anyone else on the team with issues they might be facing and gives people a sense of how their work fits into the bigger picture.

Lastly, if you heard at all the talk about syncs and working groups and such, and thought, "that's a lot of meetings" I'd agree. Syncs to build consensus across the team come at the cost of IC time (or time to actually do things). And that's overhead on your team. So it's all the more important that a balance is found. Admittedly, we initially over-indexed on syncs. Too much time spent in meetings meant not enough time to do the actual work. But we quickly found a better cadence by limiting total

meeting time.



So we talked about the “what” that was involved with Cross Play, and the “how” we went about building it. Let’s move into some of the design considerations that were made along the way.

First up is matchmaking.

Players come to Destiny 2 with many different styles of play. From a preference for each class (anyone here a Warlock main like me?!) to liking certain damage types like the Solar Dawnblade subclass. The differences in play styles don't just stop there. It extends to the platform a player is on.

For example, Keyboard and mouse players tend to have quicker reaction times compared to console players. The perception of this is especially important in competitive modes.

When you throw them all together with Cross Play, you need to consider what each type of player is coming with. For Destiny 2 specifically, some of the interesting challenges in this space include:

- The game has both Competitive and Cooperative components and players have different expectations for how they engage in each activity. Like with the reaction time difference. Maybe when you are in cooperartive mode playing a strike the difference doesn't matter, but if you are in a competitive Crucible match as a console player, if you were matched against PC players, you might perceive reaction time differences as unfair.

- Also, PC and console platforms have very different security landscapes. Players using hacks are something we're constantly fighting against and can make for a lesser experience for a console player not as used to this.
- So how do you make it "fair" for players on different platforms to play together while embracing full cross-platform?

Figuring out how to match players on different platforms in the best way possible was an important decision we had to make along the way.

We landed on making different matchmaking pools based on type of activities. This ensured fairness and balance while allowing players to play together.

Not all modes of the game needed to create different matchmaking pools. There are strong player feelings about fairness, whether it be making controllers as competitive as possible with mouse and keyboard, or with the different security landscapes on console & PC. We opted to keep PC players separate from console players for certain modes.

We felt protecting players and ensuring fairness was most important to ensure the best experience for the greatest number of players.

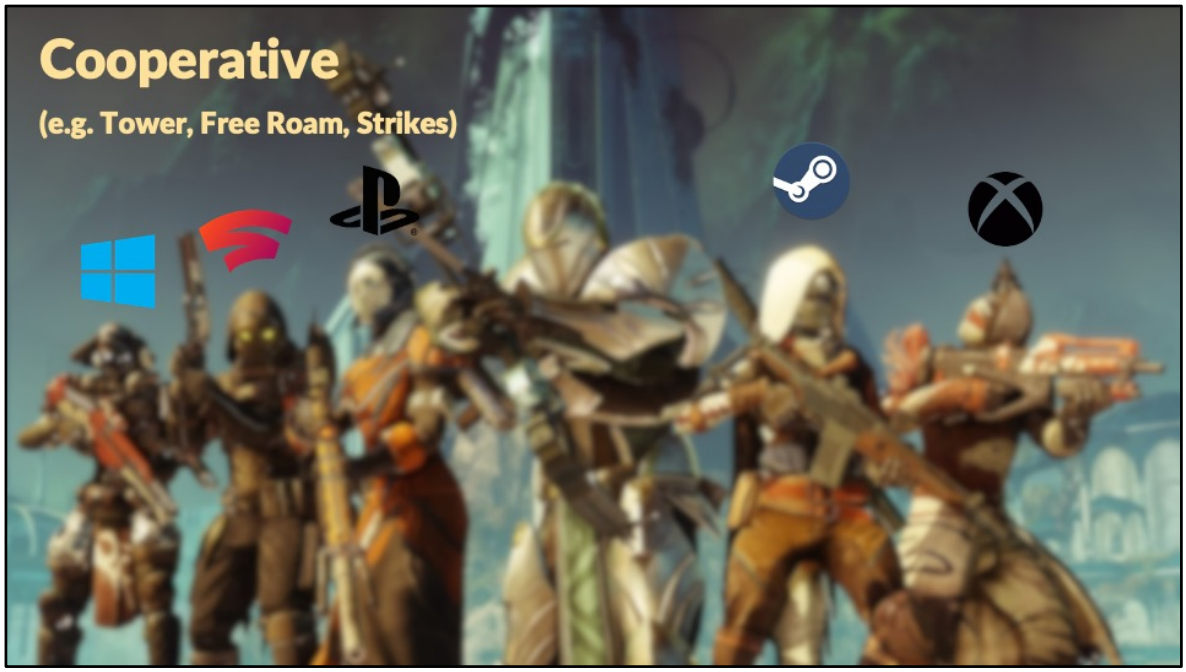
We arrived at this decision from conversations we had with our Player Support and Community teams who stressed the importance of protecting players’ experiences.

Additionally, we chose not to split pools by other factors like input modality like having a keyboard and mouse pool and a controller pool, because making that distinction has interesting complications. Like what if a PC player plugs in their controller to kick off a competitive match, but unplugs that and reverts to keyboard

and mouse once the match starts? That distinction no longer matters.

So guided by that principle of fairness and protecting players as the most important, it made our matchmaking design decisions more clear.

Starting with Cooperative modes.

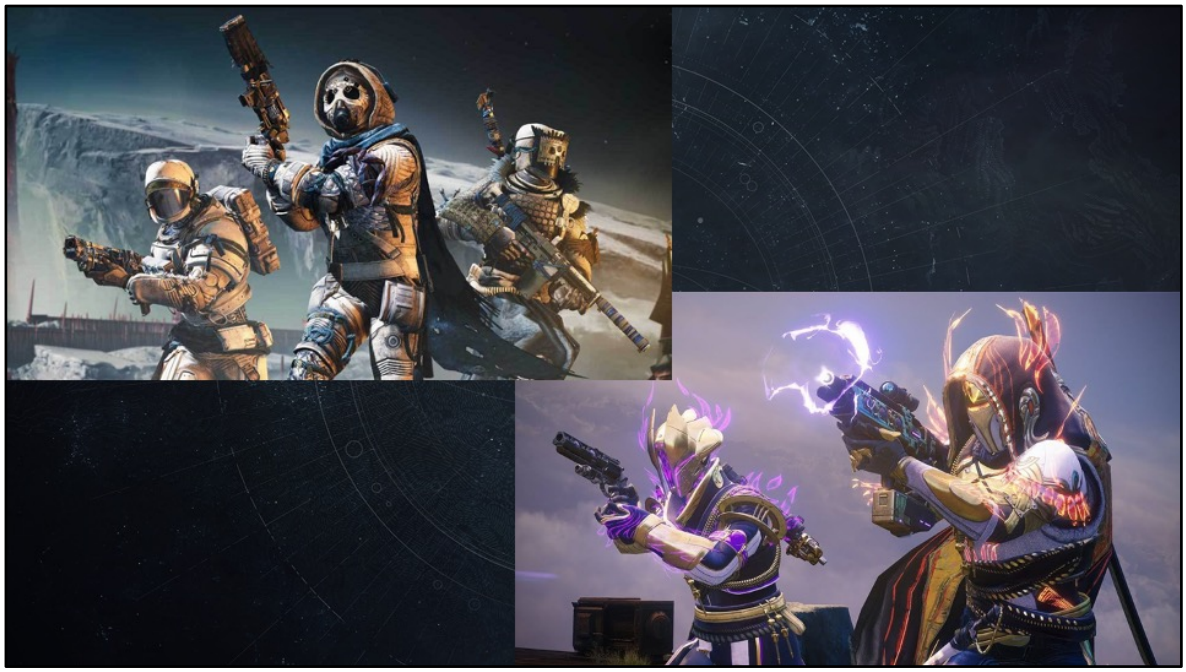


We didn't use separate pools.

Everyone plays together.

There was no need to create separate pools of players since there is no "edge" one platform would have on another and the experience for players in modes like Strikes would not be drastically affected.

So when you land on a destination like the Moon and free roam around, you're automatically matchmade in world with players on any other platform.



Moving to competitive modes is where we utilized separate matchmaking pools.

We created two different matchmaking pools.

The first is the Console pool. In this one we have Stadia, Xbox consoles, and PlayStation consoles.

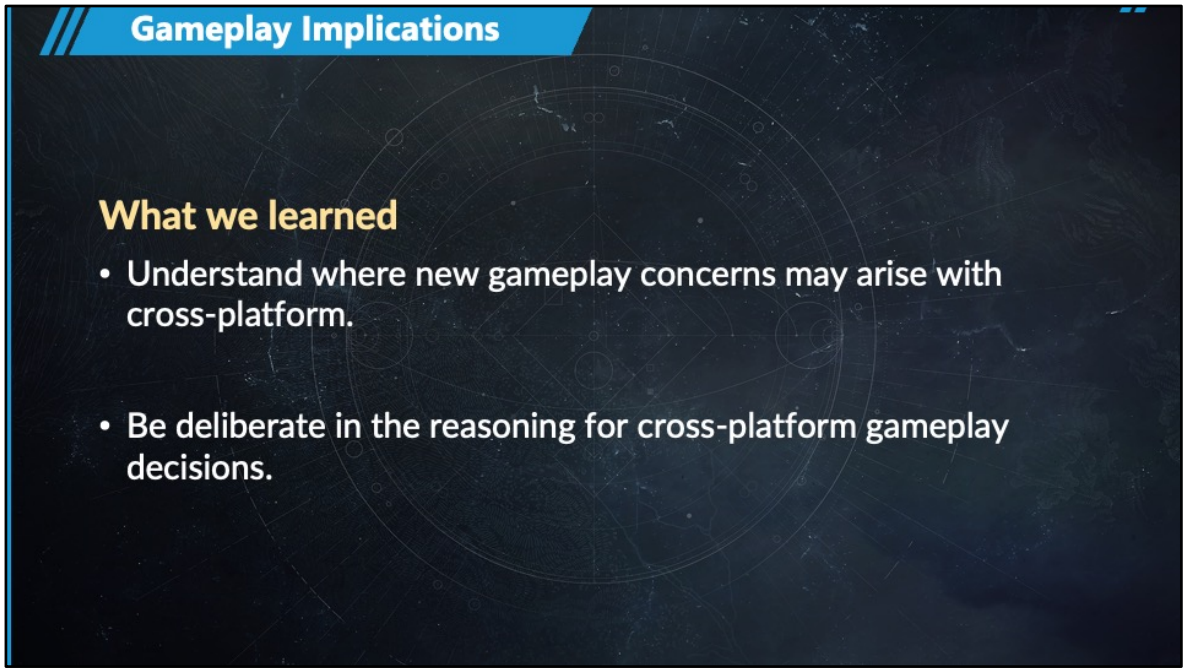
While Stadia is technically a PC (or you're streaming from one). That PC is hosted within a data center. It's unlikely to have exploits installed or tampering done with the client. So we included it with the Console pool.

