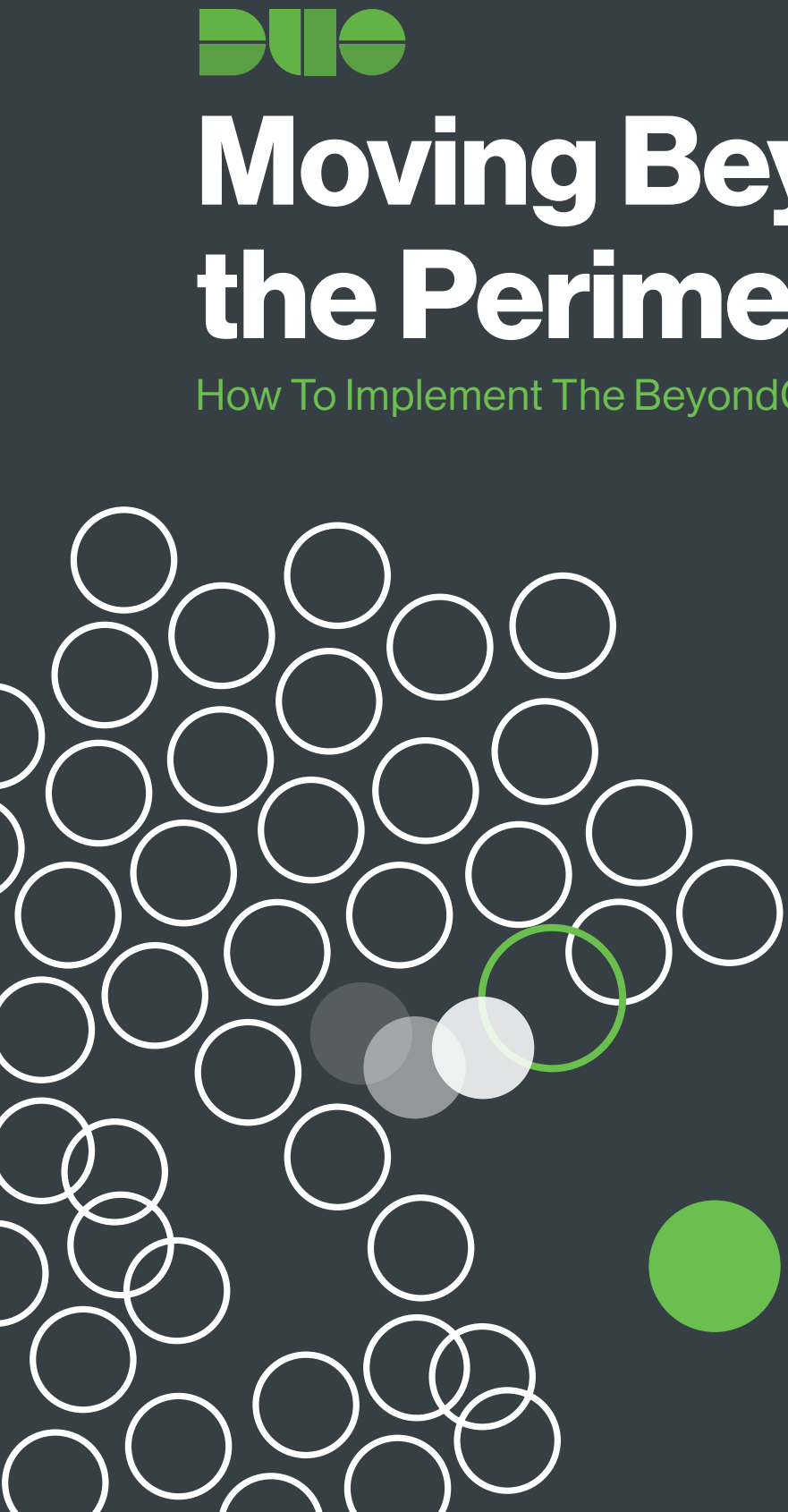
Part Two



Moving Beyond the Perimeter

How To Implement The BeyondCorp Security Model

AUTHOR
WENDY NATHER

DESIGNER
HAFSAH MIJINYAWA

Executive Summary

In the **first white paper of this series**, we described the origins and vision of the BeyondCorp security model. To mitigate the risks from placing too much trust in the internal network, Google developed a new way of thinking about Trusted Access.

By evaluating the combination of user and device against varying tiers of systems and data, an organization can enforce the same security policies regardless of where the user, device and application are located.

