How Cybercriminals Abuse Cloud Tunneling Services

Archived: 2026-04-05 13:37:44 UTC

By Ryan Flores, Stephen Hilt, Lord Remorin

Cloud tunneling services, which allow users to expose internal systems from their homes or businesses to the internet by relaying the traffic through cloud-based systems, have grown in use over the past few years. Unfortunately, as with any kind of service that helps developers and infrastructure administrators, cybercriminals have been abusing these services for various illicit operations.

Legitimate cloud tunneling services are beneficial to a wide range of people, from home users to large-enterprise employees. They are also commonly used to help developers test and deploy code, and to share services with select people and groups on the internet. The use cases for these services range from small-scale, such as playing local games with friends, to industrial-scale, including testing out large systems on the internet before pushing the code to production. Malicious actors, on the other hand, have their own method of using these services: They employ cloud tunneling to mask their real locations as well as for short-lived purposes, so they do not deploy permanent online infrastructure.

In this article, we describe the legitimate uses of cloud tunneling services for enterprises and contrast them with how cybercriminals abuse these services. We also delve into security implementations intended to help users completely block cloud tunneling, or as an alternative, since some might be using these services, to best gauge and monitor the potential risk that using cloud tunneling services could bring. We also take a look at defense strategies that involve the detection of both authorized and unauthorized use of cloud tunneling services, including any potential attempt to bypass corporate restrictions by cybercriminals or rogue employees.

An Overview of Cloud Tunneling Services

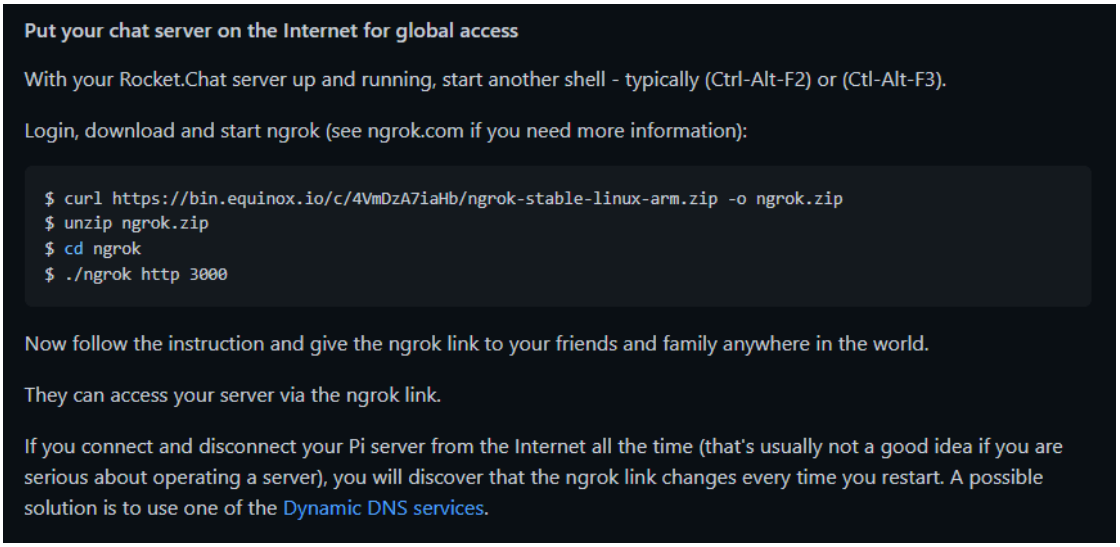
A tunneling service is used to expose a service through a cloud system so as to obscure the original source, whether for short or long periods. This is done typically because the service is behind a security system or the user wants to limit who has access to the original source. A tunneling service is a useful tool for users who want to expose only a specific portion of the service online. The use of tunneling is not limited to large systems in a corporate environment; it can also be used in smaller, more personal projects. During our research into tunneling systems, we observed it being implemented in a variety of online

systems, including login pages, security tools, chat platforms, video recording systems — such as IP cameras, network video recorders (NVRs), and video management systems (VMSs) — and even game servers.

Before we discuss how cloud tunneling services are being used for malicious purposes, we must first understand how these services are used legitimately within the network. The primary benefit of cloud tunneling services is as a user-friendly tool for exposing services to the internet. With these services, users can quickly deploy local development services online while avoiding the hassle of configuring firewalls and registering domain names.

A developer who needs to test their web applications online can simply run their service and configure a tunnel to circumvent any firewall configuration — even when the development server is running behind network address translation (NAT). We have seen a popular cloud tunneling service, ngrok, being used to expose the staging environment of a BroadVoiceopen on a new tab service used for VoIP applications. In this scenario, a tunnel enables the development team to safely perform third-party integration testing and provides the team the ability to sort out any issues before deploying the app or service into production.

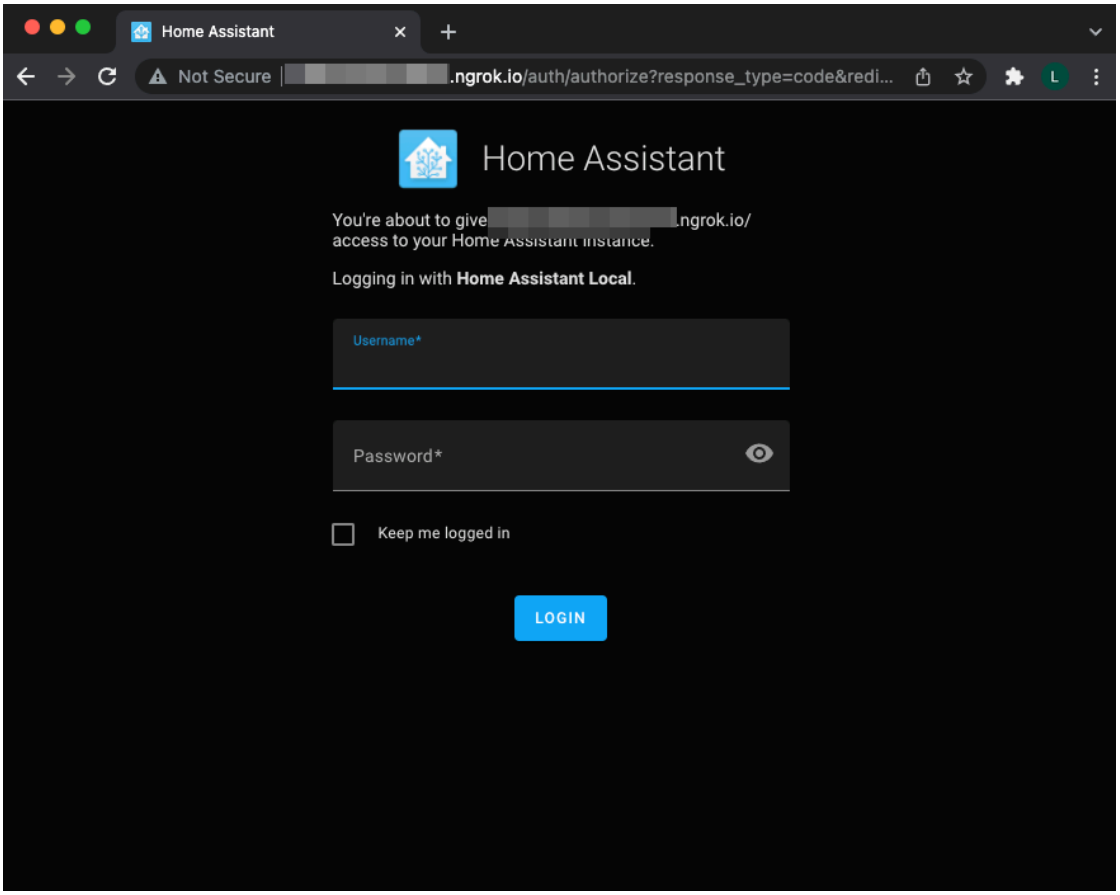
Using a tunnel can be very convenient for a home user who either is anxious about modifying the port forwarding rules on their router or has restricted access to their router's configuration because of internet service provider limitations. The user can install the service they want to run and simply use a tunnel application for clients to connect through the internet. Some services even suggest using tunnels in their tutorials. For example, Rocket.Chat's installation guideopen on a new tab on Github suggests using ngrok as an option for sharing the chat server online.