Then, we have the PC pool in which we have Steam and Microsoft PC Store players.

So now, when you launch a Competitive activity where you are playing against other players, depending on what platform you are playing from, you'll be matchmade with other players in the corresponding pool you belong to.

That’s not to say you can’t create a fireteam of mixed platform players in different pools. We accounted for this by making it so that PC players cannot cross over and enter the Console pool. Only platforms in the Console pool can cross over and play in the PC pool.

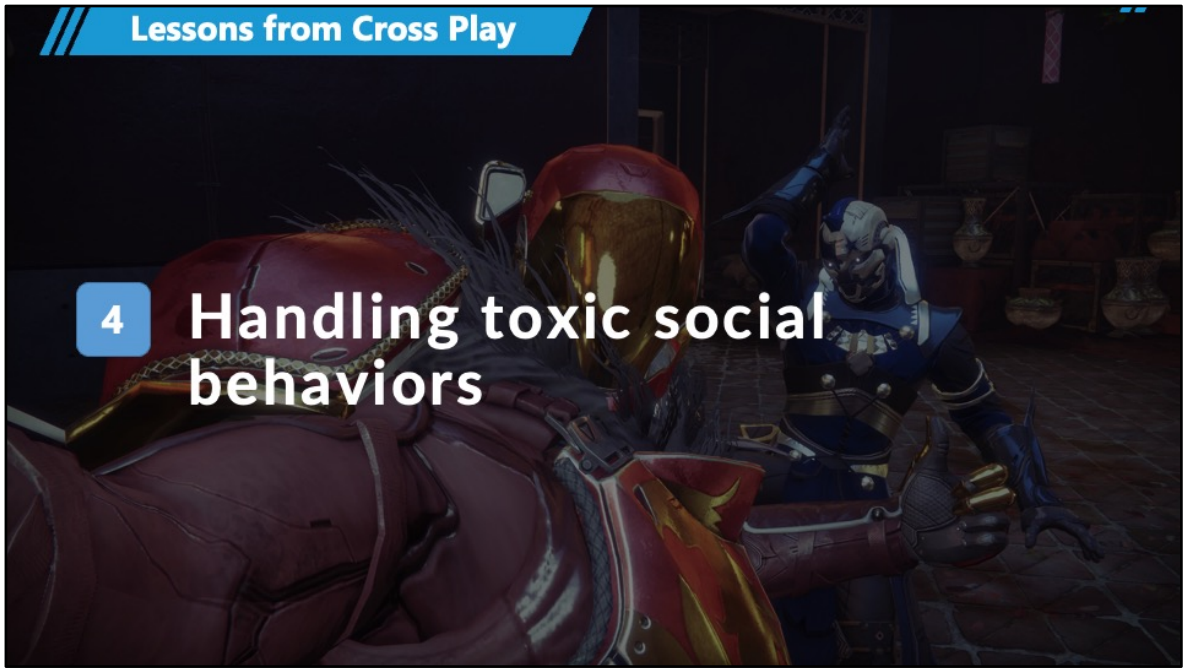
So for example, if you are PlayStation player, and you fireteam up with some friends who are on Steam and load into Gambit, even though you're on Stadia, you'll be playing against other PC players in the match. This keeps the boundaries we set for PC and Console pools.



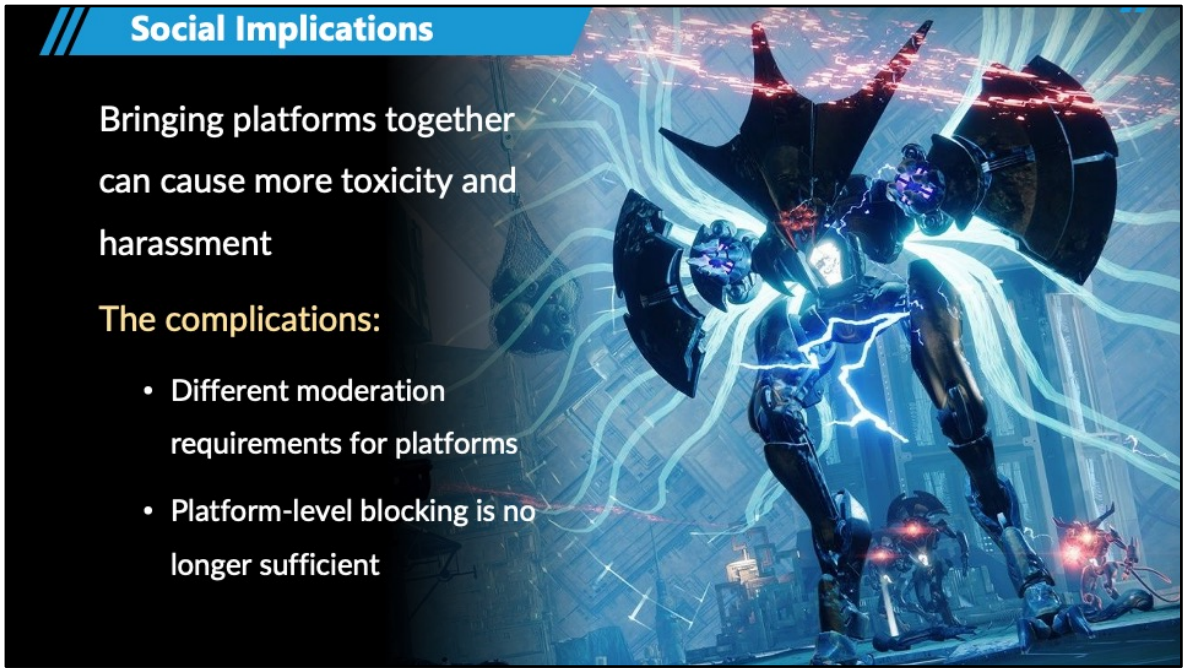
So bringing it all together for cross-platform gameplay.

First, understanding the areas of concern for gameplay implications with cross-platform will help guide your discussions and decisions. Understanding where your players' concerns and reservations lie will help identify these areas. For us, that came from engaging with Player Support and Community teams which gave us a direction so we could proactively problem solve.

Next, if you're creating different platform pools for matchmaking, being deliberate for what reasons you doing so is a useful razor for design. For us, using the guiding principles of protecting player experiences and fairness made it much easier to have discussions about matchmaking pools and how and when to split up players.



In addition to the implications cross platform has on gameplay, there are also new social behaviors from players that need to be handled.



Each platform posses their own standards for “acceptable” social behavior when online.

When you combine players from all platforms, these differences in how players are used to interacting with each other can give rise to new forms of toxicity and harassment.

Take for example the kinds of usernames that are allowed on each platform. Some platforms have strict filters against profanity and derogatory language, but on others you can name yourself anything you want. So a player coming from a platform with filtered names could suddenly be exposed to all sorts of profanity, which could range from lewd to derogatory and bullying.

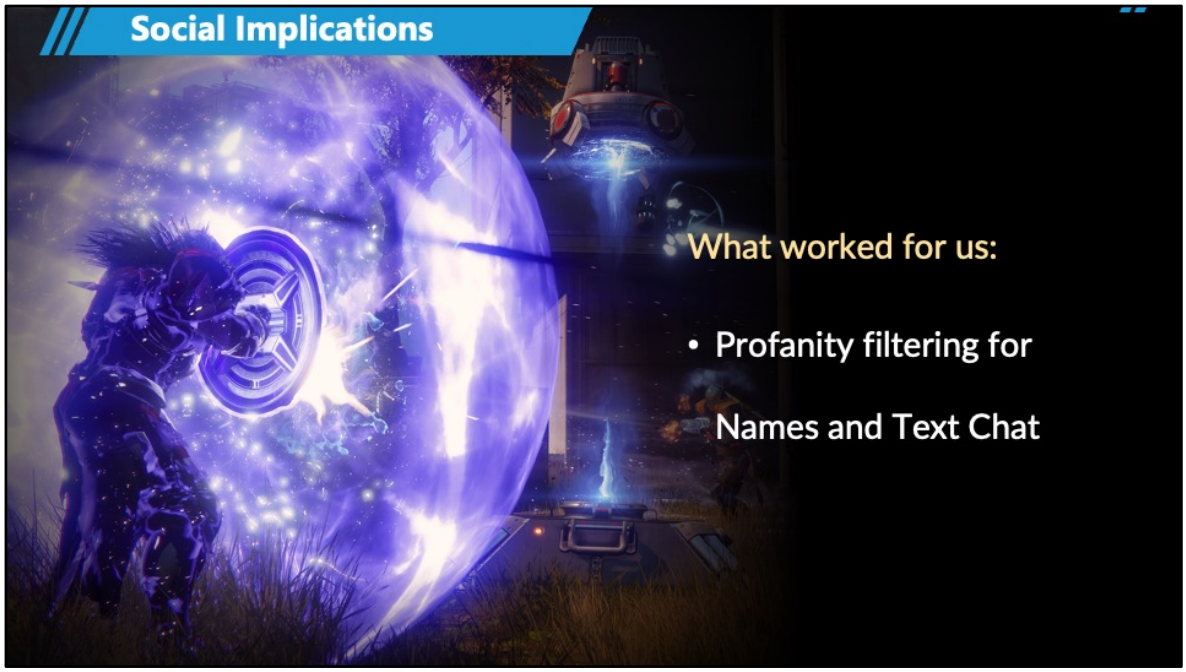
Also, platforms offer social controls like blocking and settings to prevent external contact. But with Cross Play, new vectors of attack open up. If you platform block an aggressor on your account, what is to prevent them from hopping over to another platform to continue harassing you.

So part of the work for Cross Play became addressing these social behaviors.

There’s many other forms of online harassment in games out there, and when you expose players to bigger populations, as is the case when you bring all the platforms together, that’s

a risk you take on. It's especially important to recognize that these forms of antagonistic behaviors tend to be directed at minorities and more vulnerable groups of players.

