



Dedicated IP

EXPRESSVPN'S UNIQUE ENGINEERING DESIGN

Overview

Although the assigning of new IP addresses is central to the VPN experience, the way in which these addresses are traditionally managed doesn't suit everyone. Rotating IP addresses can lead to frequent authentication challenges, unreliable access to geo-restricted services, and increased exposure to tracking and security breaches. These issues highlight the need for alternative solutions for users who require a static IP address, while still maintaining a high level of privacy and security.

ExpressVPN's Dedicated IP (DIP) solution solves these problems with technologies designed to enhance user privacy and security, without compromising on convenience. ExpressVPN's DIP service assigns a unique IP address exclusively to a single user, providing a consistent online identity that is not shared with other users, while maintaining the user's privacy. DIP offers secure logins and seamless access to IP address-sensitive sites and reduces the frequency of verification challenges.

The following innovations lie at the heart of ExpressVPN's DIP solution:

Zero-knowledge IP allocation: Advanced cryptographic techniques ensure IP addresses are allocated privately. Even ExpressVPN administrators cannot identify which IP is assigned to which user.

Blinded tokens: Blinded tokens bolster individual privacy and security, verifying user actions without revealing the user's identity.

AWS Nitro Enclaves: Leveraging AWS Nitro Enclaves to use dedicated CPU cores and memory isolation, managing sensitive data.

Provable transparency and privacy: Providing means for third parties to verify ExpressVPN's privacy claims while preserving operational integrity.

Enhanced user experience: Making installation and setup easy for non-technical users.

Alternative DIP solutions that exist today may present security and privacy vulnerabilities that potentially expose users' true IP addresses. This makes it easy for attackers to correlate them with user identities and track their activities, posing significant risks.

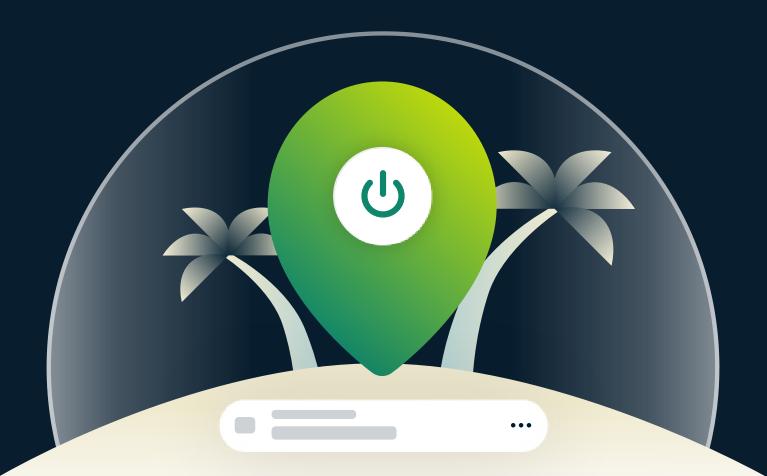
ExpressVPN's DIP solution overcomes the limitations of traditional DIP implementations and delivers an enhanced user experience with unmatched privacy and security.

Contents

Overview	2
What is Dedicated IP?	4
Benefits of dedicated IP	5
Challenges of offering DIP	6
Tracking	6
Anonymizing payments	6
User experience	7
DIP done right	7
What does our DIP achieve?	8
Provable transparency	8
Zero-knowledge IP allocation	8
What zero-knowledge means	9
How we designed secure DIP	10
JSON Web Tokens	10
Subscription Receipt Tokens (SRT)	11
Dedicated IP Access Tokens (DAT)	11
Assigning a DIP: Initial approach	12
Assigning a DIP with blinded tokens	13
Refreshing the DAT	15
Innovating with AWS Nitro Enclaves	16
Refreshing the DAT: Process	17
Secure communication with the Enclave	17
Preventing chargeback and refund abuse	21
Introducing a Dedicated IP Refresh Token (DRT)	21
Using your DIP on an additional device	23
Revoking DRTs	24
Implementing the Subscription Version ID	24
Known limitations and future improvement	25

What is Dedicated IP?