To start implementing this new framework, organizations should consider taking the following steps:

- **Enrolling users and their endpoints into inventories**
- **Using digital certificates to identify endpoints as "trusted" or "managed"**
- **Creating access policies based on the authenticated combination of user and endpoint**

Other elements in the framework include single sign-on, device inspection, the trust inference engine, and the reverse proxy that protects applications and enforces the enterprise access policies. Google describes its own migration process in **this detailed white paper** (the fourth in a series).

Enterprises often have many of these components already available and can make use of them; Duo also combines many of them in the new edition of its **Trusted Access platform**, **Duo Beyond**.

Enrolling Users and Their Endpoints

Enrollment usually involves a combination of inventory, inspection and verification. You create a list of entities to be entered into the system that you'll use to authenticate them and grant them access (in this case, a list of authorized users and a list of endpoints they're using).

You can use bulk enrollment — that is, you can use the list to create entries for each one without requiring your users to do anything — or you can use self-enrollment, where the users make contact and supply shared data so that you can recognize them.

INVENTORY

What corporate-owned or managed endpoints do you have, and who are the authorized users? What other endpoints are you going to allow?

INSPECTION

Does it conform with our security requirements? (Note: enrollment isn't the only time you'll inspect the endpoint; it should happen automatically with every access decision.)

VERIFICATION

Is this the known user who is presenting the endpoint for enrollment? Is this the same endpoint that we have in our inventory?

Inventory

Start with what you know you have. Regardless of whether you pre-enroll devices you're aware of or if you let users enroll them individually, the process needs to have controls in place so that you have visibility over which assets you expect to see. Most enterprises have some sort of IT asset and configuration management in place, whether it's Active Directory, LANDesk, Jamf, or other products.

Starting with a basic list of hardware tags (or phone numbers) and assigned users will let you recognize corporate systems as opposed to personal ones. For best coverage, plan to start with bulk enrollment, and then fill in the gaps with self-enrollment, because you'll need to plan for ...

Discovery

An important issue within the inventory process is discovery:

- **Are you sure you know all your users and all their endpoints?**
- **How will you handle new or forgotten users?**
- **How will you deal with changes in endpoints?**

One way to handle this is to put discovery into the enrollment workflow, and place it where the users have to go in order to access something important. **Make sure they will access it early and often.**

Everyone accesses an HR system sooner or later when they need to download their tax forms, but that will only be once a year. It's better to place discovery in front of something they use all the time, such as email, reference wikis or directories.

Don't neglect discovery. Many organizations have had policies against using personal devices on the corporate network, but they found out through discovery that literally hundreds of users were doing it anyway.

Ensuring Trust with User-Device Pairs

What do you do about enrolling shared devices? Remember that it's the combination of user and endpoint that you decide to trust, so you can't just decide to trust all devices independently of the users; an attacker could take control of a given endpoint and leverage any other known username and password to get access. To avoid this, you need to enforce user-device pairs by adding multi-factor authentication.

In order to break in, the attacker would need to have the username, password, access to the second factor (such as a software token on a phone), and the endpoint — making it more difficult to get unauthorized access with every piece you add to the puzzle.

So make sure that you have an entry **only for those combinations of user and device that you expect to see.** Sharing may not happen that often, but when it's needed, the enrollment process should accommodate it.

Verification

As we discussed above, it's the combination that earns the trust, so you need to make sure to authenticate the user during self-enrollment. From that time forward, the user will be re-authenticating (with more than one factor!) to the access proxy, along with that user's assigned endpoints.

How do you uniquely identify an endpoint? It's harder than it sounds, particularly when hardware components and their IDs get replaced. Google described how it used a combination of observed and prescribed data to do this. Organizations will probably end up using whatever data they can most easily obtain and match; whatever you do, aim for consistency. Google decided that it would be the certificate that was the arbiter of endpoint identity: if the certificate didn't match what was enrolled, it didn't matter whether any of the system components matched.

Inspection

It would be great if the user's endpoint were in a known clean state when it was enrolled, but this isn't always possible. At the very least, you can decide on what hygiene and configuration settings you want to see:

- **No known dangerous apps installed**
- **Encryption and lock screen turned on**
- **Updated operating systems and plugins**

If you already have an agent installed on the endpoint, you can get whatever data it provides you. If you don't, or if this is the first time you're seeing the device, you'll need something that can perform this inspection without an agent.