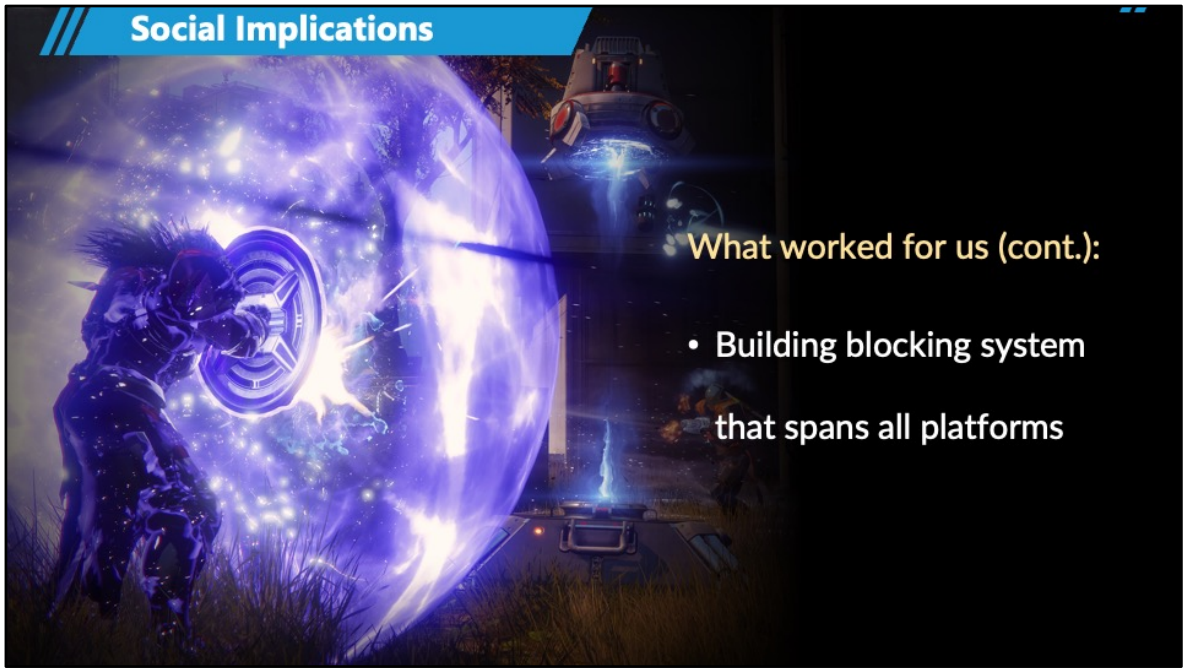
Much like how our matchmaking pools decision was guided by wanting to protect players, we did the same with social controls and systems. We dedicated a lot of time, effort, and thinking within this space to try to prevent abuse and harassment.



To accomplish this, we focused on a couple of areas.

First we addressed user-generated content that's associated with Cross Play. In this case, most important was Bungie Names. It became clear in our early discussions that we needed to filter for profanity since some platforms have various levels of screening of user names. If there us no platform-level filtering, there was the risk that inappropriate names and words used in platform names would be passed in and seed the Bungie Name.

To prevent this, we adopted a 3rd solution that is versatile and context aware. Which I'll discuss a bit later in this talk.



Another area we worked on was adequate blocking systems to manage who you could interact with. We needed our new Bungie Social Network to account for bad actors.

So in our Bungie block system, we aimed for parity with platform blocks, but extended it cross platform.

We made sure the Bungie Block system spans all platforms. If you Bungie Block a player, all accounts that are associated with them are blocked. This ensures that if a you block a harasser, they can't hop around platforms.

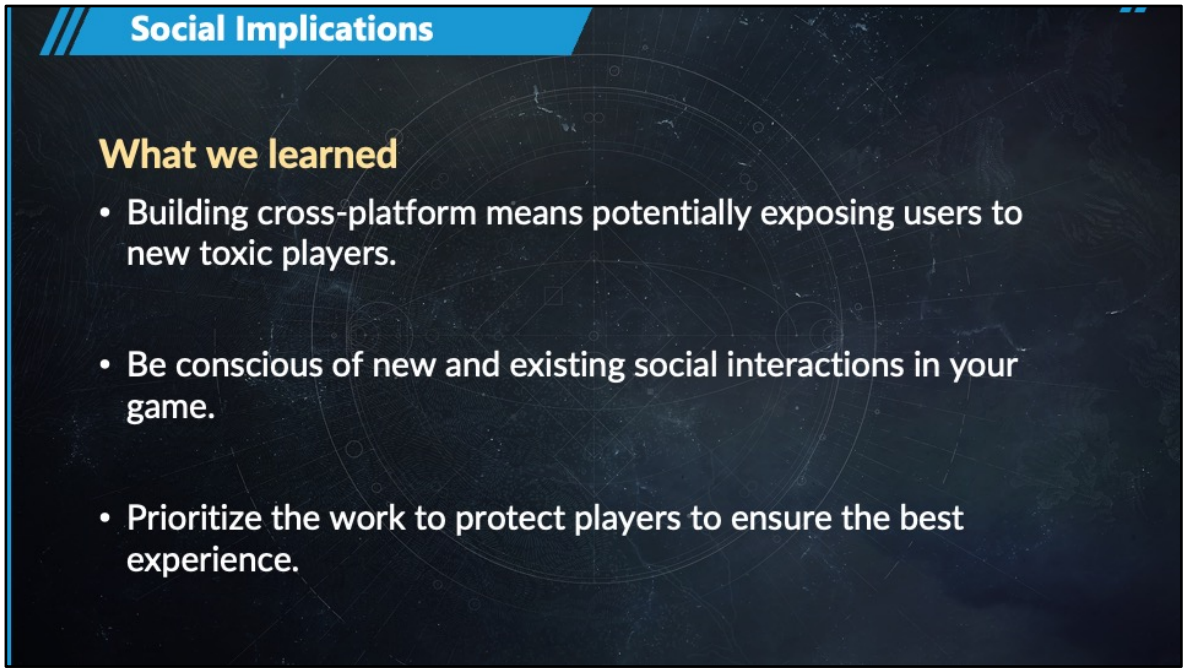
By building these systems, we better protect the player and also give them more control of their social interactions.

Here's a run through of the Bungie Block.

Upon inspecting an offending player that you intend to block, you are given the option to Block the player.

Doing that, to make it clear what this action does, a prompt is shown. Bungie Block prevents all forms of communication, whether it be chat, fireteam invites, friend requests, and more.

Once that is done, there's a very clear indication in the player details that they are blocked and this spans to all platform accounts that are linked to them.

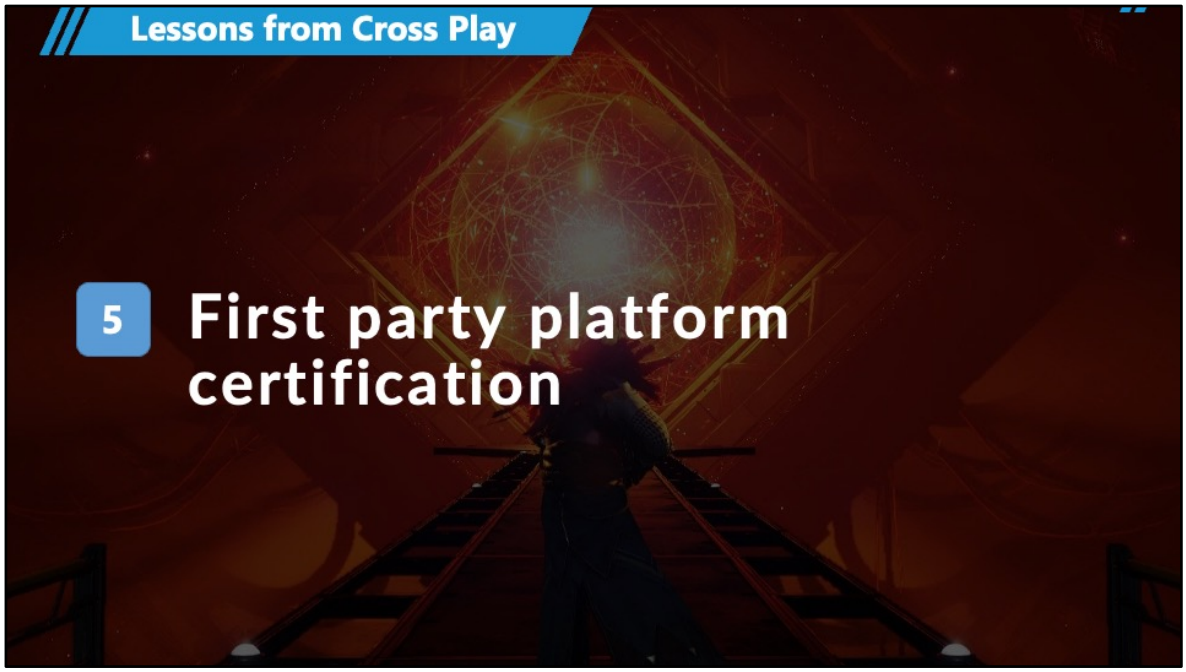


So when it comes to social systems being mixed with Cross Play, a lot of considerations need to be made.

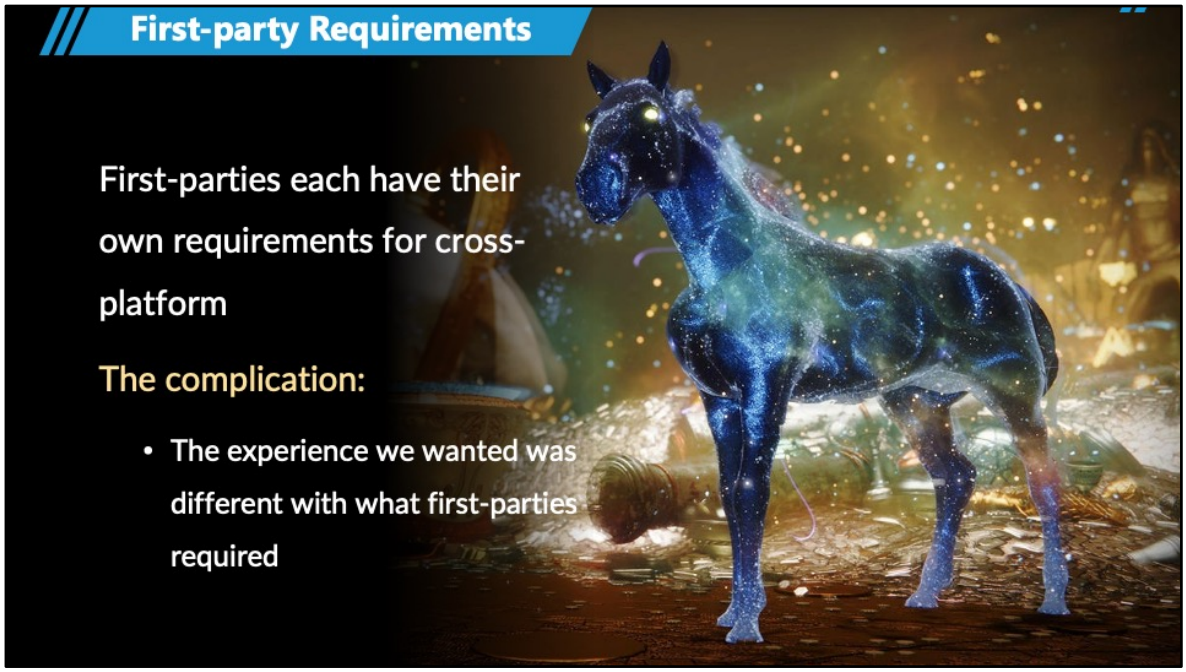
Building cross-platform potentially exposes users to new toxic players. More players and platforms means many new avenues for toxicity and harassment.

Being conscious of existing social interactions existing in your games and new ones coming with cross-platform will help you figure out areas where new forms of harassment and abuse could come from.