A Dedicated IP (DIP) is an IP address assigned exclusively to a single user that may be used simultaneously on multiple devices. Unlike shared IP addresses—typically offered by VPN services—DIP provides a stable and consistent online presence, ideal for managing access to restricted networks, supporting high-security identity and access solutions, and avoiding the need for continual authentication requests like CAPTCHA challenges.



BENEFITS OF DEDICATED IP

When a user includes DIP with their VPN subscription, the user can assign that DIP to some or all of their devices at the same time. This offers all the benefits of a VPN service, with the added flexibility of a stable, unchanging IP address when needed. Here are a few scenarios where a user might benefit from a DIP:

Accessing restricted networks: Many organizations implement IP whitelisting to limit access to important networks. A DIP gives the user an address that the organization can add to its whitelist. The user and the organization know that the IP will not suddenly change (thus locking out the user and adding overhead for the organization while it takes steps to restore access). They also know that the IP is only ever going to be used by that user. A frequently changing IP address (generally a benefit) is a common source of problems and often forces the user to turn off their VPN if they want to connect to their organization. With ExpressVPN's DIP solution, the user no longer has to choose between privacy and accessing the network.

Managing secure logins: Some online services, such as online banking, corporate VPNs, and remote access systems, require the user to maintain heightened security on their device. Those systems often consider a changing IP as a sign of elevated risk, which can lead to problems if the user accesses the system through a traditional VPN. ExpressVPN's DIP solution addresses this problem by ensuring the user is always connecting not only from the same country and region but also from the same IP address—even if traveling or using different Wi-Fi networks. This significantly reduces the likelihood of triggering any IP-based security alerts.

Avoiding CAPTCHA Requests: CAPTCHA requests present inconveniences for users, particularly those employing traditional VPN solutions with frequently changing IP addresses. Such users experience an increased likelihood of encountering CAPTCHA challenges. DIP addresses this issue by providing users with a consistent and trusted IP address. Exclusive use of a single IP by one user allows servers to establish prolonged trust in that IP. Consequently, this approach significantly reduces the frequency of CAPTCHA challenges or potentially eliminates their occurrence.

Challenges of offering DIP

VPN services face various challenges if they want to offer dedicated IPs.

1. TRACKING

DIPs present inherent privacy challenges. While DIPs provide users with a consistent internet presence, enhancing navigation and user experience, they also make user tracking easier. Websites and ad networks can correlate visitor IP addresses with user login records—a practice that has emerged again due to recent restrictions against cookies by regulators and browsers.

In scenarios where a DIP is most beneficial, this tracking may not be a significant concern for users. However, for activities where privacy remains a concern, such as general browsing or other privacy-sensitive activities, the traditional approach to IP addresses remains more suitable. Hence, DIP is best utilized as a complementary tool alongside traditional VPN services from ExpressVPN.

2. ANONYMIZING PAYMENTS

Some DIP implementations prioritize anonymizing payment and decoupling it from IP address assignment. While this simplifies implementation, it also prevents features like a money-back guarantee. Without a mechanism to associate payments with specific IP addresses, providers cannot effectively prevent abuse or manage chargeback fraud.

Conversely, linking payments to IP assignments raises privacy concerns, as users could potentially be identified through their assigned DIPs. Resolving this paradox is crucial for offering a high-quality service that balances user privacy with fraud prevention and abuse resistance.



3. USER EXPERIENCE

It's difficult to offer user-friendly setup processes while maintaining robust privacy and security measures. While highly secure systems can be developed, they often present complex user interfaces that may intimidate non-technical users, potentially leading to frustration or, more critically, misconfiguration. Furthermore, DIP solutions that are not meticulously designed or properly implemented with privacy as a core consideration may function adequately but potentially compromise user privacy.

DIP DONE RIGHT

While DIP offers numerous advantages, the inherent risks must be addressed. This was the challenge we aimed to solve, with the end goal of developing a solution that not only meets the needs of our users but also maintains our strict standards of security and privacy.

Through a rigorous analysis of existing DIP solutions and a comprehensive evaluation of various components, ExpressVPN is proud to unveil our own DIP solution, specifically designed to mitigate the risks associated with secure implementation.

This white paper details the engineering design and implementation processes, for greater trust and transparency in our approach.



What does our DIP achieve?