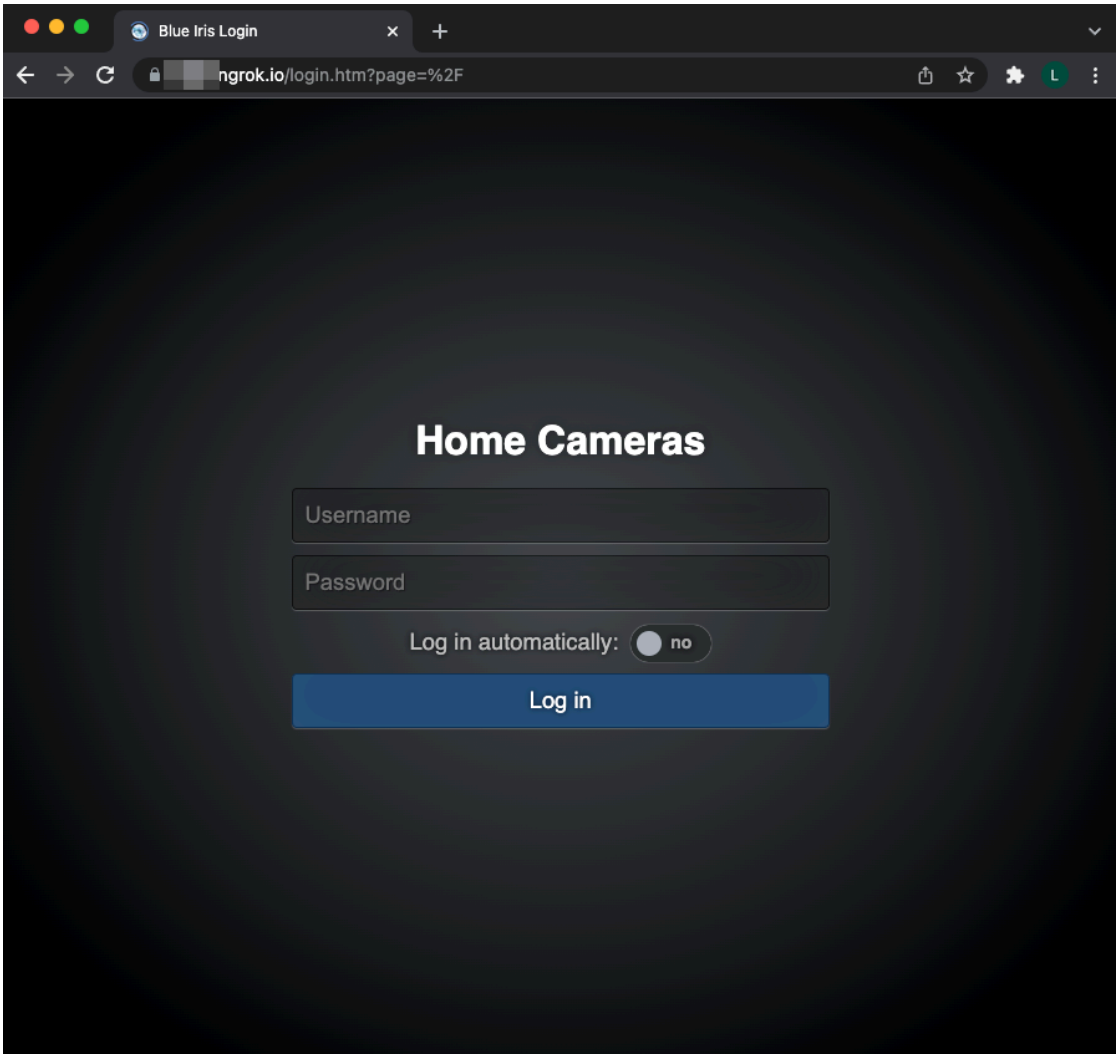
open on a new tab

Figure 1. Rocket.Chat's installation guide suggesting the use of ngrok

In addition to chat platforms, we have also seen ngrok being used for home automation applications like Home Assistant, Homebridge, and OpenHAB. For DIY users who want to install video monitoring or surveillance in their homes, Blue Iris is a popular choice among VMS applications and provides tutorials on its forum on how to use ngrok to easily access its web portal and perform monitoring away from the home network.



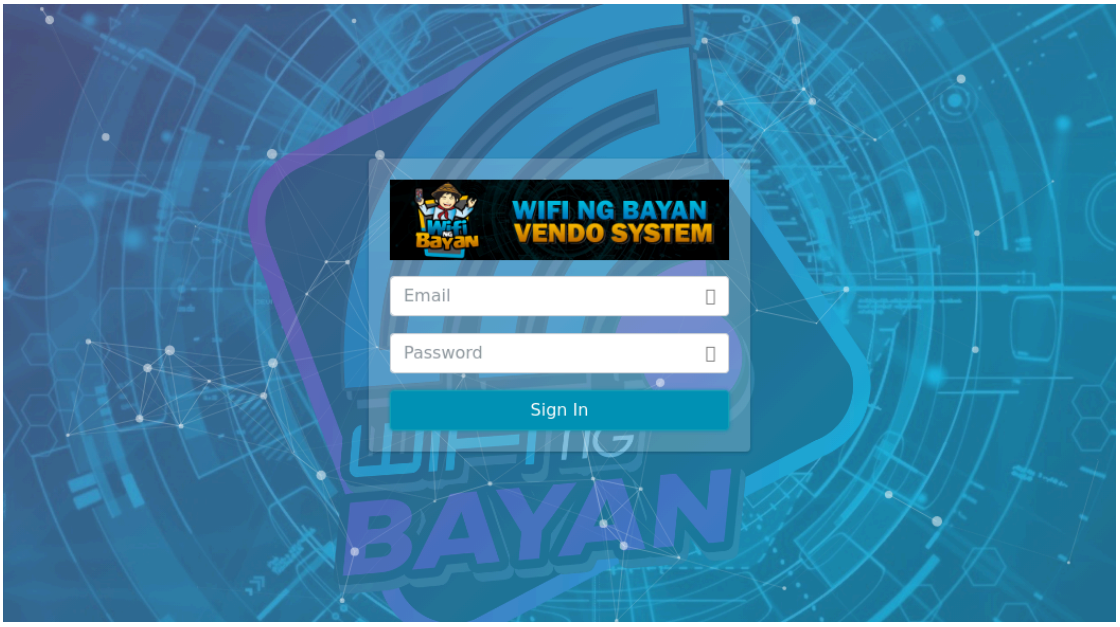
open on a new tab



open on a new tab

Figure 2. Screenshots of the login pages of Home Assistant (top) and Blue Iris (bottom)

Aside from developers and home users, we have also seen ngrok being used by e-commerce businesses in some parts of Asia. A good example of this is the web panel for administrative access for coin-operated Wi-Fi network access in the Philippines. In this example, the machines act as a gateway for selling internet access via Wi-Fi that is paid by the minute. The management of these machines can be done locally, but since it involves managing multiple machines from various locations, administrators can simplify this by using tunneling services for access.