Additionally, when these areas are identified, prioritizing the work to protect players is very important for ensuring the best experience for all players. For us, it was requirement for us to land blocking for ship. Only then could you ensure you are properly protecting players.



Let’s move onto some of the more external factors we faced when building Cross Play. While you can design and build all the perfect cross-platform systems, to be able to release, you also have to meet the requirements of first-party platforms.



Destiny 2 ships on a lot of platforms.

Having previously shipped on all these platforms, we're pretty familiar with cert requirements that need to be met.

So going into the Cross Play project, we did our due diligence. Many first-parties have whole sections in their certification requirements dedicated to cross-platform play and their requirements to allow for that.

However, from our initial designs, there were differences in what we wanted in the experience with what first-parties require.

Some examples of this include:

- Some platforms require players on their platforms to be primarily identified by their platform names.
- In other requirements, some platforms don't allow the showing of logos or icons of other platforms.

These are just a few of the cert requirements we needed to work through to ship Cross Play.

To ship Cross Play, we had to work very closely with first-parties to align their requirements with our aspirations. One of the stated constraints we had at the beginning of the project was to respect first-party partner relationships, and work to obey the spirit of requirements. It was a long process that involved many engagements, but we feel like all parties came out happy.

It's important to call out here is that first-party relationships vary drastically from studio to studio. At Bungie, we're very fortunate to be able to work very closely with first-parties to reach a mutual understanding.

So what steps did we take to accomplish that?

First, during pre-production planning we made sure to go through the requirements for each platform partner and flagged areas that differed with our designs. This helped us identify areas that we wanted to focus on.

Being able to meet with first-party partners helped us understand the spirit of the requirements and how we could make that work with our intended experience.

It was a long approval process, but starting the thinking and engagements early made sure we were set up for success.

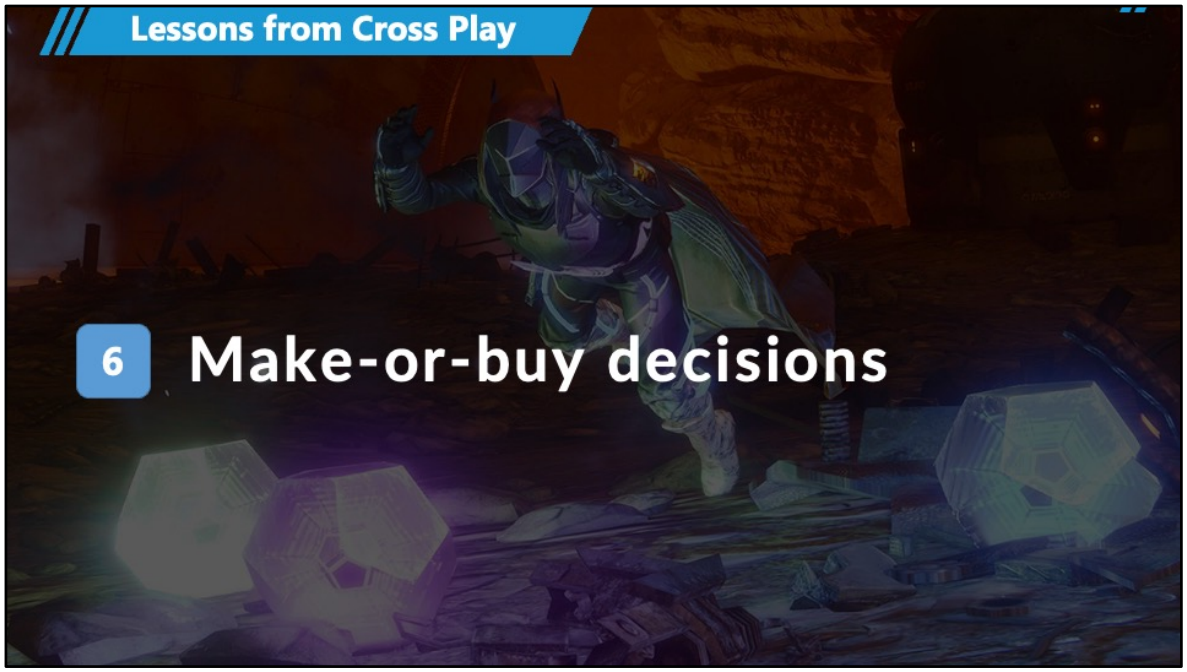
So the key things for you to takeaway from our experience:

You need to be very aware of first-party requirements with cross-platform play from the very beginning of your production and accounting for them in your designs is vital to figuring out a path forward to ship.

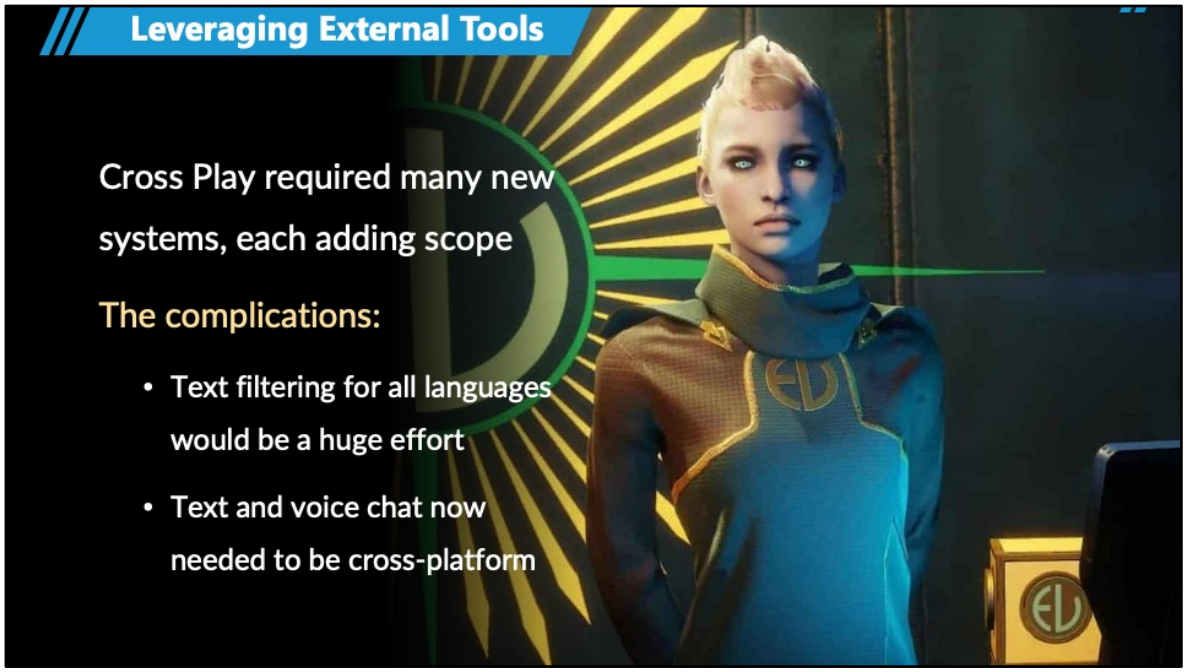
Executing towards your vision while trying to best respect their requirements allows you to deliver the experiences that work for everyone. You want to ship something great, and on the platform partner's side, they want to uphold brand integrity.

There’s was room between the needs of both sides to be successful.

And from that Cross Play can be made a reality.



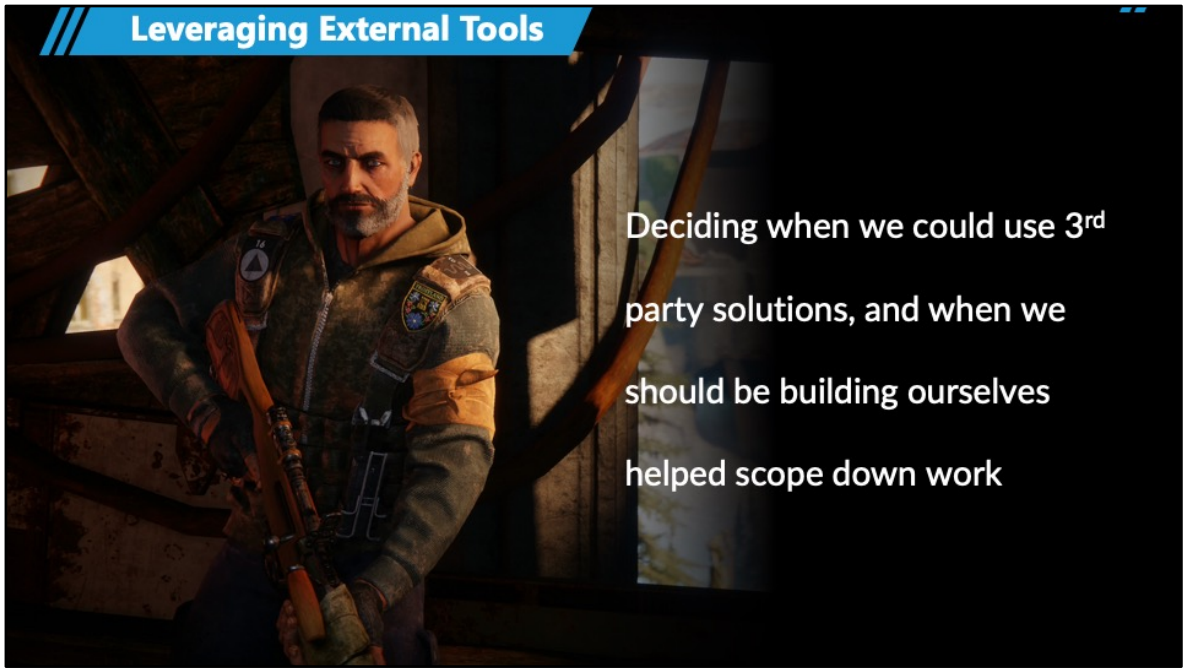
In terms of other external factors to consider when building Cross Play is with 3rd party solutions and figuring out when and how to use them to aid your development.



Like mentioned before, standing up cross-platform play means building a lot of systems that were previously reliant on platform level APIs/Services: Friends, Voice Chat, Text Chat, Invites, and more

When building these features a couple specific use cases or areas emerged that complicated these systems but were still necessary. For example, needing to ensure only appropriate names are accepted for Bungie Names, or needing a voice library that was portable across all our platforms.