To solve these challenges while maintaining security and usability, it was imperative to design a solution that is simple to set up and accessible to non-technical users. To ensure robust technical security and privacy controls, we introduced provable transparency, provable privacy, and zero-knowledge IP allocation, while also leveraging advanced cryptography—including the adoption of Amazon's innovative secure compute environment, Nitro Enclaves. These technologies combine to help ensure user activities are confidential and private.

Underpinning all of this is ExpressVPN's platform, powered by TrustedServer and Lightway. Our DIP infrastructure, like all components of ExpressVPN, is continuously overseen and secured, ensuring that a user's experience remains as safe as any other aspect of our service.

Ø PROVABLE TRANSPARENCY

Provable transparency is an important design tenet that allows third parties to independently verify that our service functions as described, assuring trust and accountability. Through public attestation, ExpressVPN has built a transparent system where every action and process can be independently audited, without compromising the user's identity.

ZERO-KNOWLEDGE IP ALLOCATION

Zero-knowledge IP allocation allows ExpressVPN to assign the DIP to a user without knowing which user. This assignment ensures the user's privacy is preserved both on allocation and on renewal.





WHAT ZERO-KNOWLEDGE MEANS

Here is how we categorize zero-knowledge and look at how zero-knowledge can be achieved for DIP.

Separating customers from addresses

A fundamental design principle is that no untrusted component ever sees customer information, such as the username or subscription ID (which is associated with payment information), or logins with DIP reservations.

This ensures that, even in the case of bugs, misconfigurations, or similar circumstances, this sensitive association is not stored in error logs or elsewhere.

Trusted vs. untrusted components

Everything the user cannot control or verify themselves is considered untrusted. We considered our apps that run on the user's device to be trusted, as they are in the open and can be examined. We consider our own API servers untrusted if we cannot prove to the user that they are running what we claim they are running. As changes to the APIs cannot be attested to or verified by the public, we believe each of these should be considered a risk as part of the security model.

However, some of these components that we marked as at risk are needed to make the DIP solution possible. It was thus necessary to find a way to make these components trusted so that we could keep the user safe.



How we designed secure DIP

ExpressVPN prioritizes transparency and openness as fundamental elements of trust and security. This section provides an in-depth examination of our DIP solution's architecture, to provide a comprehensive understanding of how we assure unparalleled security and privacy. We begin with one of our initial decisions: the use of JSON Web Tokens as the authorization mechanism. Following this, we will detail each stage of the DIP address allocation process to illustrate how the components interconnect.

🥮 JSON WEB T<u>okens</u>

JSON Web Tokens (JWT) are a compact means of representing claims that can be transferred between two parties. Tokens are essentially JSON objects containing attributes known as "claims" that are accompanied by a header and a cryptographic signature.

The signature is important as it makes it impossible for an attacker to tamper with the contents or to generate a fake token if they don't have access to the private key. Anyone with the public key can verify the authenticity of a token.

ExpressVPN uses JWTs for authentication, and once a user signs in, each subsequent request includes a JWT, allowing the user to access networks, services, and resources that permit that token. JWTs are the perfect way to securely transmit information between parties because the digital signature assures both the integrity and authenticity of the exchange.

📼 SUBSCRIPTION RECEIPT TOKENS (SRT)

One of the foundational components in our solution is the Subscription Receipt Token (SRT). Like Apple's in-app purchase receipts, the SRT validates an active paid subscription. This token includes a sub-claim referencing the customer's subscription ID and an entitlement claim that lists the features they have an entitlement for, such as "xv.vpn.dip" for Dedicated IP.

Unlike the subscription itself, which may have a longer validity period, the SRT is renewed every three days to ensure it is continually (and securely) verified. If a subscription is canceled prematurely, for example, due to a refund or chargeback request, the SRT expires within 72 hours, effectively revoking access to that service.

An SRT alone doesn't fully authorize a DIP VPN connection because it lacks a direct reference to the user's DIP. Including the IP address in the SRT violates our principle of keeping customer information separate from IP details, since one of our design tenets is that untrusted components, like the VPN server, cannot access both simultaneously.

To meet this requirement, we created another type of token, the Dedicated IP Access Token (DAT).