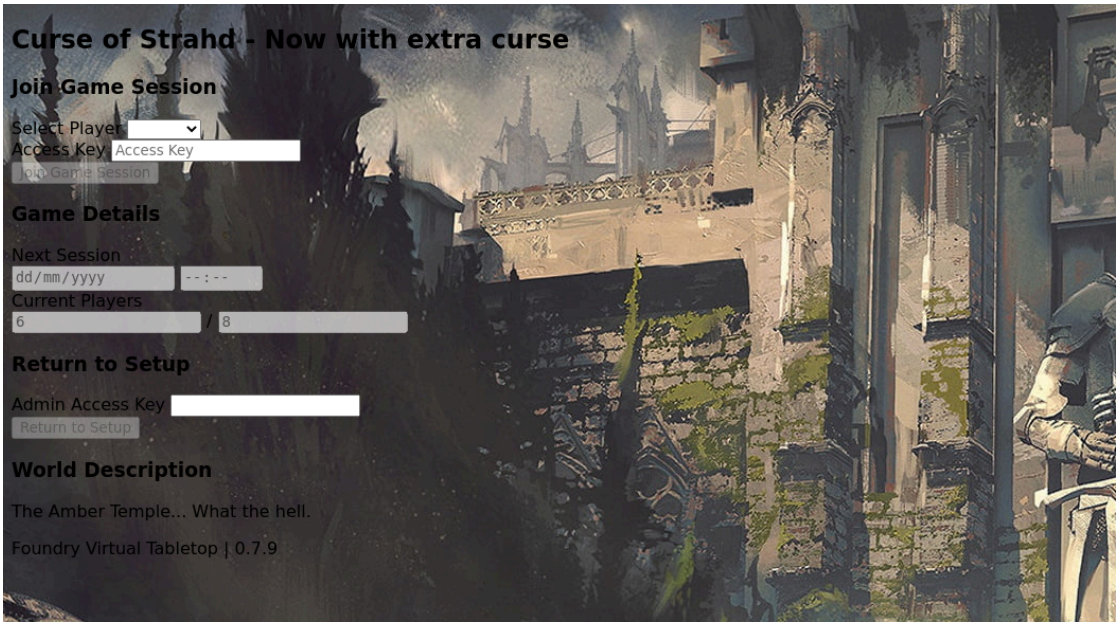


open on a new tab

Figure 3. A screenshot of the web panel for coin-operated Wi-Fi access

Yet another use of ngrok we have commonly observed is for hosting local game servers on the internet. Games that have locally hosted servers, like Minecraft and Foundry Virtual Tabletop, frequently popped up while we were investigating ngrok traffic. Employees can deploy local game servers from their home networks and host them on ngrok to be accessed on a corporate network. For organizations that restrict access to gaming servers, it is an added challenge to identify gaming-related network traffic.

open on a new tab

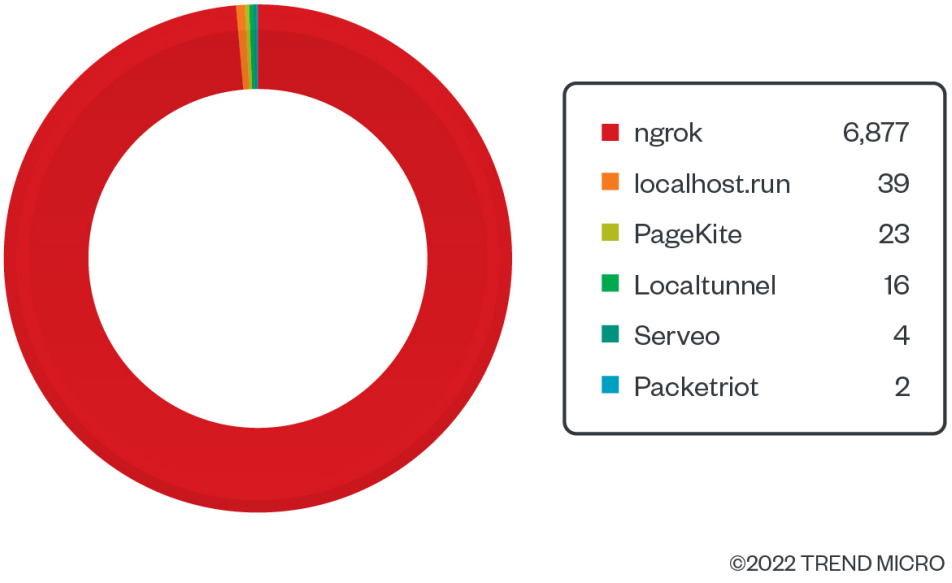


open on a new tab

Figure 4. Servers for Minecraft (top) and Foundry Virtual Tabletop (bottom) hosted on ngrok

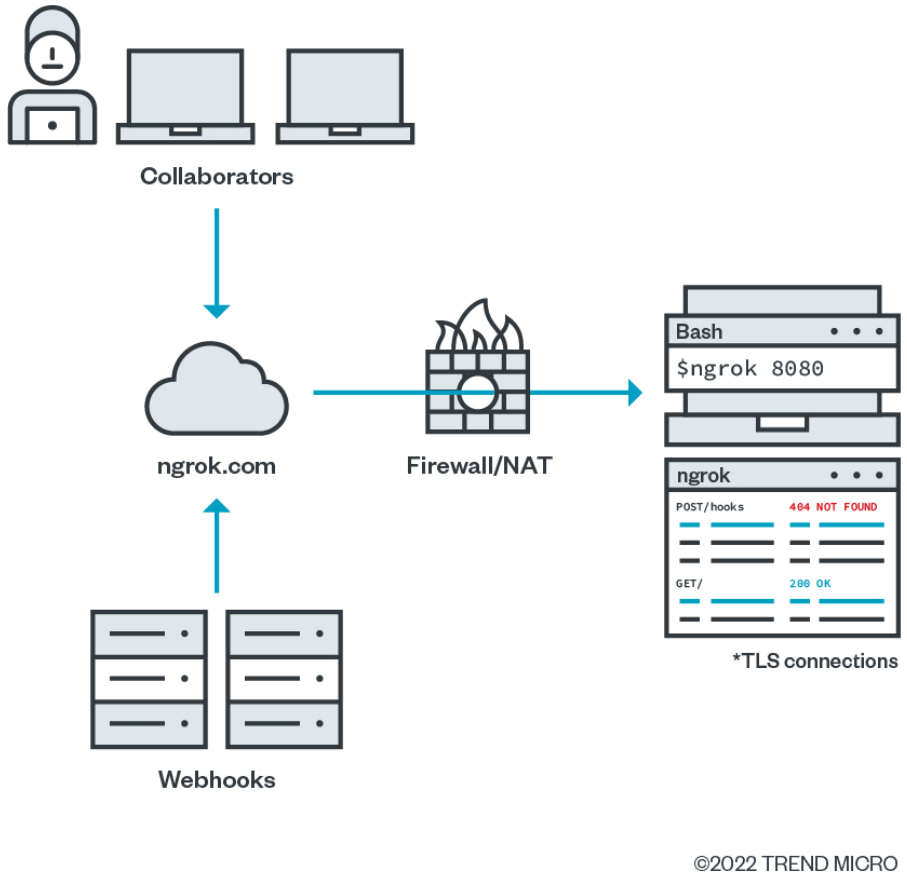
Testing Methodology

While looking at a few cloud tunneling services, we determined that the primary application we would focus on would be ngrokopen on a new tab because of its overwhelmingly large market share that emerged in our datasets, roughly 99% of the HTTP traffic of cloud tunneling services. The other cloud tunneling services we considered were localhost.runopen on a new tab, PageKiteopen on a new tab, Localtunnel, Serveoopen on a new tab, and Packetriotopen on a new tab. While these services could be used by malicious actors in the same manner, this paper focuses on ngrok as our use case.



open on a new tab

Figure 5. The distribution of HTTP traffic for cloud tunneling services based on our datasets in 2021 (in terms of how many times data was sent over HTTP)



open on a new tab

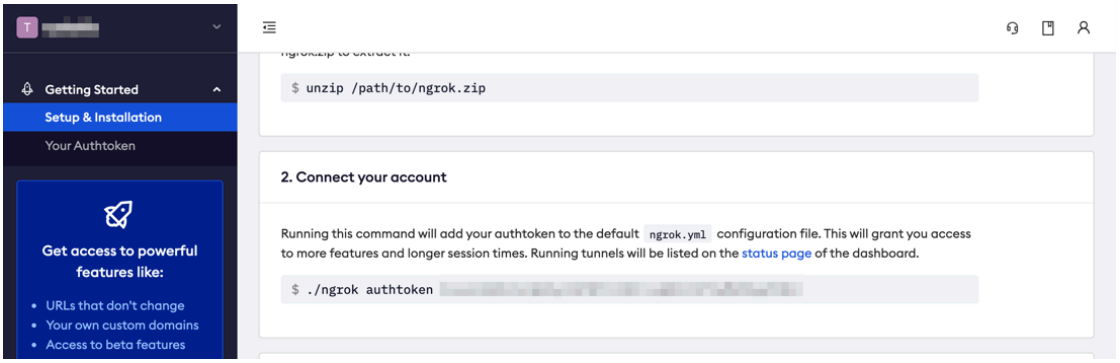
Figure 6. An overview of how ngrok worksopen on a new tab

Not all of the services use the same back-end implementation, but the concepts of how they operate are the same. For ngrok, the user downloads a binary, for Windows, macOS, or Linux.