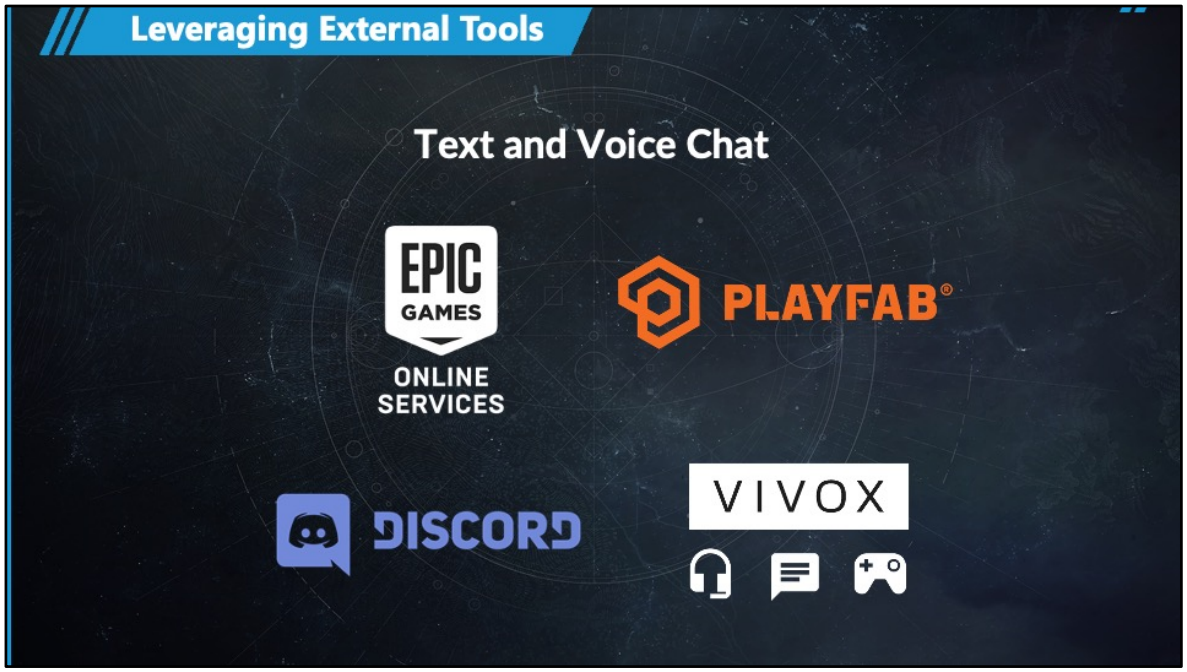
Handling these cases like building a new language filter would have added scope of the project, and being on such a tight timeline, tough calls had to be made.



Certain functionalities required lots of effort that would not fit within our timeline. For example, building a good language filtering for Bungie Names and user-generated content would be a massive undertaking. Destiny 2 supports 12 languages, screening and filtering in all these languages requires a lot of expertise and time.

So for the specific areas of voice chat, text chat, and content filtering and moderation, we conducted make or buy analyses on what we needed to make ourselves or whether alternate solutions existed that were worth it for us to buy.

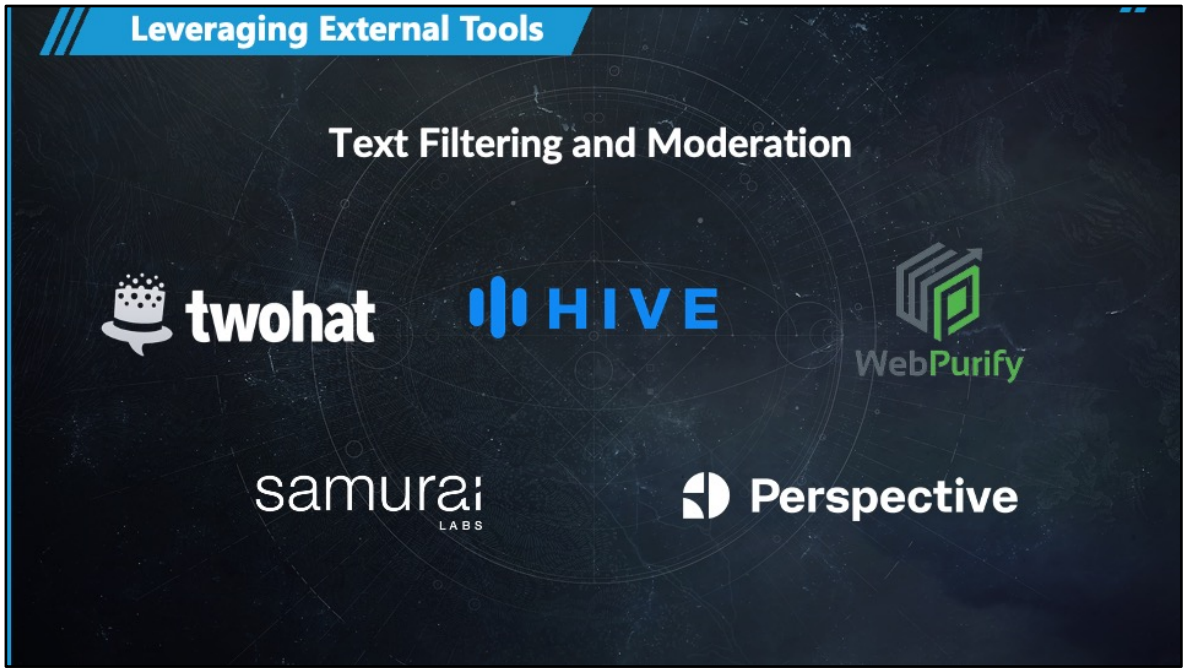
There are existing 3rd party options immersed in these spaces and build products that can be better than anything we could make given our timelines. In many cases these options offered a chance save time and would enable us to ship a better experience.



Let's start by looking at Text and Voice chat. There are many options in this space.

Some considerations we made in our decision include:

- The solution needed had to have libraries that supports all the platforms we ship on.
- The cost of relaying data ourselves was cheaper compared to using a 3rd party service to handing the volume of data. But came with long terms costs of upkeep and maintenance.
- Lastly, the ease to integrate a 3rd party solution. What kind of documentation and support was available.



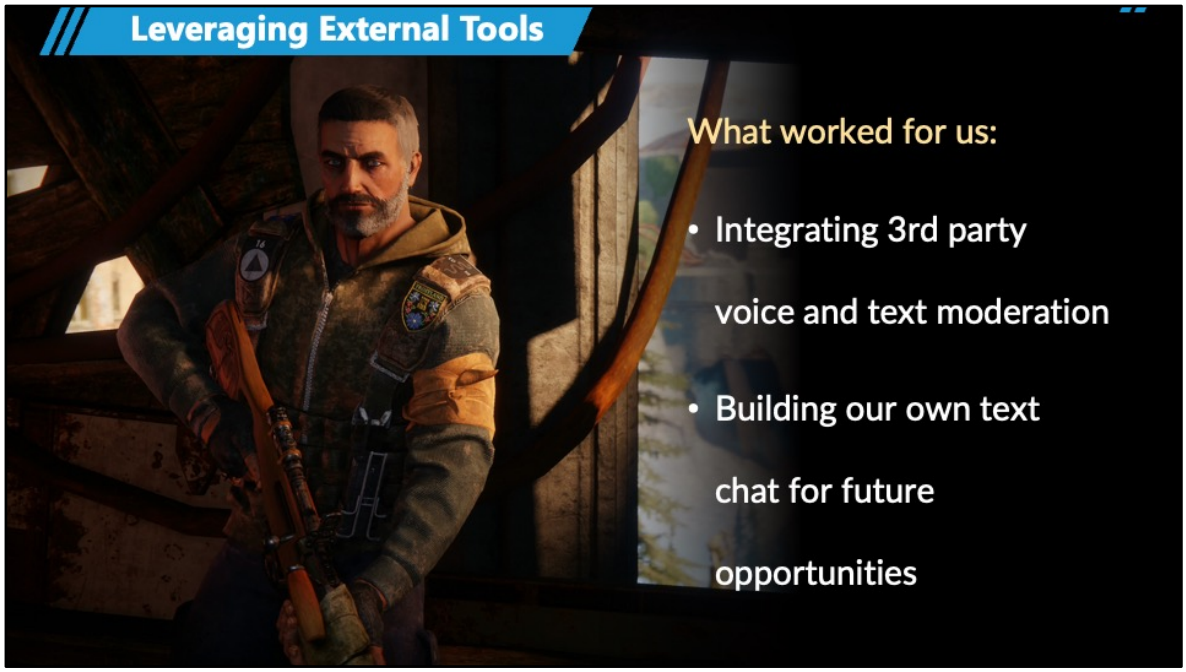
On the text filtering side, there's many different technologies available like regexs or flat lists. Some vendors offer Machine Learning solutions that are contextually aware. Being able to distinguish the difference between someone saying “You are a f***ing idiot” as bullying versus someone saying “This game is f***ing awesome” as positive sentiment. Building that ourselves would be difficult to train up a model for that and require a lot of language expertise in terms of staffing.

Other things to consider include

- Making sure any filtering solution supported all the languages we needed.
- Figuring out what moderation tools are available for repeated offenders and how it compares to what we have and how it could aid our player support teams.

All these considerations were made to combat toxic social behavior!

Being highly aware of what you need for your game, and making sure a solution fully addresses those needs is key to making these decisions. If something fits right, it’s a great candidate to use to save time from having to build from scratch.



So what did we end up doing?

For us, we ended up integrating a 3rd party a voice chat solution. To save on maintenance overhead if we built it ourselves, we found an external solution to handle voice. This allowed us to direct our attention to landing other parts of the Cross Play experience.

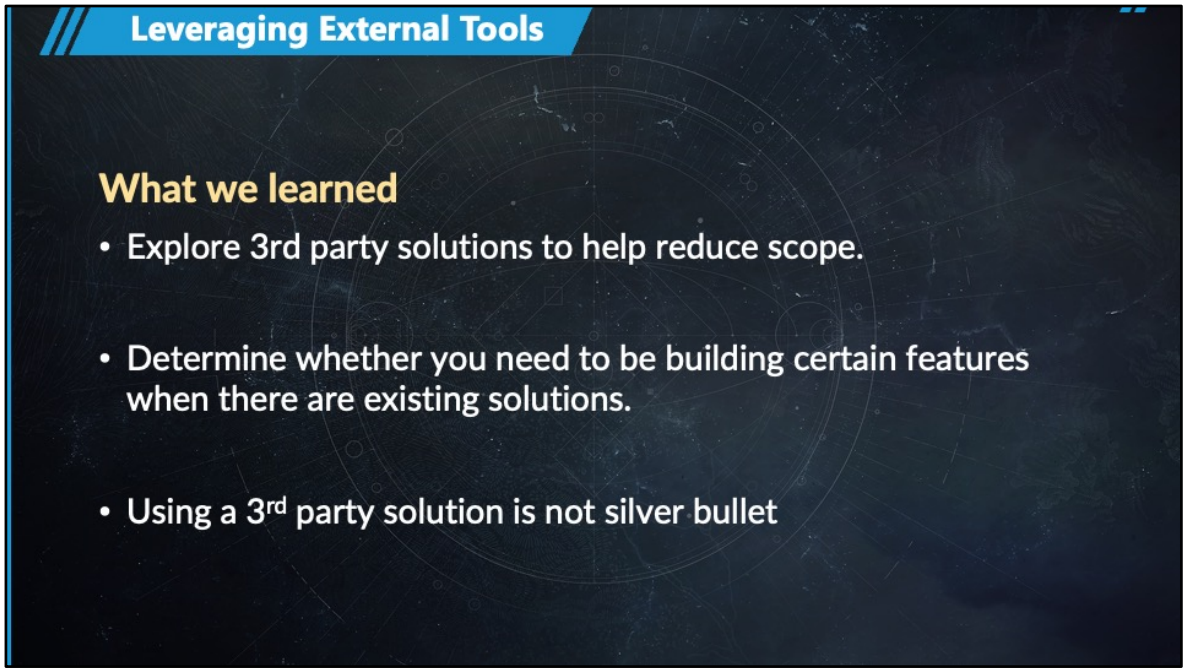
For text filtering and moderation, the massive amount of work it would take to build a system to support so many languages led us to integrate also a 3rd party.