DEDICATED IP ACCESS TOKENS (DAT)

The DAT is the user's entry ticket into the VPN connection. It contains no reference to the subscription or the customer, but it does include a claim for the DIP it authorizes, which looks like this:

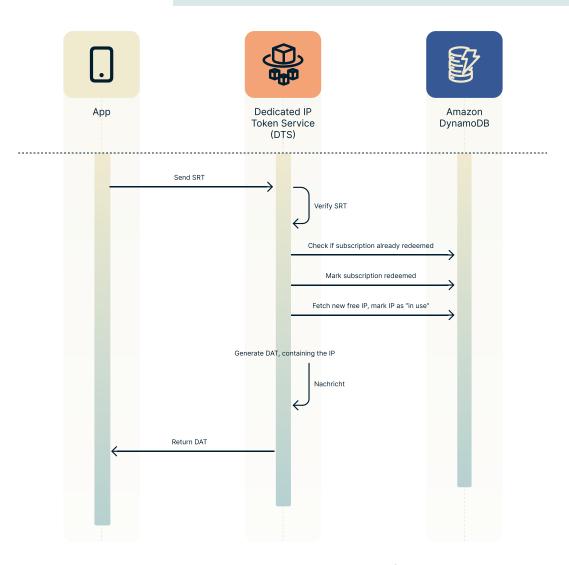
```
"xv.vpn.dip.details": {
"ip": "201.34.5.6"
}
```

When a VPN connection is being established, the VPN server checks this claim to ensure it matches the server's DIP. The expiration of a DAT is managed carefully. It does not directly reference the subscription but must still be renewed regularly. If a subscription ends prematurely, the token cannot be automatically revoked, so it expires and is not renewed.

ASSIGNING A DIP: INITIAL APPROACH

Before issuing a DIP Access Token, the system assigns a free DIP reservation and ensures it is unique. Here's the workflow for ExpressVPN's approach to assigning the initial DIP:

- 1. The customer makes a payment, and the app receives an SRT.
- 2. The app sends the SRT to the DIP Service, requesting a new DIP reservation.
- The DIP Service verifies the SRT and assigns a new DIP, marking the subscription redeemed.
- 4. The DIP Service generates the DAT and returns it to app.



While this might sound straightforward, the approach requires our DIP Service to access subscription IDs and track redemptions, which contradicts our other two design tenets—zero knowledge IP allocation and the principle of provable transparency and privacy. Therefore, we needed a way to anonymize the subscription ID while still verifying redemption status.

We adopted a new approach to solve this problem using blinded tokens.

Assigning a DIP with blinded tokens

To solve the dilemma of anonymizing the subscription ID while still verifying redemption status, we use a technology called blinded tokens. Blinded tokens allow our solution to issue tokens without revealing the recipient's identity, like spending cash instead of using a credit card.

Here's a high-level overview of how blinded tokens work:

Blinded tokens are tokens issued by a server without the server knowing who the token is assigned to, while still allowing the token to be redeemed. They function like "cash," which can be spent without revealing your identity. The process involves some cryptographic techniques, which can be explained this way:

- Imagine a random number, let's say "1476465," on a piece of paper. The chance that someone else will generate the same number needs to be extremely small, so this number should contain many figures.
- Next, you place this number inside an envelope, effectively "blinding" the number. This envelope is also lined with carbon paper.
- 3. You then present the sealed envelope, along with your subscription receipt, to our Subscription Receipt Token Service. The service verifies that your subscription is valid and has not yet redeemed any DIP, without being able to see inside the envelope. It then signs the envelope on the outside. The carbon paper transfers the signature to the paper inside, thereby "blindly" signing your number.
- 4. When you open the envelope, you retrieve the signed number, which is now "unblinded."
- 5. You then send this signed number to the DIP Token Service. The service can verify the authenticity of the signature without knowing your identity. Since the number is random and was never seen by any of our systems when you provided your proof of purchase, your anonymity is preserved.



There is an extremely low probability (almost zero) of a collision occurring, where a random number is generated that has already been issued. If this happens, the user would find themselves with a subscription marked as redeemed, but without having spent a blinded token. In such cases, an error message should prompt the user to contact Support, who can then reset the subscription status to "not redeemed."