The ngrok platform offers three account levels:

- Free use that does not require an account
- A free account with limited features
- A paid account with extra features that can cost up to US$25 a month

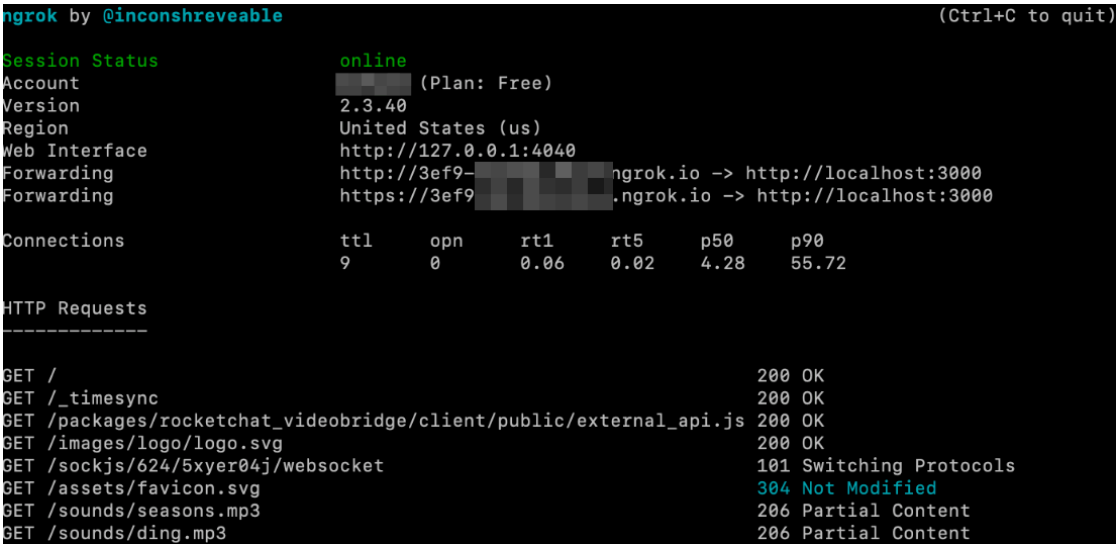
Initially, we chose a free account and went right to setting it up. After downloading the binary, we needed an authtoken to have access to more features.



open on a new tab

Figure 7. A screenshot of the ngrok website

To test the application, we set up a Rocket.Chat server and pointed ngrok to port 3000, which is the port we used for our Rocket.Chat install.



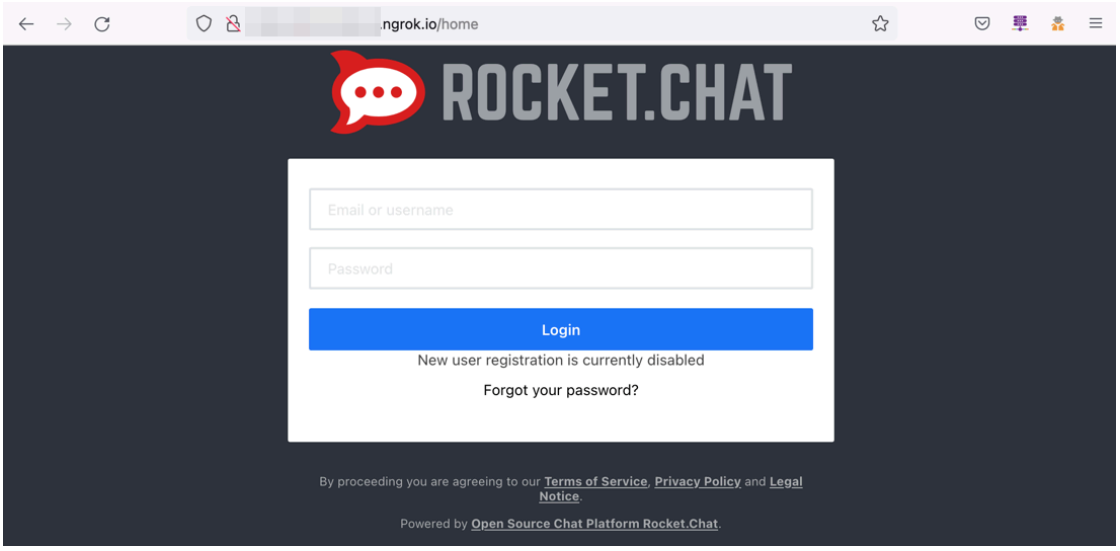
open on a new tab

Figure 8. Building the tunnel up with the ngrok command

The way ngrok works is based on the type of account, and whether the TCP or HTTP options are being used. When we started this research using the ngrok HTTP URLs, we included the computed subdomain to ngrok.io in the URL to point the browser to the HTTP-type connections.

Based on our observations, the subdomain contained 12 characters, either with or without hyphens based on when it was generated. A newer generated subdomain contains hyphens (specifically, separating the IP address of the machine that is running ngrok) and starts with a 4-byte hex word. On a higher-tier paid account, the user can use a custom subdomain that can contain, for example, their organization's name or another important aspect of the company or service they are trying to expose. If the user is not careful, they could expose their internet point of presence to an attacker. At the same time, an attacker could also have their real IP address exposed.

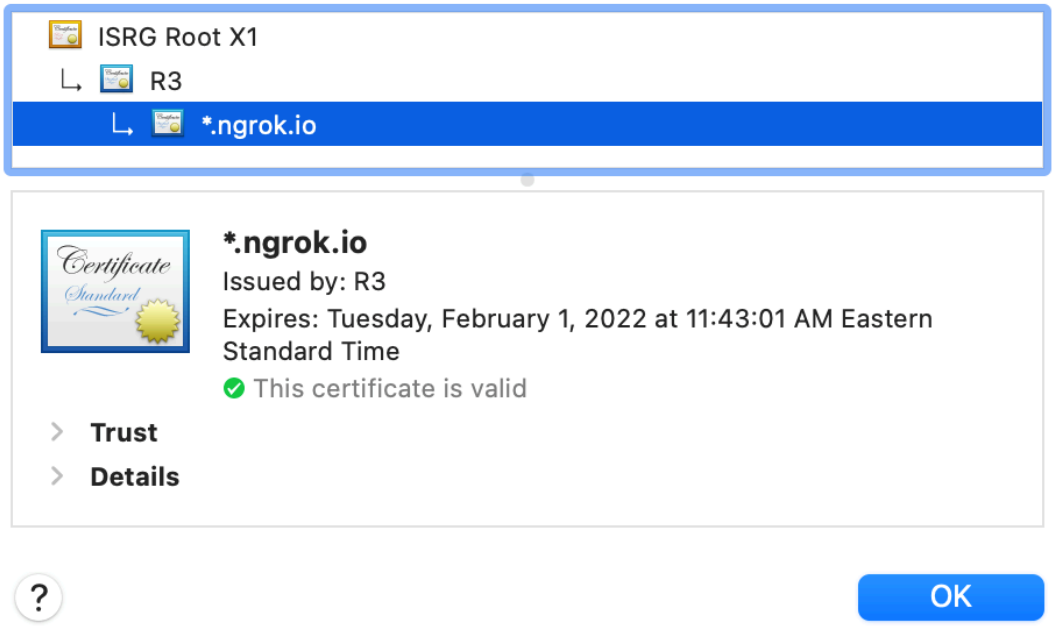
An attacker could perform some basic open-source intelligence (OSINT) to find the target domain. Examples of this include searching GitHub for code that might point to cloud tunneling services and using the search strings "site:victim.com" and "ngrok.io" on Google. Furthermore, if the TCP option is in use, instead of the 12-character subdomain, the host number appears as the subdomain along with a randomly assigned port. An example of this would be 4[.]tcp.ngrok.io:10667.



open on a new tab

Figure 9. A Rocket.Chat example used for testing purposes

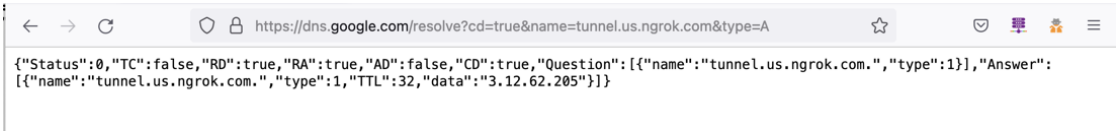
As expected, the service was exposed and a user could log in and use the system as if locally connected. Based on some testing, we discovered that the system running the ngrok tunneling service would communicate only with ngrok's servers via an encrypted SSL tunnel over port 443, while the client machine would interact with the URL that was provided. Interestingly, ngrok provides an HTTPS option using a valid certificate that is wild-carded for *.ngrok.io.