When you're building a device inventory and collecting data on the state of those devices at scale, you'll need to build and manage the data pipelines separately. Google's **BeyondCorp paper** described how its multiple device inventories required collecting and normalizing everything into a meta-inventory to feed its downstream components. Configuration data, event log data, information from security infrastructure such as endpoint monitoring, anti-malware and SIEM can all potentially have a role to play in how you infer the current security state of the device at the point of an access request.

If You Liked It, Then You Should Have Put a Cert on it

What Does "Trusted" Mean?

Trust policies and their requirements will be determined by each organization. It used to be that if a user provided the correct login name and password, it proved that the right person was at the keyboard — and we all know how well that worked out.

We ran into the same problem with devices: because it was on the corporate network, we assumed it was supposed to be there, and it got access to anything it asked for. Both of these "tests" failed for a number of reasons:

- **Stolen passwords**
- **Spoofed network addresses**
- **Compromised endpoints**
- **The ability to spread out laterally to other vulnerable systems**

Now, the path to trust needs more checkpoints, such as authentication factors and conditions placed on the device. One of these conditions can be whether it's a managed, corporate-owned endpoint.

Why "Managed?"

A managed endpoint is presumably owned by the enterprise, or at least known: it may be tracked as part of an inventory, enrolled in a configuration and patch management program, and monitored for security events. For this reason, you may choose to trust it more than you would trust an unmanaged, personal device.

Many organizations have the policy that only the endpoints they own and assign to staff can be used to access business data. However, this policy can be difficult to enforce, especially if there's no way to check. There are different ways to try:

VIRTUAL PRIVATE NETWORK (VPN) SOFTWARE

If the endpoint has the VPN client installed, it's assumed to be an approved and managed asset, so whoever is using it will be allowed to access the internal network from the outside (say, at home, or from a hotel or coffee shop). SSL VPN software doesn't require an installed client, so it provides more convenience for the user, but it also removes that implicit enforcement.

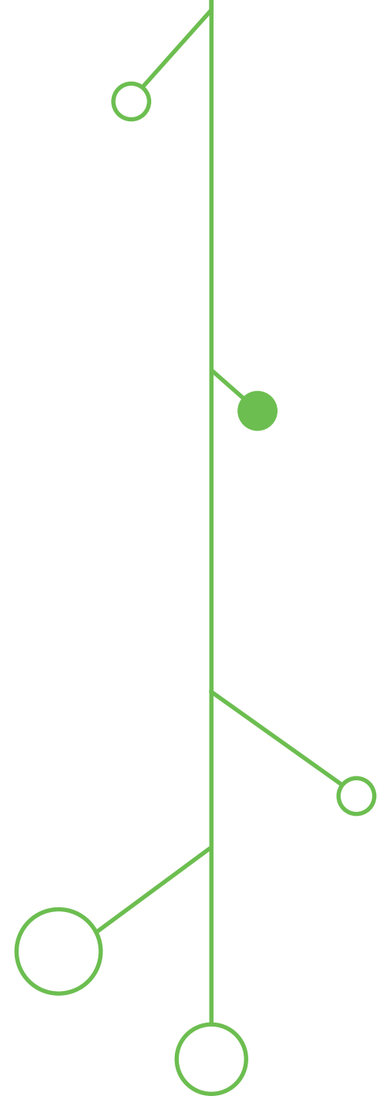
NETWORK ACCESS CONTROL (NAC) SOFTWARE

With common port-based NAC, if the endpoint has an 802.1x certificate installed, it's assumed to be an approved and managed asset, so whoever is using it will be allowed to connect to the internal network from inside the building.

MOBILE DEVICE MANAGEMENT (MDM) SOFTWARE

Enrolling mobile devices into this system allows you to enforce configuration policies by installing an agent.

In each of these cases, you've marked the endpoint as trusted by installing something on it (or given it a second factor, "something it has"). What else could this marking mean for a "trusted endpoint?" It could be used for endpoints that don't belong to the organization, but that have been vetted (for example, a consultant's laptop that has been scanned). The important point is that **you've seen the device before and expect to grant it access**, as opposed to endpoints that are trying to access your applications that you've never seen before that may be used by attackers. Either way, it can be used to control which devices can access your business data.



Unfortunately, if you can't manage an endpoint, it's generally more difficult to convince that endpoint owner to let you install something on it. A certificate or other method of fingerprinting is lightweight, and may be more acceptable than installing running software. Still, the key requirement is to make that marking unforgeable and prevent it from being copied to another device.