Since we assure the user's security and anonymity with blinded tokens, we could leverage the same approach to improve the DIP assignment.

Here is the revised approach including blinded tokens in the process:

- 1. The customer makes a payment, and the app receives an SRT.
- 2. The app sends the SRT, along with a blinded token, to our Blinded Token Service.
- 3. The Blinded Token Service verifies the SRT and signs the blinded token, flagging the subscription as redeemed.
- 4. The app unblinds the token and sends it to the DIP Service, requesting a new DIP reservation for that user.
- 5. The DIP Service verifies the signed blinded token, marks it as redeemed, and issues a DAT.

Refreshing the DAT

By adopting blinded tokens through this assignment process, we provide the anonymity buffer that ensures ExpressVPN's DIP solution remains unaware of which customer ID received the DAT. However, this is not the final step; as previously mentioned, DAT must be refreshed every three days.

To achieve this while maintaining the same level of assurance, it was determined that the best approach is for the user (or the application) to keep proving the subscription's validity. The streamlined process is as follows:

- When a DAT is close to expiring, the app sends the DAT and a fresh SRT to the Dedicated IP Service.
- The Dedicated IP Service verifies the SRT without accessing the user's subscription details, maintaining the separation of customer and IP data.

This brings back the same issue encountered during the assignment phase, where the DAT contains the IP, the SRT, and the subscription ID— potentially classifying our DIP service as an untrusted component.

Enter AWS Nitro Enclaves. AWS Nitro Enclaves are this trusted component, and they are the game-changers at the core of our solution.





Amazon Web Services (AWS) offers Nitro Enclaves, running virtual machines on regular EC2 instances. The power of an Enclave lies in its completely isolated environment, protected by the AWS Nitro Hypervisor. Unlike standard virtual machines, Enclaves don't have permanent storage or even network access. They can only communicate through a special dedicated interface known as VSOCK. Critically, the AWS Nitro Hypervisor can generate an attestation proof showing which image is powering the virtual machine. This allows anyone to verify that the Enclave is running what it claims to be running.

ExpressVPN leverages the security of Nitro Enclaves to generate our Dedicated IP Access Tokens (DAT). By making our Enclave code opensource, we have allowed everyone access to verify that the Enclave isn't keeping track of any associations between customers and DIPs. This transparency, combined with the ability to check the attestation and confirm that the claimed image is indeed the one running, allows us to consider this a trusted component in our architecture.

However, since an Enclave can't persistently store data or access external storage, the Enclave wouldn't be able to perform tasks like fetching the next free IP from a database. This is why our Dedicated IP Service undertakes these tasks, which is acceptable as long as we ensure that it never sees both the IP and subscription ID.

Given the approach of bringing AWS Nitro Enclaves into our architecture, we moved to implementing the following for both the assignment and the renewal flows:

- The app sends a request to the Dedicated IP Service, this time sending an encrypted SRT along with the signed unblinded token.
- 2. The DIP Service validates the signed unblinded token, fetching the next free IP address in the available pool.
- The DIP Service forwards the IP using the AWS Nitro Enclave VSOCK Communication Interface and the encrypted SRT to the Enclave to generate the DAT.
- 4. Lastly, the Enclave decrypts the SRT and verifies its validity.



The periodic refresh of the Dedicated IP Access Token (DAT) remains a necessary process. With the integration of AWS Nitro Enclaves, the new procedure is as follows:

- 1. The app sends a request to the DIP Service, this time sending an encrypted SRT with the DAT.
- The DIP Service forwards the request to the Enclave, since it can see the IP in the DAT, but it cannot see the subscription ID since the SRT is encrypted.
- 3. The AWS Nitro Enclave decrypts the SRT, verifies its validity and proceeds to issue the new DAT.
- 4. The DIP Service finally extends the reservation date for the IP in the DAT for the next refresh period (three days).

SECURE COMMUNICATION WITH THE ENCLAVE

Ensuring secure communication with the Enclave is crucial. The app encrypts the SRT so that only the Enclave can decrypt it, ensuring that the untrusted DIP Service cannot access its contents.

AWS Nitro Enclaves can generate a cryptographically signed Attestation Document, signed by the AWS Nitro Root CA. This document serves to prove which image the Enclave is currently running, but it also contains an additional public key. Each time the Enclave boots up, it generates a new public/private key pair.