open on a new tab

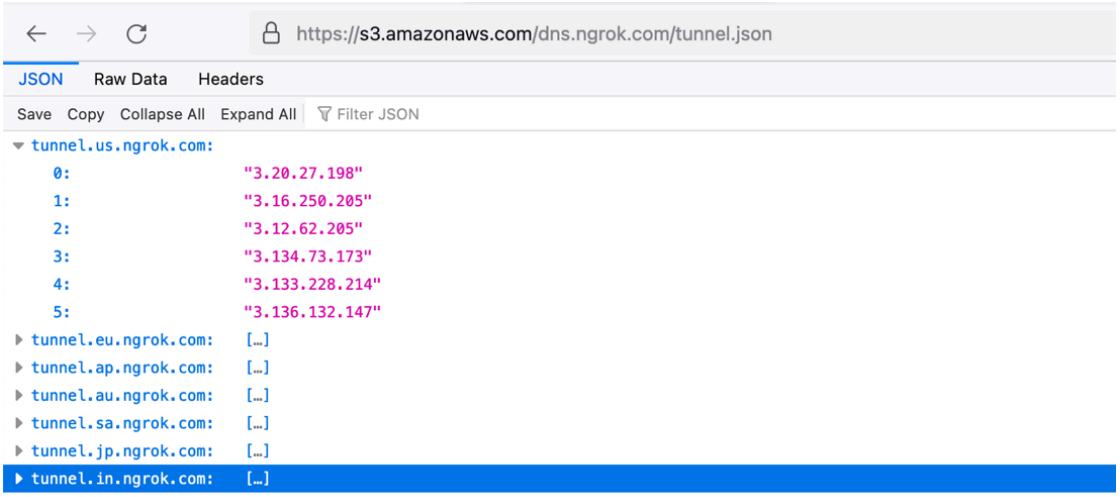
Figure 10. A screenshot of the certificate information used by ngrok.io for HTTPS

One characteristic we discovered while looking into how the ngrok service works was that it is persistent in trying to build the tunnel from the machine running ngrok. It first tries to connect to tunnel.REGION.ngrok.com, and if that fails, it tries to connect to equinox.io to look for an update. It then tries to connect to dns.google.com to look up ngrok.com's IP addresses and, finally, if that fails, it pulls a JSON file from s3.amazonaws.com, hosted on Amazon Simple Storage Service (S3).



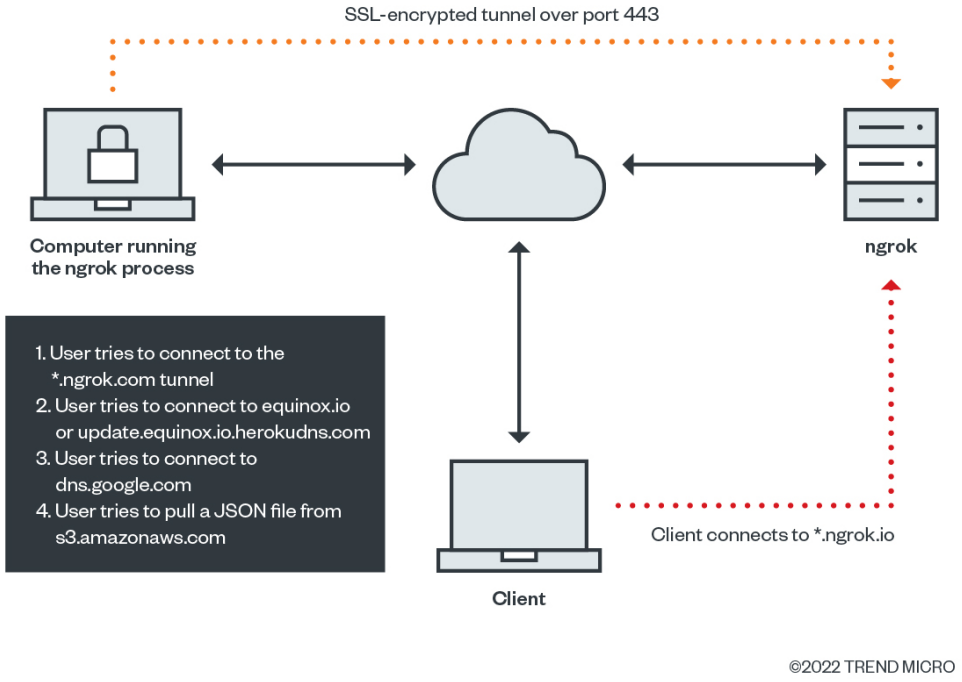
open on a new tab

Figure 11. A screenshot of dns.google.com looking up tunnel.us.ngrok.com



open on a new tab

Figure 12. A JSON file hosted on an Amazon S3 bucket with the IP addresses of tunnel.REGION.ngrok.com



open on a new tab

Figure 13. A diagram of how ngrok tunnels are built and how clients interact with them

Possible Malicious Uses of ngrok and Other Cloud Tunneling Services

As with any other online platform, malicious actors could take advantage of ngrok and other cloud tunneling services, and use them for malicious purposes. Services like ngrok and Tor could be used by cybercriminals to hide their true IP addresses. However, the difference between ngrok and Tor is that services tunneled through ngrok are still accessible via the regular internet while services tunneled through Tor can be accessed only through Tor. This can be a deciding factor for a cybercriminal when choosing a target network, as they might prefer one with wider access over a network that can be accessed only via Tor.

During our research, we categorized cloud tunneling service abuse into two malicious use cases: internal threats and external threats. Internal threats are attacks where cloud tunneling services are unknowingly used on an infected endpoint or network to expose internal services like SMB, FTP, and HTTP, while external threats are the more typical cyberattacks such as phishing, drive-by download, and malware command-and-control (C&C) communication through the cloud tunnel network.

Aspect	**Internal threats**	**External threats**
Attacker already present in target machine or network	Yes	No
Cloud tunneling service binary execution	Server is running in a machine on corporate network	Server is running remotely
Possible tactics, techniques, and procedures (TTPs)	• Lateral movement • Data exfiltration	• Malware file hosting • Malware C&C server • Phishing scams • Exploit kits

Table 1. A comparison of internal and external cloud tunneling service threats

Internal Threats

A cloud tunneling service could be used maliciously on a target machine or network to unknowingly expose nonpublic services to the internet. These services could be anything that is accessible from the machine the cloud tunneling service is executed on, such as internal web applications, database servers, and file-sharing services.

Cybercriminals and insider threats such as rogue employees could easily expose company-restricted services that are meant to be accessible only on the intranet without having to deal with router configurations. Attackers with access to compromised servers could also host services like web servers and malware C&C servers for nefarious activities.

In July 2020, we discovered an attack that exposed an SMB port using ngrokopen on a new tab. After exposing the service, the attacker used tools like SmbExec to run a local Windows shell on the infected machine, and then downloaded and executed a keylogger. This incident shows that a legitimate application used for tunneling services could be misused. And since the network traffic while establishing a tunnel to ngrok's servers is normally seen on DevOps networks, a malicious actor could mask its network activity and avoid its being flagged as malicious.

The data exfiltration capabilities of ngrok include a built-in feature to host a local directory straight to the internetopen on a new tab without starting a file-hosting service on the affected machine.

External Threats

While we have seen cloud tunneling services being used to bypass the firewall rules set up by network administrators to expose internal services, a far more common use of the service by cybercriminals is for malware traffic or for hosting

phishing websites. For malicious actors using malware that requires communicating back to a C&C server, using cloud tunneling services could be advantageous because of the ease with which users can set up infrastructure. In addition, malicious actors using paid plans could also hide their true IP addresses via custom reserved subdomains. Thus, they could avoid registering domains and setting up SSL certificates, which, if not done properly, could lead to identity exposure through an OSINT investigation.

Tutorials on sites like YouTube and GitHub also contribute to the growing popularity of cloud tunneling services for exposing services to the internet. There are a good number of tutorials on YouTube that show cloud tunneling services being used for malicious purposes, such as setting up a remote administration tool (RAT) C&C server or an HTTP server for hosting a phishing page. We believe that these tutorials cater to the more entry-level cybercriminals who are looking to deploy RATs for their own use.