It's important to call out that in our experience, it did take a significant amount of time to integrate both these solutions. Choosing to buy a solution requires engineering work to get it working with the game, and benefits are not instant.

Additionally when you use a 3rd party, you are effectively tying your game to a product. The releases and updates to that solution are dependent on someone else that you can't control.

Lastly for text chat, while there were solutions out there that accomplished what we

needed, we felt that being able to control our own text chat platform upon which we could build deeper integrations into the game in the future offered much more potential. So we built our own Text chat service.



What we ended up building ourselves or buying was unique to our game and needs. When faced with a decisions like these, it's important to consider:

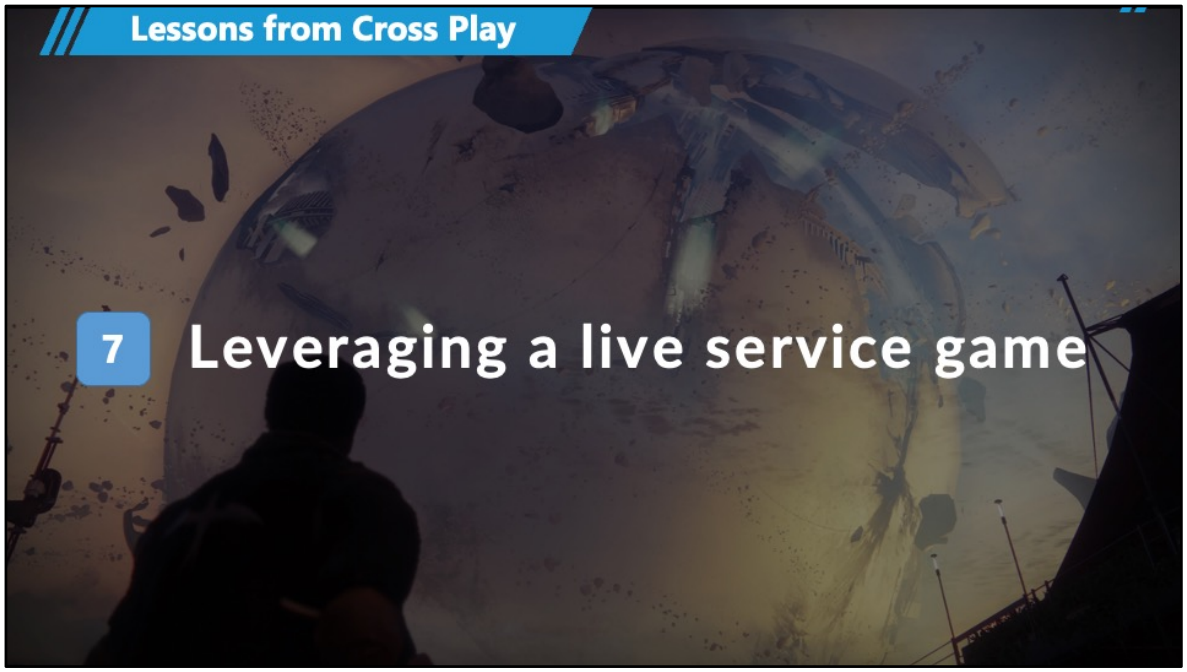
What features are fundamental to the experience you're building. Identifying what was within scope to land made it clear what areas we should be exploring 3rd party partnerships.

Whether your team actually needs to be building certain features. With so many pre-existing solutions are out there, knowing what needs to be built in-house is important for determining scope. Figuring out if you are building a certain system for reasons like maintaining a competitive advantage will help adjust your priorities.

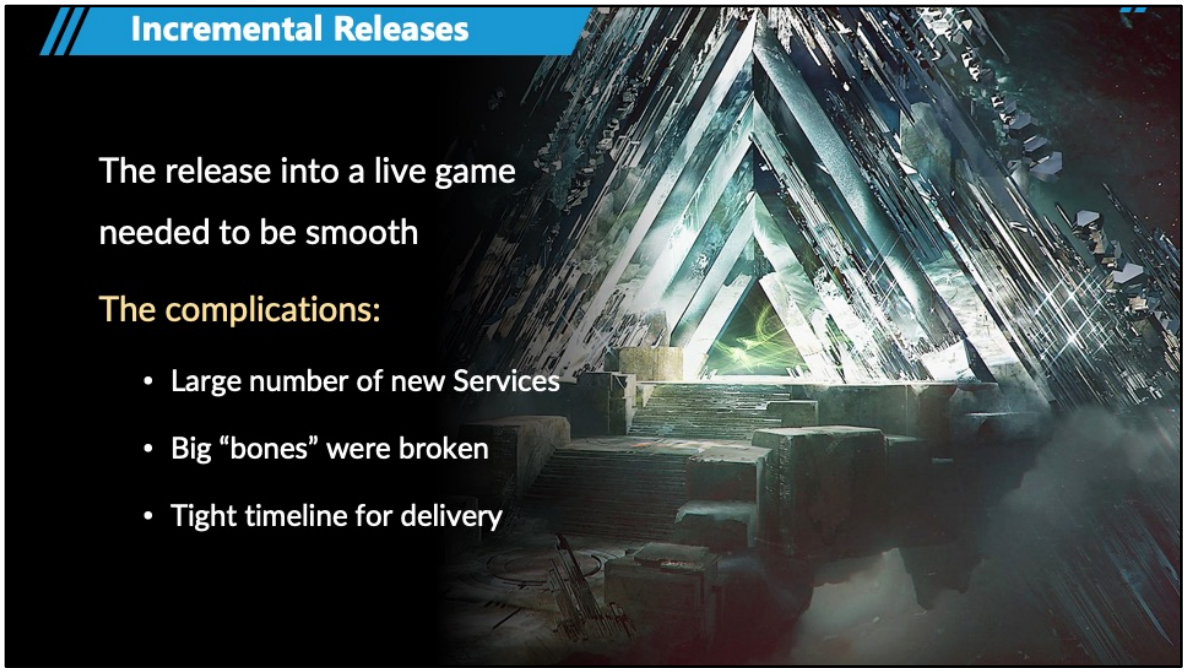
And lastly, one important note is that integrating a 3rd party solution is not silver bullet. Adding any kind of new tech always takes time and won't be done instantly. This needs to be considered in your planning and timelines.

In our case, we did end up being over-scoped with landing these integrations had had to slip them to later releases. But from a longer-term outlook, one of the things we saved on was maintenance and upkeep of homebrewed systems.

It’s the classic iron triangle of production. Scope vs Time vs Cost. Looking into 3rd party options may help find solutions when faced with finding a balance.



And now, the last lesson we learned from building Cross Play. And in this case it's about how we went about releasing it, taking advantage of Destiny 2 as a live service game.



Cross Play was built for an existing game. So the release needed to not disrupt our players.

But there were a lot of moving parts to this project and it was a massive thing to bring to the game. We had created the most amount of new Services since original Destiny 2 launched and broke a lot of “bones” like networking to make it possible to for platforms to play with each other.

And this was all under a tight timeline to deliver this huge experience. There were big concerns about how it would all come together in a single release, and that posed a huge risk. If we chose to release in a single season. we’d have only one try to get it right.

So what did we do?

We spread out the release of many of the Cross Play systems over multiple releases and seasons, incrementally adding more functionality as we went along.

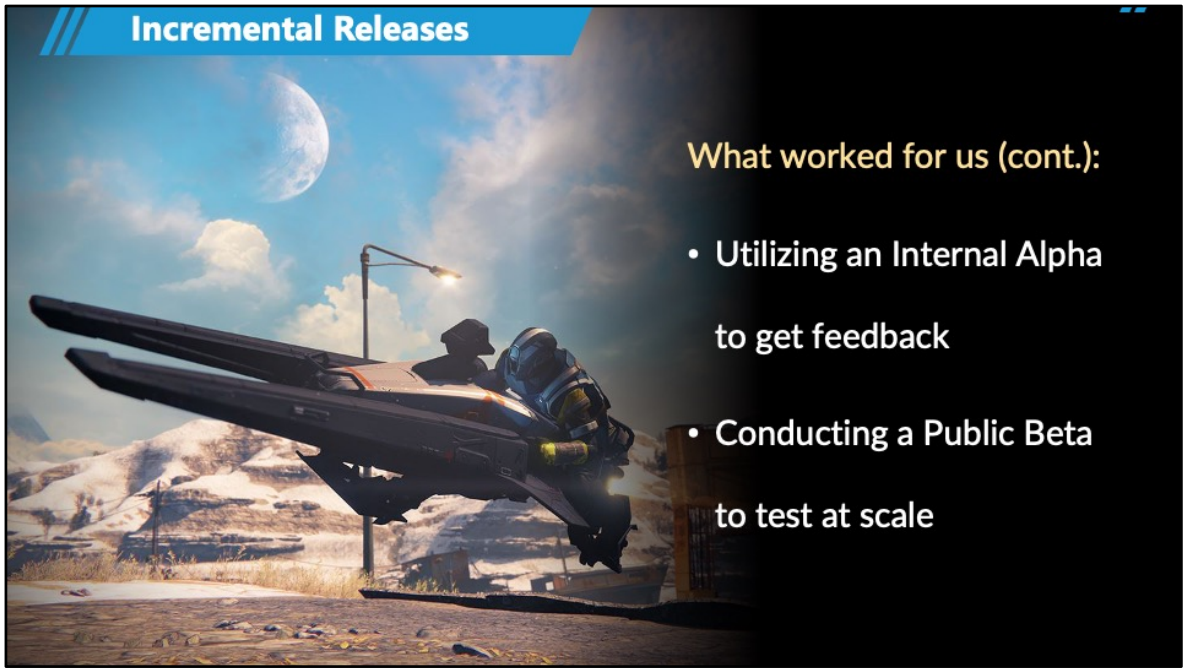
Since there were so many big changes coming in, we wanted to have more time to get them in, let them bake and stabilize. After which they could be tested at scale with the live population.

We embraced this method of incremental building and releasing of Cross Play.

We scaled from small 10-person internal playtests to just make sure things weren't crashing to later a studio-wide playtest of ~500 people. Doing this at an increased scale allowed us to test all the tech changes across all the platforms and on lots of content.