Our apps fetch the Attestation Document, verify its authenticity, extract the public key, and use it to encrypt any data sent to the Enclave. The Enclave is thus the only one that can then decrypt this data using its private key, ensuring that sensitive information remains secure throughout the process.

Enclave attestation and the public/private key

To verify that our Enclave is running the code that we claim to run, the Enclave can ask the AWS Nitro System to issue a signed Attestation Document.

The "pcrs" field is important, as it contains the hash of the image the enclave is running.

The Attestation Document is fully defined here - <u>Attestation Document</u> <u>Definition</u>.

Note the Attestation Document contains some optional fields:

- public_key: user_data, ; an optional DER-encoded key the attestation consumer can use to encrypt data with
- user_data: user_data, ; additional signed user data, defined by protocol
- nonce: user_data, ; an optional cryptographic nonce provided by the attestation consumer as a proof of authenticity

Public key

When the Enclave is started for the first time, it generates its own public/ private key pair. As it is not possible to access the Enclave, we cannot access the confidential private key. However, we can obtain the public key as part of the Attestation Document. This key can be used for encrypting data, and it can also be used as a way to verify the authenticity of the Enclave (as only that specific Enclave will have the matching private key to decrypt data). We can thus leverage what is now an authenticated public key in the Attestation Document for secure communication between the outside world and the Enclave. This is required later on to establish a secure communication channel with the app and to prevent MITM attacks (including from us), as we generate and encrypt a session key using the attested public key.

Nonce

To prevent an old Attestation Document from being replayed, the client will generate a random nonce when asking the Enclave to deliver a "fresh" Attestation Document. The optional fields in the Attestation Document will then contain the nonce and will be signed by the Nitro Enclave Security Module, so that we're sure the Attestation Document is valid.

Why is the nonce required?

The nonce is very important, as it ensures that the response a user receives was generated specifically for their request. Without the nonce, they would have no way to know whether the response was generated for them, or someone else an hour ago.

This means they could be subject to a replay attack, which is where they receive a valid response, but one that wasn't necessarily created for them. In our security model, this would not directly impact their security, as they would still verify that it was an authentic Nitro Enclave running the correct software. However, by including the nonce and ensuring that it is verified as part of the Attestation Document, the user can rest assured that the response they get is real and was generated in response to their request.

VSOCK implementation

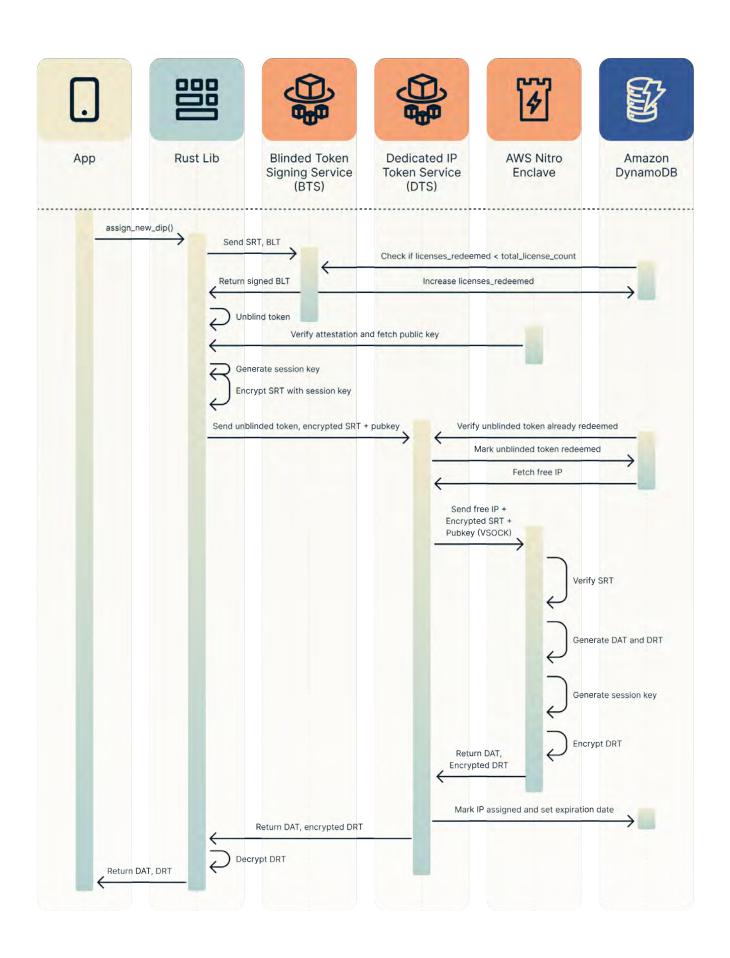
As mentioned above, the only way to communicate with the Enclave is via a VSOCK connection. The Enclave has no access to any other external resource—that is, it has no network access, and it has no disks. The EC2 parent instance acts as a proxy to forward specifically defined API requests (received via regular HTTPS) into the Enclave. Only the defined API interface is allowed in or out of the Enclave.