Since you will be making trust decisions based on the marking's presence or absence, it functions as yet another authentication factor, and it needs protection in the same way that you must protect the primary user credentials (username and password) and the second factor (such as a **one-time password**, **U2F device**, or **push-based authentication)**.

All the Single Endpoints

In Google's BeyondCorp framework, certificates offer a way to identify the device as managed. You can take it a step further by including device and user data in the certificate, tying them together so that **neither one's credentials can be leveraged alone.** You can set policies so that users must use known and approved endpoints to access the most critical data and applications (for example, privileged users must use a corporate-owned device).

Likewise, even if a user loses credentials to an attacker, the attacker still needs to use a valid endpoint belonging to that user to get into an application — it's not enough to have the username and password with a different corporate device. **Trusting the devices only if they're with the right user** is a new step towards tighter security that the BeyondCorp model makes possible.

Creating Policies for Big Success

Your access proxy takes on the role of enforcing access to corporate resources, regardless of whether they're outside or inside your traditional perimeter. Enforcement strategy is one way we express risk tolerance; rightsizing those policies depends on factors such as sensitivity, threat, user community, regulatory requirements, and any number of other things. And enforcing policies consistently for both sides of the firewall is a key tenet of the BeyondCorp model.

Tiers of Trust

A major drawback to the classic network perimeter security model was that organizations tended to have one level of trust everywhere on the inside. Building in different tiers required network segmentation that was often too complex to implement. With BeyondCorp encouraging a new look at separating out levels of trust at the application layer, it's important to determine where your most critical and sensitive data, applications and control functions are so that you can protect them with higher trust requirements.

Some examples of the most critical accesses might be:

- **Control systems, which are used to grant or change access (such as administrative consoles, configuration management systems, identity stores, certificate authorities, and authentication servers)**
- **Systems which manage availability (load balancers, backups, HVAC systems)**
- **Financial and human resources applications (including payment systems)**
- **Research and engineering systems holding intellectual property**
- **Applications and storage for customer, patient, student or citizen data**

In order to access these, users and their devices may need a higher level of trust, which means they need to pass more tests and comply with stricter requirements. Start with a baseline level of trust for all users and all devices regardless of what they're accessing, and then add more to reach the level of risk management you need for access to the most sensitive tiers.

Wielding Access Policies

Your access policies are much more flexible than a stop-or-go approach. Like a multi-use tool, you can use them to bludgeon, nudge, slice or tap. Here are some of the types of access policies to consider.

Warning - strongly recommending or requiring action at some point in the future.

Blocking - the heaviest of the policies, preventing access entirely.

Logging - taking note of a condition or event.

Mitigating - loosening or reversing the effects of another policy based on certain risk scenarios.

Responding - taking short-term actions to react to a particular situation.

Warning

You can use **warning policies** to drive behavior.

> **A warning is a reminder with a little weight behind it: if you don't do what the reminder says, sooner or later, you will suffer a consequence.**

For example, most organizations put a grace period in their policies to give users time to update their software before they're either forcibly upgraded, or they're blocked until they catch up. So, if a new version of a particular browser comes out, your users have one month to upgrade to it, or be blocked after that grace period has expired.

If your warning policy has no consequence attached to it — that is, the user may override or ignore the warning every time — then it's little more than an irritating flag that pops up in the middle of that user's workflow. And if the warning is about something that the user can't take action on, it's even more frustrating.

If a system can't be updated because of some other dependency, then the warning serves no purpose and merely trains the user to ignore the irritant. When it comes to access policies, make sure that you ask for a concrete action that's within the recipient's capability, and be prepared to take an enforcement action within a reasonable time period based on your risk estimates.

Blocking

A policy for blocking is best suited to situations where you don't have wiggle room. For example, many organizations want to block access to critical applications from non-managed personal devices. Either the device is corporate-owned and "blessed," or it isn't.

Many organizations are interested in **blocking based on geolocation**. If you are quite sure that you never need to allow access from certain regions, a general block will work, but that's not always an option if you do business with them or you have users who travel there.

Bear in mind that blocking based on IP address or a derived geolocation won't necessarily protect you from a determined attacker who can spoof those things, but in general, it can work as a filtering mechanism for large segments of the population who should not even be trying to authenticate to your applications.

Mitigating

There are some policies that are used to mitigate the effects of other policies. Multi-factor authentication is an important security control, but some users don't like having to use it every time they need to use a resource. An organization may decide that after the initial authentication to a system, the risk is low enough to delay re-authenticating for a certain period of time.