As of this research, the most common malicious use of cloud tunneling services is for setting up phishing pages and malware C&C servers. We discuss these further in the following subsections. It is important to note, however, that these are just the most common and there are other malicious use cases, such as using ngrok for hosting malware binaries or exploit kits.

One example of ngrok's being used in an attack occurred in 2019, when the threat actors behind the Lord Exploit Kitopen on a new tab (not connected with one of the authors of this research) used the service to host its web server. Initially, they used it to deliver the njRAT malware, before eventually expanding its use to the distribution of the Eris ransomwarenews-cybercrime-and-digital-threats. During this campaign, the random generation of subdomains by ngrok became an advantage to the threat actors running the exploit kit.

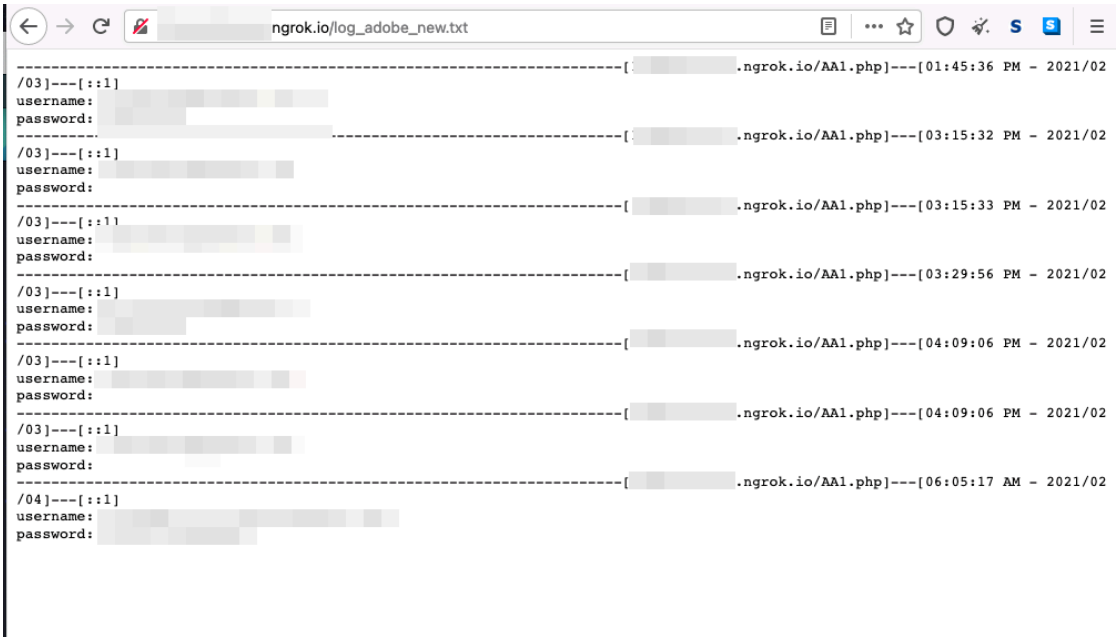
Phishing

Penetration-testing tools such as SocialFishopen on a new tab have integrated cloud tunneling services into their phishing toolkits (specifically, version 2 for SocialFish, which, unlike the latest versions, has ngrok integrated). With such toolkits, penetration testers and red teamers, as well as cybercriminals, could not only create phishing pages with just a few keystrokes, but also generate working cloud tunneling service URLs that could be immediately sent out via email, chat apps, and a variety of other methods. This makes it very convenient for would-be phishersopen on a new tab, as they would not even need to register any domain name, deal with any web hosting service, or go through the trouble of hacking web servers and hosting the phishing page there.

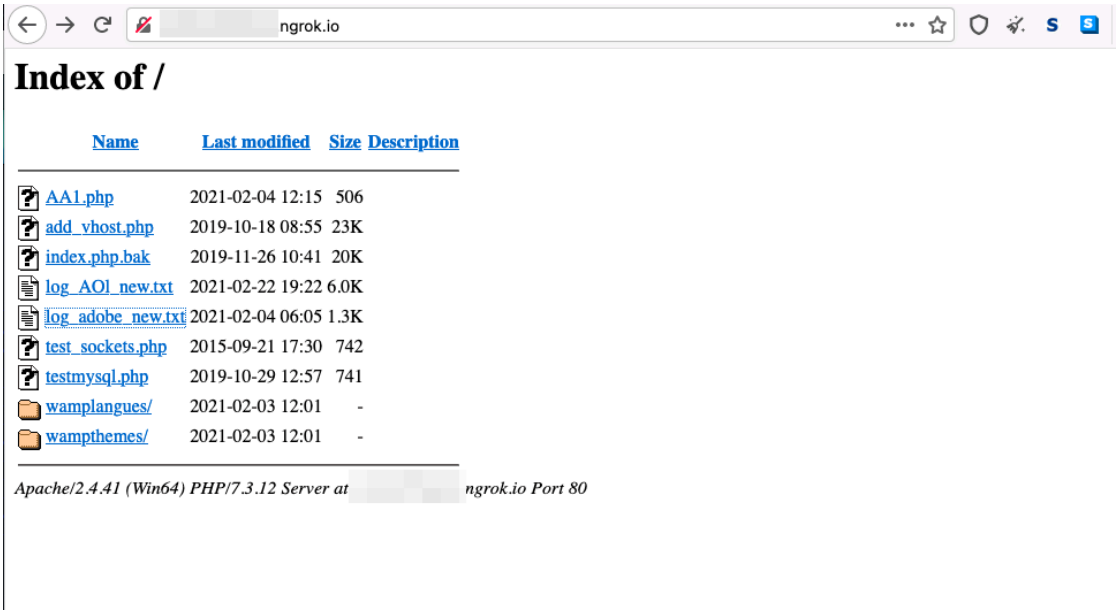
open on a new tab

Figure 14. The Social Fish website offering an older version with ngrok integrated

The most popular phishing kits we found were targeting users of Instagram, Facebook, Bank of America, Gmail, Paypal, Netflix, and Dropbox. We also found phishing kits for ANZ, Bank of Colombia, Chase Bank, and Banco de la Nación Argentina. However, the latter phishing kits were smaller in volume than the former group.