Encryption process

As asymmetric encryption is not very performant (and has other size limitations), we combine it with a symmetric algorithm: AES-256-GCM.

The symmetric key is negotiated directly between the app and the Enclave, utilizing the well-known Elliptic-Curve Diffie-Hellman (ECDH) Key Exchange protocol.

- The app will fetch an Attestation Document, providing a random nonce, verifies the Enclave's attestation, and then will extract the public key of the Enclave from the attestation. It then checks if the nonce in the Attestation Document matches the one it generated previously. This ensures that a "fresh" Attestation Document was generated.
- 2. The app will use ECDH to generate a session key.
- 3. The app will symmetrically encrypt the sensitive payload data using the session key.
- 4. The app will send the encrypted payload, along with its own public key, to the Enclave (via the Dedicated IP Service).
- 5. The Enclave can generate the same session key, using the public key of the client, and will decrypt and process the payload.
- 6. The Enclave generates the DAT and sends it back to the DIP Service.
- 7. The DIP Service extends the reservation of the IP.





Preventing chargeback and refund abuse

So far we've explored how our service can issue a new DAT, and how it can refresh a DAT and extend the reservation date for a DIP.

In our flow, we are sending the encrypted SRT and the old DAT to the Enclave to get refreshed. But what prevents someone from sending any arbitrary SRT along with a DAT? To abuse the system, you only need one SRT, and you can use it with any number of DATs to keep refreshing them.

What would this sort of attack look like? Well, you could buy 10 subscriptions, issue 10 DATs, store them somewhere safe, and then request a chargeback for 9 of them and keep only 1. You then use the SRT to keep refreshing all 10 of the DATs, effectively getting 9 free DIPs. It seems we need a way to prove that a DAT was issued using a specific SRT.

INTRODUCING A DEDICATED IP REFRESH TOKEN (DRT)

Thus, we introduce another token, very similar to DAT, but with the addition of the subscription ID. So this token essentially now contains the DIP and the subscription ID and as such is highly sensitive and needs to be encrypted as well.

But now that we have secure Enclaves, we can issue this token along with a DAT during the assignment phase. The Enclave can securely encrypt this token using the same method that the app uses to encrypt the SRT and then send it back to the app.

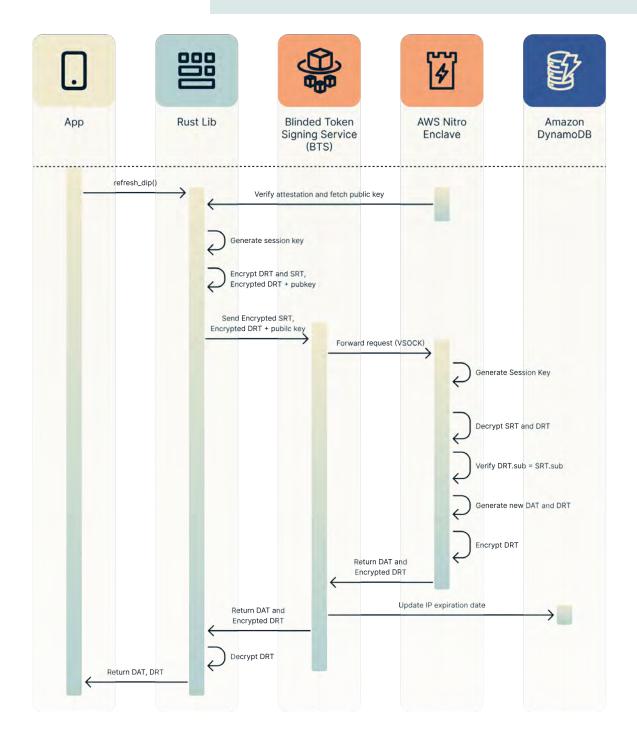
This now results in two tokens being issued when the IP assignment is first made: One, the Dedicated IP Access Token, which is used to establish the VPN connection; and two, the Dedicated IP Refresh Token, which is used to renew the DAT.