One example of this is "remembering" a user or device, or both. Most services that offer MFA allow each user to "**remember this device** for 30 days," for example. Setting that time period involves making a risk calculation on your side as to how likely it will be that a user's device could be lost or stolen; it's a tradeoff against convenience. The same principle applies to application session length — how often you need the user to re-authenticate if, say, they don't lock their device when it's not being used.

Another possible mitigating policy is to skip the second authentication factor for devices on particular trusted network segments. However, once you begin trusting something more when it's on the "inside" of your network perimeter, you're in danger of undermining what BeyondCorp is all about: the idea that you shouldn't trust the inside any more than the outside. So use these "loosening" policies with caution.

Responding

Organizations can also put temporary policies in place to respond to a particular event. If a critical vulnerability is announced for a plugin, for example, and you know your users are at risk because the vulnerability is already being exploited, then you may want to **block users** until they get the patched version installed. In other words, you would shrink the time window or grace period of a regular policy for just this one situation.

Other response-type policies could include placing geolocation or network restrictions on a device that someone can't find — until they either find it again, or determine that it was really lost or stolen. If they find it where they expected, they can use it again right away, but if someone else tries to use it from a different location, they won't be able to access corporate data with it.

The same idea applies to an employee who is leaving; while they work out the notice period, their access policies might be tightened so that they can't access applications that contain large stores of sensitive data.

Managing Exceptions to Policies

For every policy, there is an equal and opposite exception. There may be good reasons why a set of endpoints can't be fully updated:

- **They don't have regular access to enough network bandwidth**
- **They're dependent on one application that requires a certified stack to operate**
- **It's too politically sensitive to block your CEO even if she rooted her own phone. You never allow traffic through an anonymized proxy, except that one time when an employee is traveling abroad and can't access some home resources any other way.**

Strictly speaking, a firewall is an exception in itself: you know it's risky to connect to the Internet, but you do it anyway because there are strong business reasons to do so. The firewall embodies and manages those exceptions ("Okay, but only for web applications ..."). For your users, have a workflow process ready to receive exception requests, and for yourself, be ready to record and approve them with reminders to follow up if the policy exceptions are only temporary.

Another purpose for adding policy exceptions is to introduce change over time. You may have stricter policies in place for a smaller user group to try them out before deploying them to the rest of the population. Exceptions can also help to troubleshoot all sorts of problems if you suspect they're being caused by an access policy: for the one user, you create an exception for each policy that you know is being applied to them, until the guilty one surfaces (or all of them are ruled out).

From the Big to the Small

Access policies can be used not only at the network and application levels, but also at the device and behavior levels. You can start by blocking access to whole categories of outliers (such as banning any use of an insecure browser), and then work your way towards requiring better endpoint hygiene, such as screen locks. In some cases, you can require your users to validate their 2FA confirmation with a fingerprint, so that even if an attacker has access to the unlocked device, they still can't finish logging into the application.

The most important thing is to carve away at the devices, software, sources and behaviors you know you don't want to allow, thereby reducing your exposure to attacks.

> **Changing the security lifestyle of an organization takes dedicated work, but once you have the controls fit more closely to where they belong — the users, their devices and the applications — then you'll be addressing the gaps in today's traditional security paradigm and moving Beyond it.**

The Maturity Process With BeyondCorp

Rome wasn't built in a day, nor was BeyondCorp. Google describes in detail what they learned from the deployment process (ACLs are complicated), and building the architecture from scratch offers many such learning opportunities. As we mentioned before, organizations don't have to consider it a cutover-style project, but rather an evolution as new controls are added to fill the gaps in the old ones.

Some proposed stages of maturity are listed below.

Early Maturity: Building the Inventories

This is where you should start if you don't already have centralized identity management. User, application and local system accounts need to be tracked in one place, even if they're not managed from there. You don't have to collect them all at once — strictly speaking, you could just collect the first set of users from the first application you plan to protect in the BeyondCorp fashion — but if you plan to implement BeyondCorp throughout your enterprise, you'll eventually end up with all users in the same repository.

The same goes for devices: knowing what you have, sorting out the ones you manage, and tracking changes to them. The output from this inventory will help you decide what device policies you want to enforce (can you require encryption for all of them?).

Who can reach this stage: organizations of any size, although for larger ones it will take longer, usually due to decentralized user and asset management.