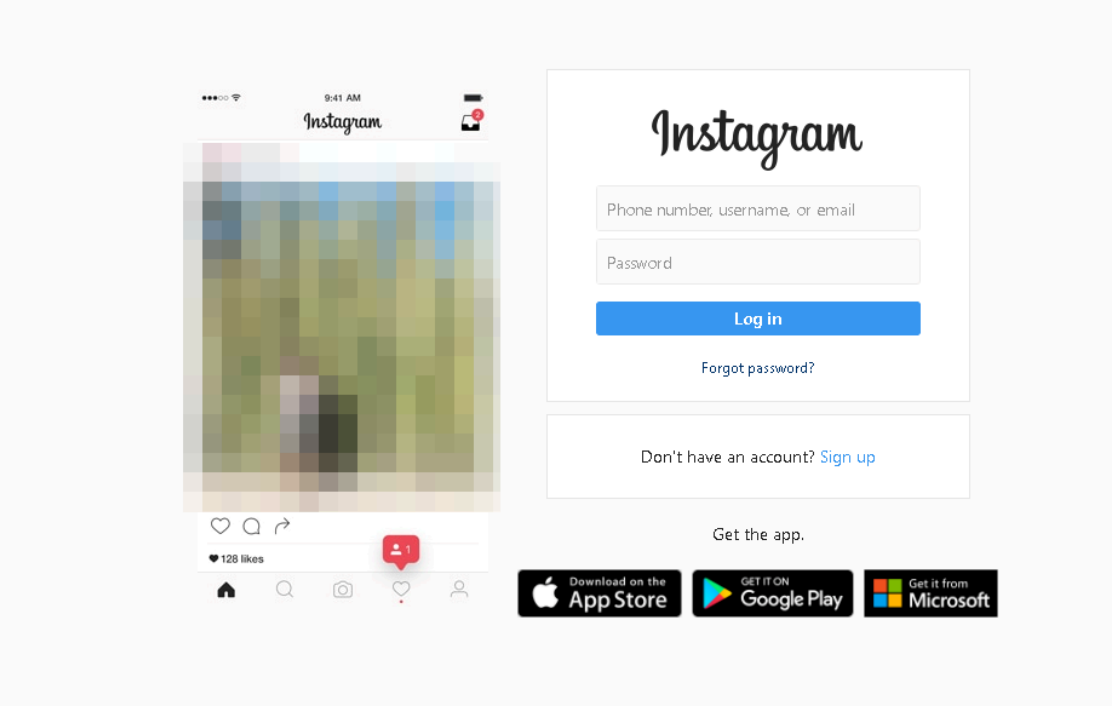


open on a new tab



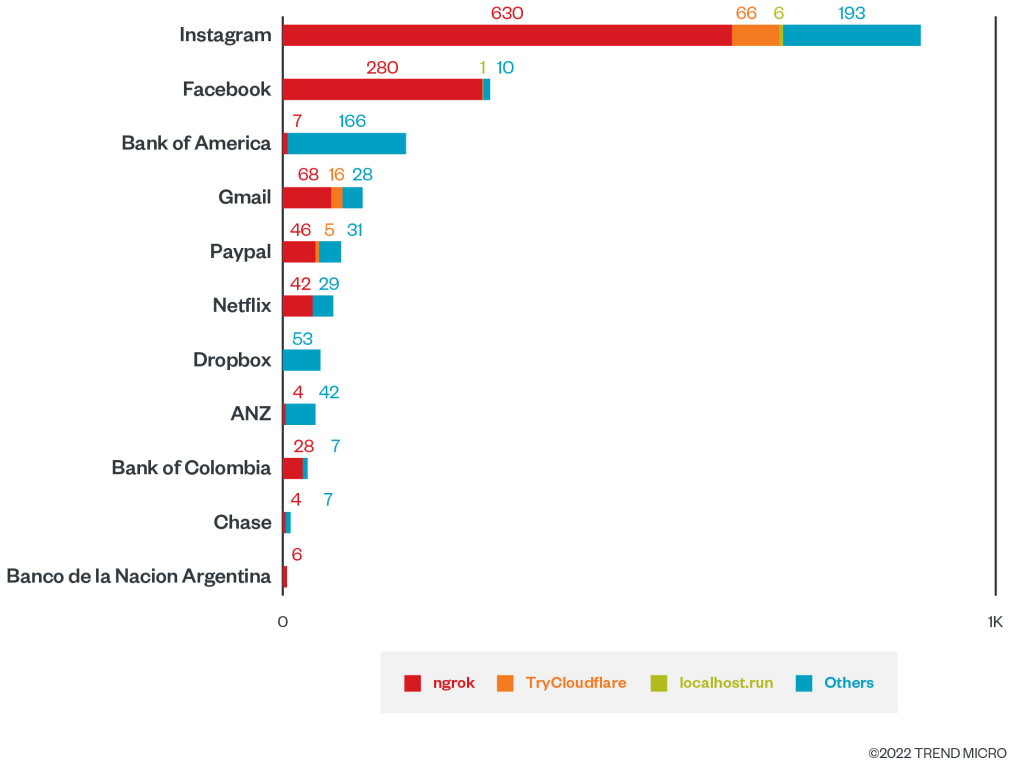
open on a new tab

Figure 15. The open directories of phishing pages targeting users of Adobe (top) and AOL (bottom)



open on a new tab

Figure 16. A sample Instagram phishing page found in ngrok URLs



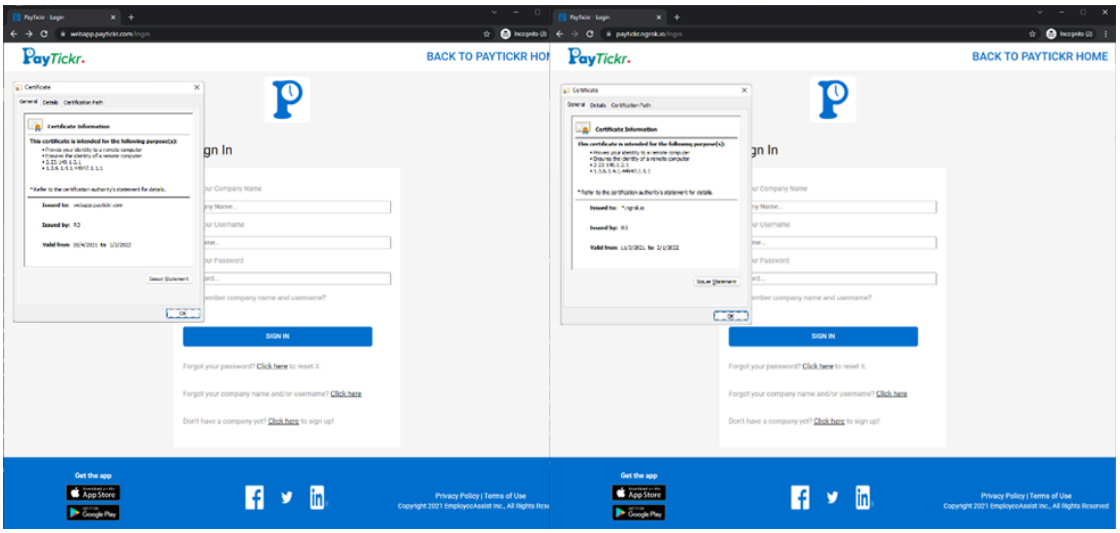
open on a new tab

Figure 17. The distribution of phishing kits according to volume (in terms of how many times data was sent over HTTP)

Normally, phishing kits abuse web hosting services or are hosted on compromised sites and typo-squatting sites. Tunneling services have been gaining popularity among malicious actors since they provide a convenient alternative for hosting phishing pages.

One reason phishing pages that use cloud tunneling services could be dangerously deceptive is that if a user is sent, say, an ngrok-based phishing URL, there is a high chance that the recipient would assume that it is a legitimate site, since they can visit the HTTPS version and get a valid signature from ngrok.

Over the years, people have been trained to view the lock icon in their browsers as a guarantee that the websites they are visiting are secure. In this case, there is a valid certificate for the domain, which in turn might lead to higher victim counts for a phishing campaign. Furthermore, if they are using a paid ngrok subscription, for example, attackers could use a subdomain similar to the one they are mimicking to add authenticity to their phishing campaign.

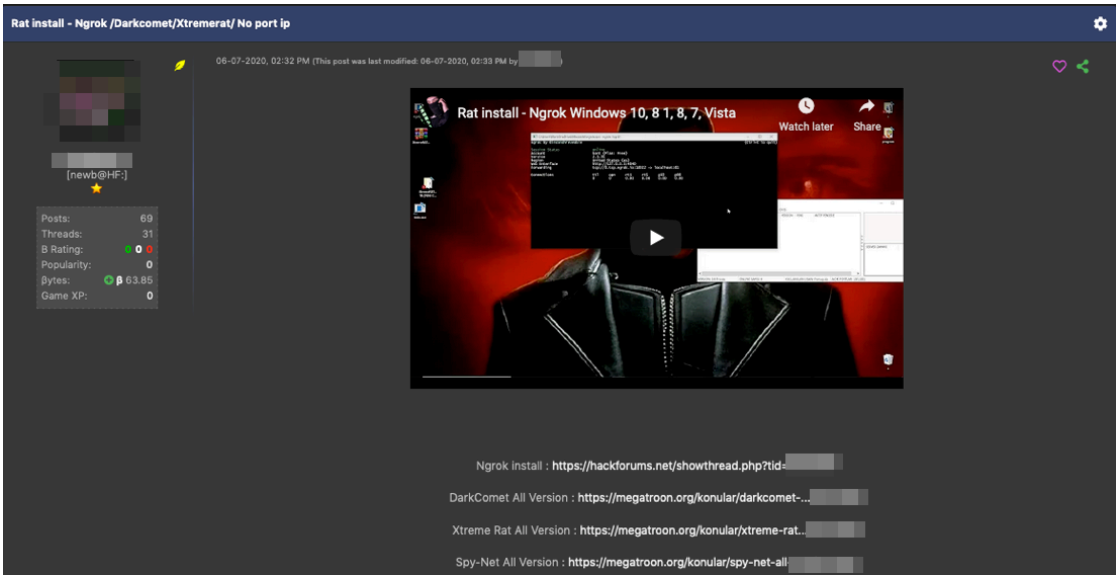


open on a new tab

Figure 18. A comparison of a legitimate page (left) and a phishing page (right) for PayTickr logins