When it came to releasing to the live game, we used a earlier seasons to introduce big tech pieces. We front-loaded the work to get big bone-breaking changes like new networking changes in early. This allowed us to get these features checked-into builds very early in a development cycle, so we had plenty of time to test and stabilize.

Having these big changes out of the way early and stabilized helped give us more confidence to tackle the other Cross Play systems.



We also used an earlier season to introduce new UI and some of the systems being built for Cross Play to start getting feedback and live testing.

In season ahead of our launch, we created an Internal Alpha for Bungie employees to start using Cross Play. In the live retail game, employees were given access to additional features like the new Roster, and Bungie Friends list. Employees were actually Cross Playing with the public months before we actually released.

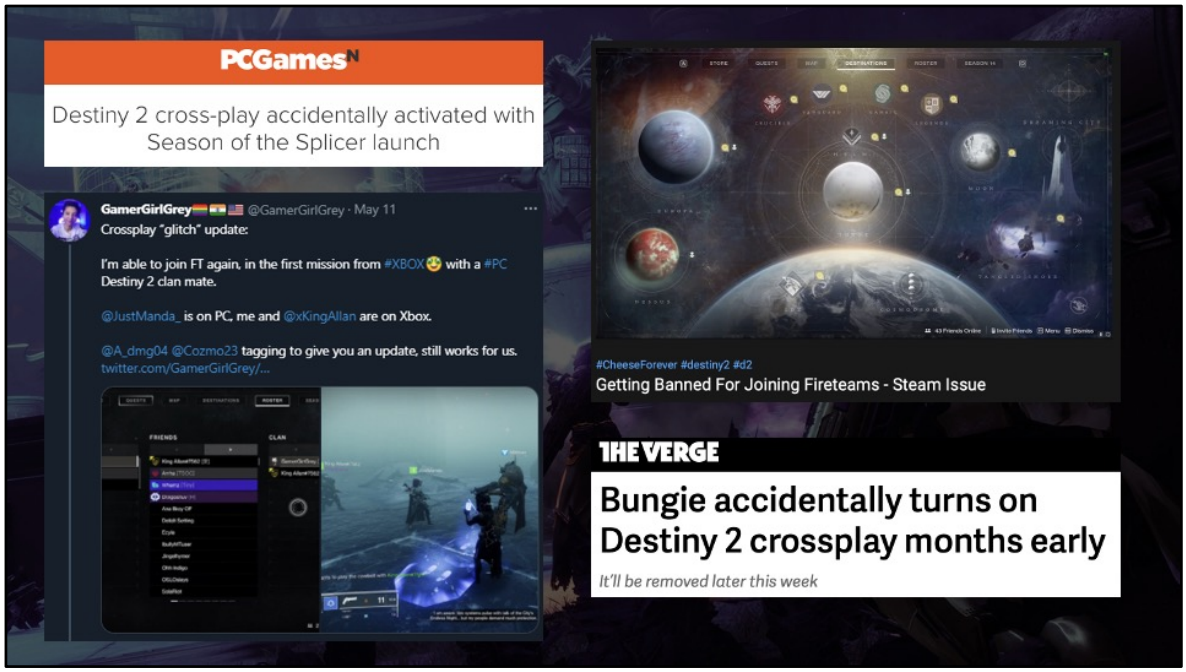
This helped us find some initial experience bugs to fix before any players tried using Cross Play. It really gave the team a better sense of how Cross Play would “feel”.

To really stress test our new networking and matchmaking tech, we conducted a limited live Beta for all players to participate in and try out Cross Play.

The high amount of participation helped test out and exercise these new code paths we were creating.

Between the Internal Alpha and Public Beta, we were able to gain more confidence in what we were building. Committing to early releases and landing the components necessary affirmed our timelines and helped us finish building Cross Play strong.

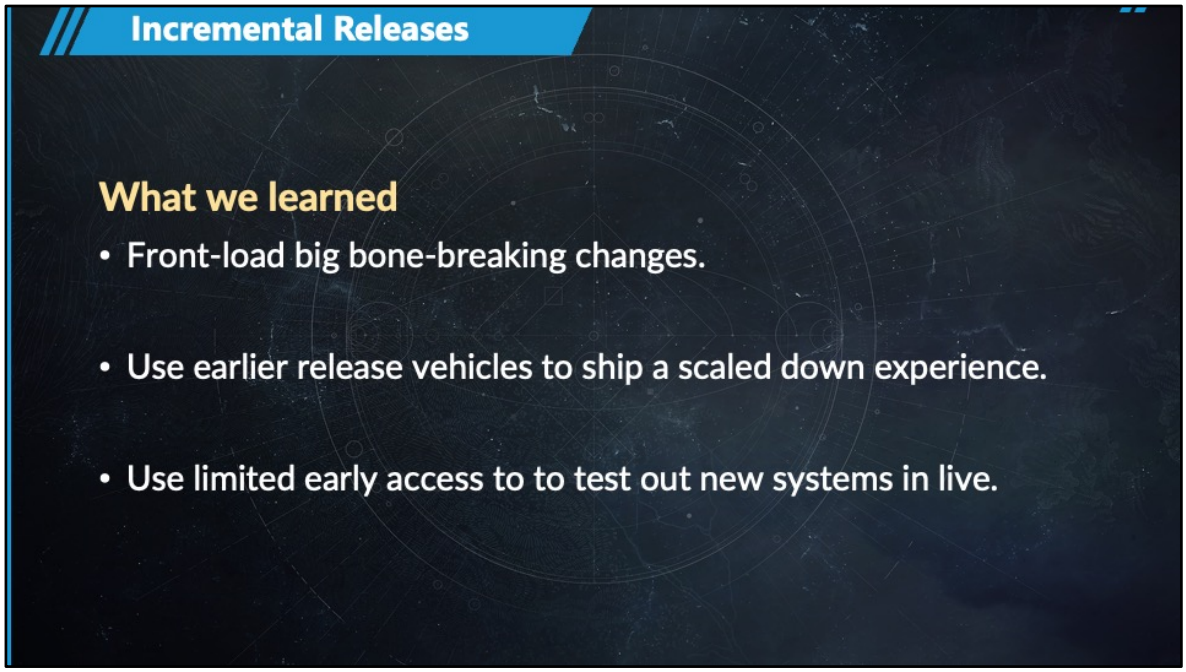
Also, as an added plus it helped build some buzz and excitement around Cross Play!



And while we saw the Internal Alpha and Public Betas early release as net positives. It wasn't without some bloopers. It turns out, having parts of Cross Play available in the game earlier than release made for some funny occasions where players stumbled into Cross Play before we intended.

When you're adding such a large feature into a live game, you have to understand that sometimes, unexpected things happen.

Overall, we saw these as kind of funny occurrences that exposed some bugs. These early accidents just continued to get Cross Play on the mind of our players and build more excitement for the real thing.

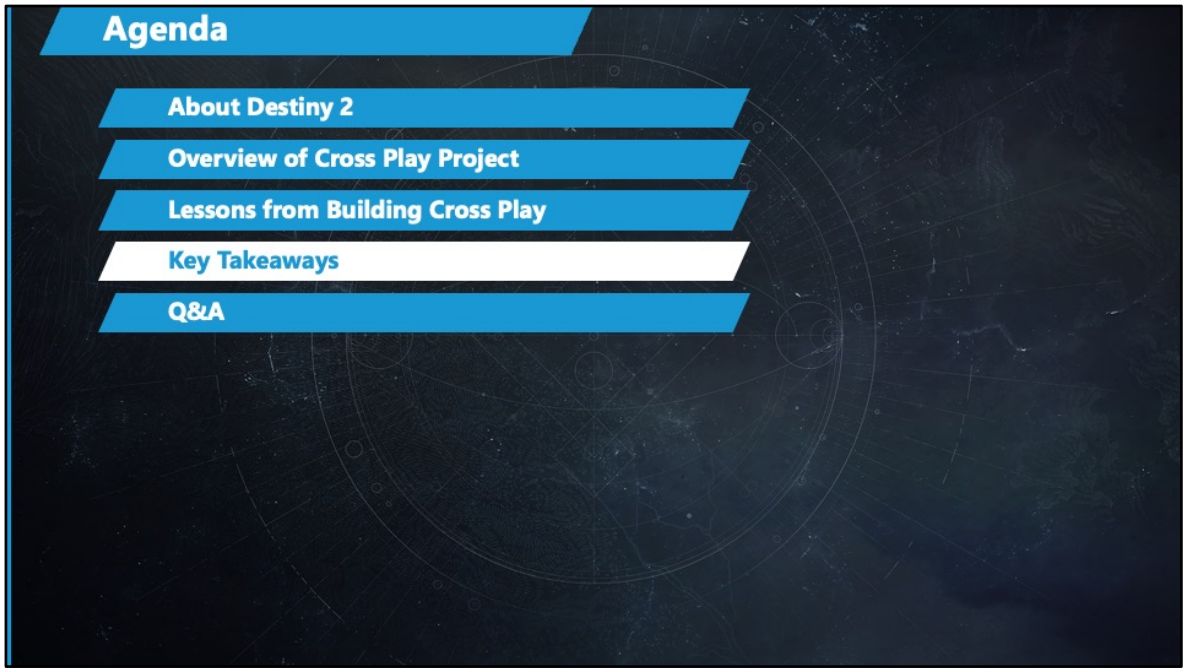


Using the live game to ship parts of Cross Play early really helped us ensure a smooth real launch.

By front-loading the big bone-breaking changes during your development, you can introduce high-risk changes to your game earlier to have more time to stabilize them.

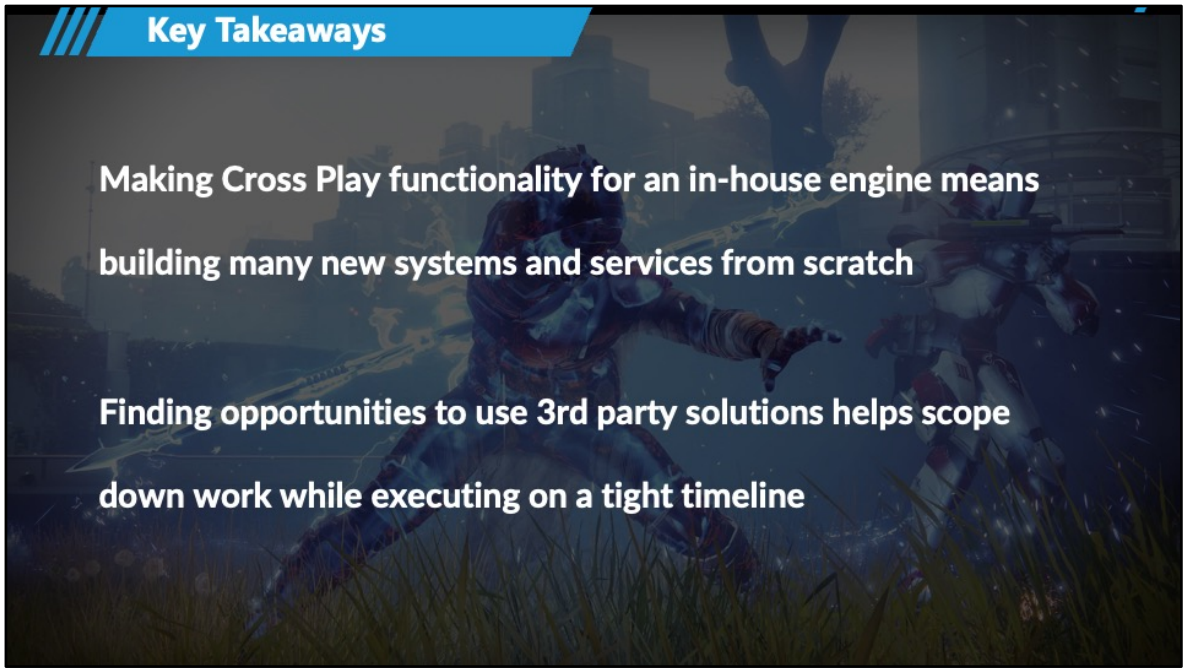
Having a scaled down experience available in earlier release vehicles helps you build confidence in what you are building and offers an avenue for early feedback to improve for a full release version.