The Dedicated IP Refresh Token is ultimately the proof of continuous ownership of a DIP. If that token expires, the associated DIP reservation expires as well, and the user has lost ownership. To prevent this, the Dedicated IP Refresh Token needs to be extended as well, and it gets refreshed every time the DAT is refreshed.

This process requires the apps to actively initialize the refresh. If a user did not use the app for more than three days, the DAT and DRT would expire, and they would lose the DIP reservation. To prevent this, the lifetime of a DRT is 60 days instead of three. This allows a user to be inactive for up to 60 days, and only then would they lose the DIP reservation. Users can then contact our Support Team to get their subscription reset so that they can receive a new DIP.

Now, during the refresh flow, the app encrypts the DRT and sends it along with the encrypted SRT to the Enclave:

- 1. The app encrypts the DRT and SRT.
- 2. The app sends a request to the DIP Token Service.
- 3. The DIP Token Service forwards the request to the Enclave.
- 4. The Enclave decrypts the DRT and SRT and verifies whether the sub-claim of both tokens matches, the SRT is still valid, and the DRT is still valid, then issues a new DRT and DAT.



Using your DIP on an additional device

For users wanting to use DIP on an additional device, it is important to recognize that the DRT serves as proof of ownership and purchase.

To facilitate this, we will implement a mechanism for backing up and restoring the DRT on any additional device. Given the sensitivity of the DRT, applications must always encrypt it before transmission to our backup service.

We employ AES-256-GCM encryption, using a customer-provided backup password—referred to as an Access Code in the applications—to secure the DRT during each issuance or refresh.

Revoking DRTs

If customers forget their backup password and are unable to recover their Device Renewal Tokens (DRTs), our systems cannot retrieve them. In such instances, users must undergo the entire DIP assignment process once more.

To prevent another abuse vector, where customers "claim" they lost their password but remain in possession of the previous DRT, we should implement a way to "revoke" DRTs.

As previously noted, JWTs cannot be revoked, they can only expire. However, we could prevent a revoked DRT from being renewed, thereby minimizing the impact.

IMPLEMENTING THE SUBSCRIPTION VERSION ID

Our solution is that every subscription is considered to have a "version," starting from 0. If our Support Team resets a subscription, granting a customer an additional DIP license, this version number is increased.

This version number will be added to the SRT as an additional entitlement property in the xv.vpn.dip entitlement:

"xv.vpn.dip": {"did" : <version>}

The same version is put in the DRT. This links the DRT to a subscription ID and a specific version.

The next time a DAT/DRT refresh flow is triggered, the Enclave will additionally check whether the SRT's version number matches the version number of the DRT. If the subscription gets reset in the meantime and the version number does not match the one associated with the DRT, the DRT gets rejected.

Known limitations and future improvement

As outlined above, the DIP assignment process is designed to avoid associating a customer's identity with their DIP. No single component has access to both pieces of information simultaneously.

However, there is a potential risk of side-channel attacks by an internal attacker who could monitor database operations and infer relationships between subscription redemptions and IP allocations based on the timing of these operations. Additional metadata, such as the customer's IP address, could also be used to establish correlations.

To mitigate these attack vectors, we could introduce delays between the subscription redemption and IP allocation phases, as well as implement further IP obfuscation techniques, such as multi-hop. At ExpressVPN, we are committed to continuously enhancing our solutions and work towards a fully zero-trust environment.

In the meantime, we will maintain our zero-knowledge commitment by subjecting the design to regular reviews by external auditors.