Mid-Stage Maturity: Core Deployment

As you start increasing the level of control you have over access to your most important resources, you'll grow the groups of user-device pairs that you manage. This generally happens on an application-by-application basis, since the authentication for each one will need to move to the access proxy. An organization with a core deployment using the BeyondCorp framework might have its system administration and infrastructure tools (such as Active Directory) migrated, along with financial systems, human resources, and applications using intellectual property or regulated data.

Who can reach this stage: Any organization that is able to create policies, issue endpoint certificates, set up the access proxy, implement MFA, use an identity provider for primary authentication, and change the domain name service (DNS) entries for the relevant applications. This requires a certain level of technical expertise as well as control over the environment, so smaller organizations who outsource all their support may run into obstacles here.

Peak Maturity: All the Users, All the Devices, and All the Apps

Can you ever make the network irrelevant? That's the end state of BeyondCorp, and although that theoretically means that enterprises can ditch their traditional firewalls, practically speaking, it's not likely to happen.

Any enterprise that is still hosting any connected infrastructure will be responsible for protecting it against many other sorts of network-based attacks (such as denial-of-service), not just authentication-level ones.

BeyondCorp is not a silver bullet that will take care of all risks; it's a way of increasing the security level of what used to be viewed as a "safe" environment. Until you remove the complication of legacy systems, software and protocols, or move all the hosting off-premises, you'll need your traditional perimeter to continue standing its watch.

By contrast, cloud-first organizations can use the BeyondCorp model to increase the control that they have today over access to third-party SaaS applications. **By allowing only known, validated user-device combinations to authenticate, they can filter out a whole realm of daily attempts from attackers trying to take over accounts.** Anyone can try to log into a public SaaS application today with a stolen set of credentials - but it'll be harder if they have to use that user's fully-patched endpoint and their 2FA on yet another device.

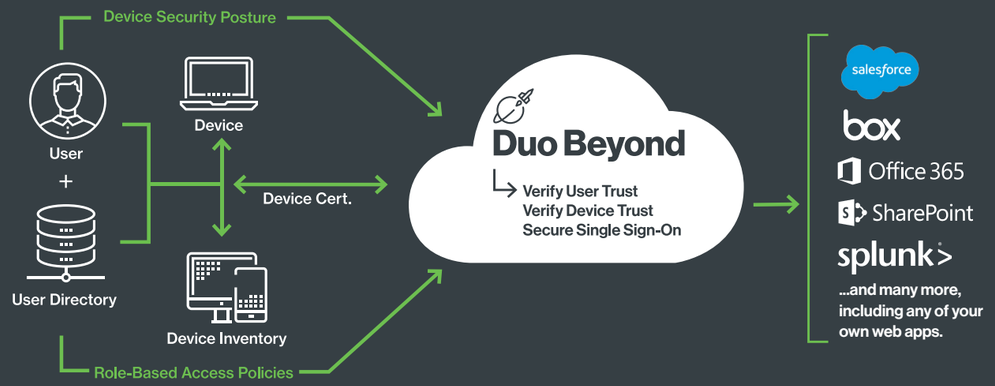
> **Ultimately, BeyondCorp is a new way of thinking about security and trust. Applying the "zero-trust" attitude to every enterprise design and process is the real peak maturity on this curve.**

Read Part 1 of our Moving Beyond the Perimeter series to learn about theory behind the BeyondCorp security model: **duo.sc/beyond-part1**

How Duo Beyond Can Help

Building a whole infrastructure to accommodate a new way of thinking takes a long time. At Duo, we've shortened the path by building a platform, called **Duo Beyond**, that contains most of the components already included:

- **The device inventory**
- **Identification of trusted devices**
- **Access control engine**
- **Access proxy**
- **Secure single sign-on**
- **Multi-factor authentication (MFA)**



Just bring your own identity provider, your list of users and corporate-owned endpoints, and, of course, your strategy for building tiers of trust.

Duo Beyond

With Duo Beyond, organizations can easily implement a BeyondCorp security model within their own organization based on the identity of users and security of their devices. Give your users a consistent user experience while securely accessing cloud or on-premises applications.

Regain trust of your endpoints with Duo Beyond:

- **Easy-to-use two-factor authentication**
- **A secure single sign-on experience**
- **Complete device visibility**
- **Identify corporate vs. personal devices**
- **Easy device certificate deployment**
- **Block untrusted endpoints**
- **Secure access to internal apps, without a VPN**
- **Phishing simulations to assess risk**

Try it today at **duo.sc/beyond**

The Trusted Access Company

Follow **@duosec** on Twitter and Instagram.