And lastly, giving internal employees early access to cross-platform play with real players is a useful mechanism. It enables you to test out new systems in the live game that might not be public-ready, without the risk of exposing all your players to a lower quality experience.

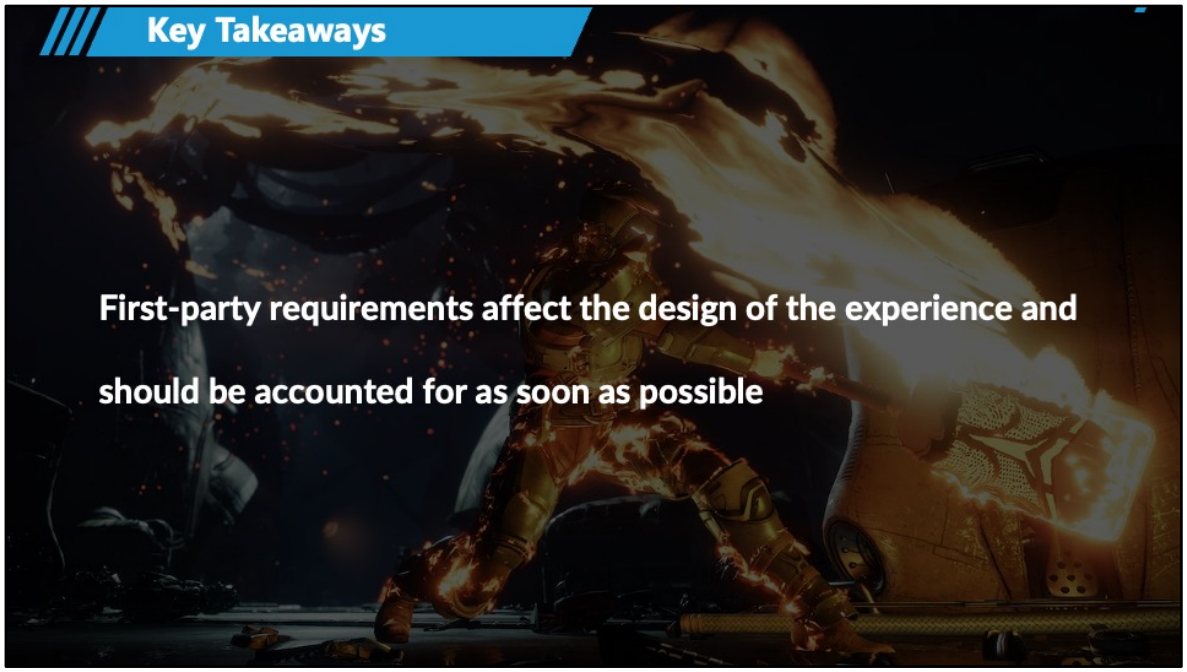


And with that, those were the 7 lessons we learned from building Cross Play! There was a lot that was covered because it was a long journey for the team!

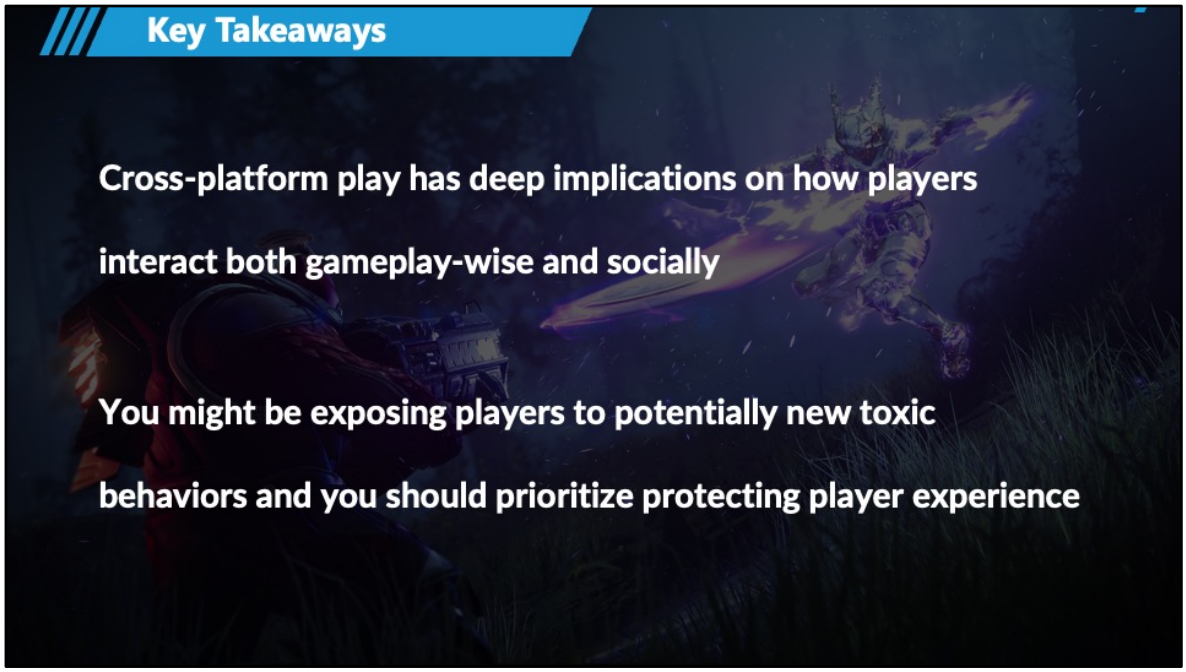
So if you're looking to build your own cross-platform experience for you game, here's the key takeaways I think would be most valuable for you and your team:



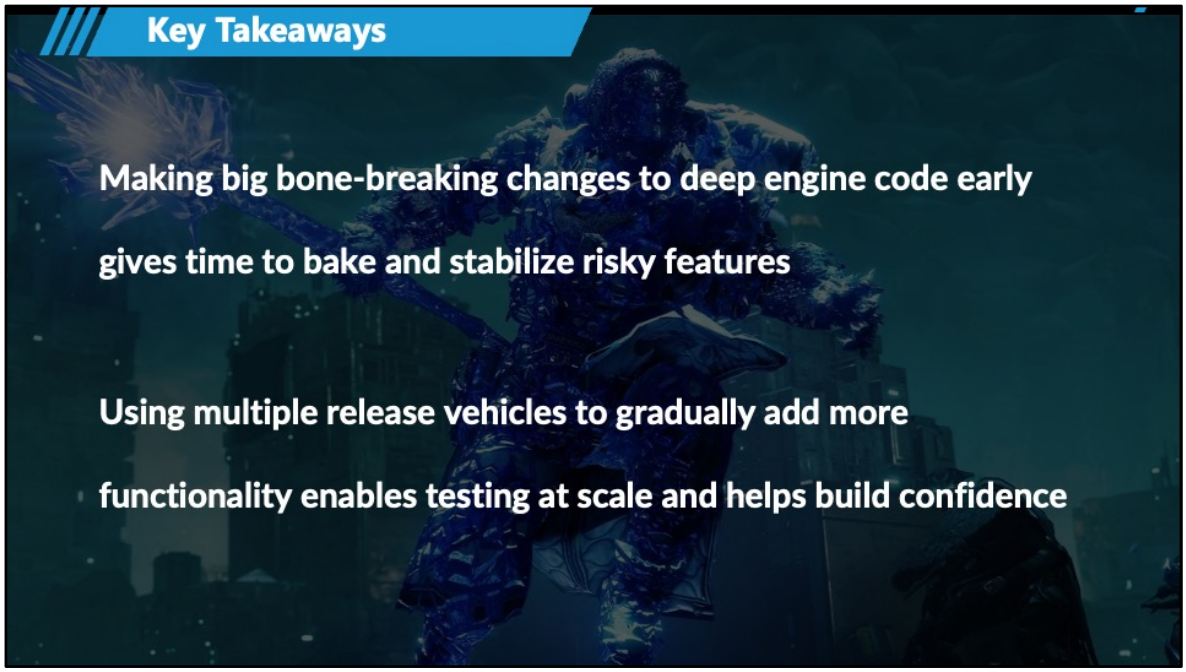
- Making Cross Play can mean building many new systems and services especailly if you rely on platform APIs or use your our own engine that might not support cross-platform
- While many new functionalities were hand-rolled internally, finding opportunities to use 3rd party solutions helps you scope down while executing on a tight timeline



- First-party requirements affect the design of the experience and from the very start of your planning, you should be taking these into account



- Cross-platform play in a game with established behaviors has deep implications on how players interact both gameplay-wise and socially
- And sometimes that means exposing players to potentially new toxic players and behaviors. This is a risk that you take on, and you need to ensure you are properly protecting your players.



- It’s good to front-load big bone-breaking changes to deep engine code so that you have plenty of time to bake and stabilize risky features
- If possible, using multiple release vehicles to gradually add more functionality to Cross Play allows you to scale out testing with a live population and in general helps build confidence in the project coming together

And that’s all that we learned from building Cross Play.

Thank you for listening!

Hopefully, you're able to take some of these lessons and apply them to your game.

As a note of thanks, this talk was only possible because so many people worked so hard to make Cross Play happen. Special thanks goes out to all the Bungos who worked on Cross Play and made such an amazing experience for players.

Also, I'd like to thank Veronica Peshterianu for being an amazing advisor and guiding me through the talk process.

Also thanks to Caitlin Yu who shaped a lot of the format and content for this talk.

And Lastly, thank you to the CAs and Event Support staff making sure GDC runs smoothly.

Also, we're hiring!

If you're interested in working at Bungie, we have many openings across disciplines.

Check out our careers page and follow us on Twitter @BungieCareers!

Thank you again. I've been Jon Chu, and now for some Q&A!