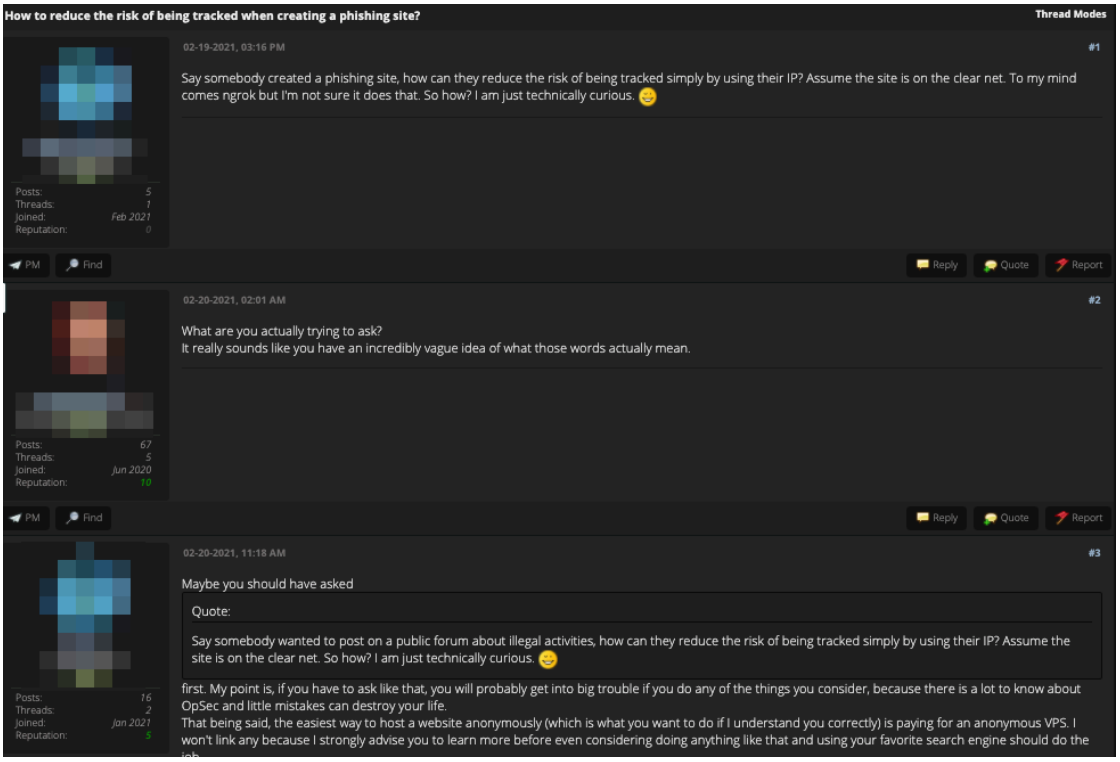
Malware Command-and-Control Servers

In the cybercriminal underground, there is plenty of discussion surrounding cloud tunneling services, particularly ngrok, and how they could be abused for malicious purposes. When we started this research, we browsed underground forums and found plenty of examples, from developing or hosting malware to tutorials on making phishing sites using cloud tunneling services.



open on a new tab

Figure 19. A screenshot from a thread on Hack Forums on how to install the DarkComet RAT using ngrok

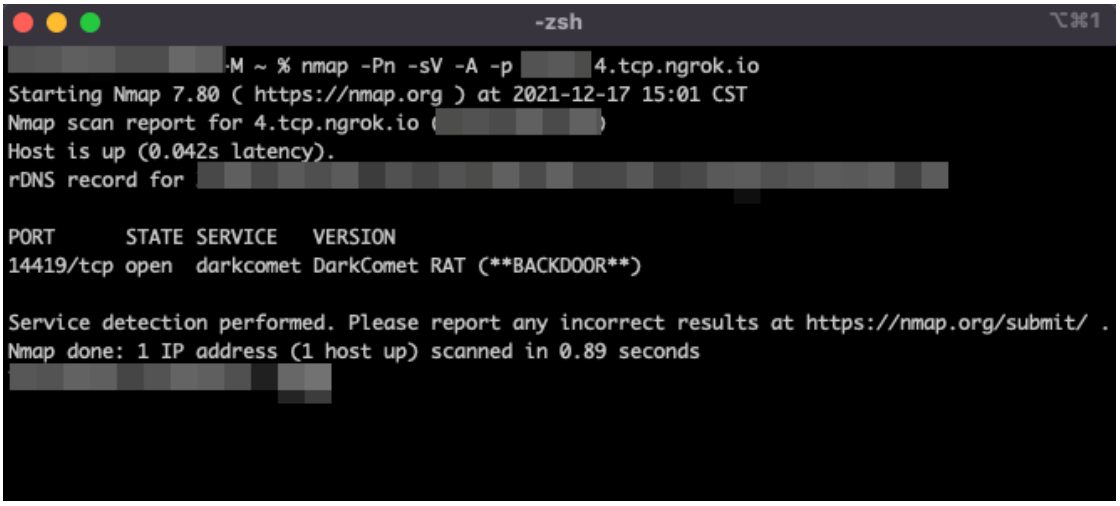


open on a new tab

Figure 20. A screenshot from a discussion on ngrok and phishing

The discussions in the underground match what we have observed as the main form of abuse of cloud tunneling services, such as for phishing and for tracking avoidance. Cybercriminals are paying attention to cloud tunneling services as they attempt to hide themselves from their victims by tunneling communications through the services' systems.

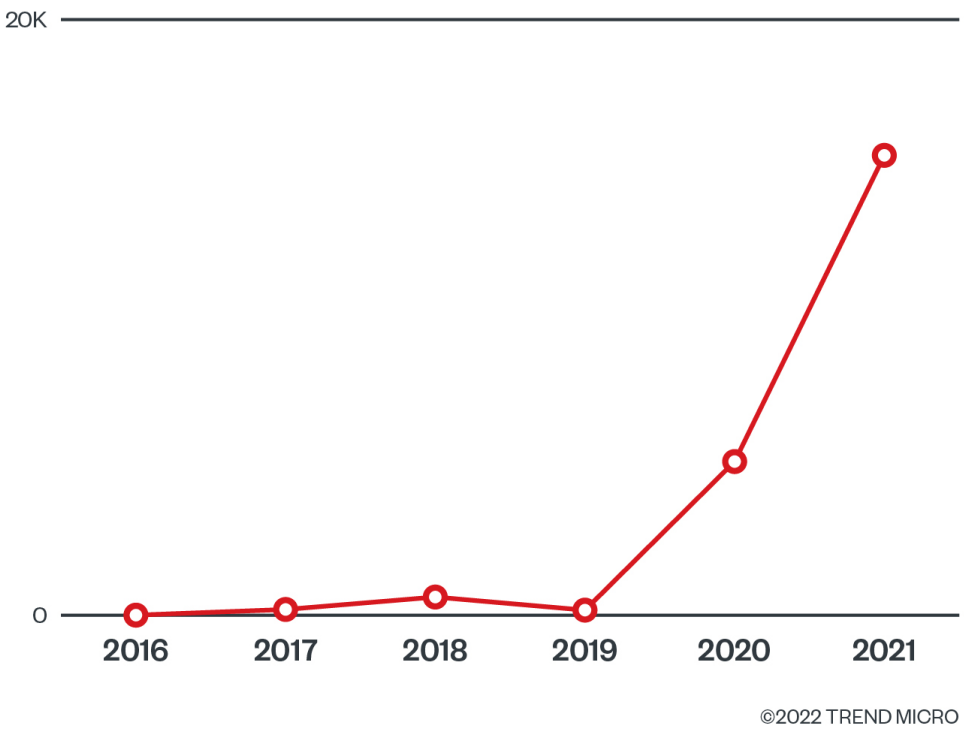
Malicious actors could hide their true identities by not exposing any IP addresses or domain name registrations. By using cloud tunneling services, they could make the network traffic of their malware look as though it were being used for legitimate purposes, making it seem that the target's network is communicating with clean IP addresses and domains. For this reason, we have seen an increase in commodity malware families (which are commonly used by entry-level cybercriminals) that use cloud tunneling services to expose their C&C servers. One popular example is DarkCometopen on a new tab, which has had C&C servers hiding behind ngrok's network.



open on a new tab

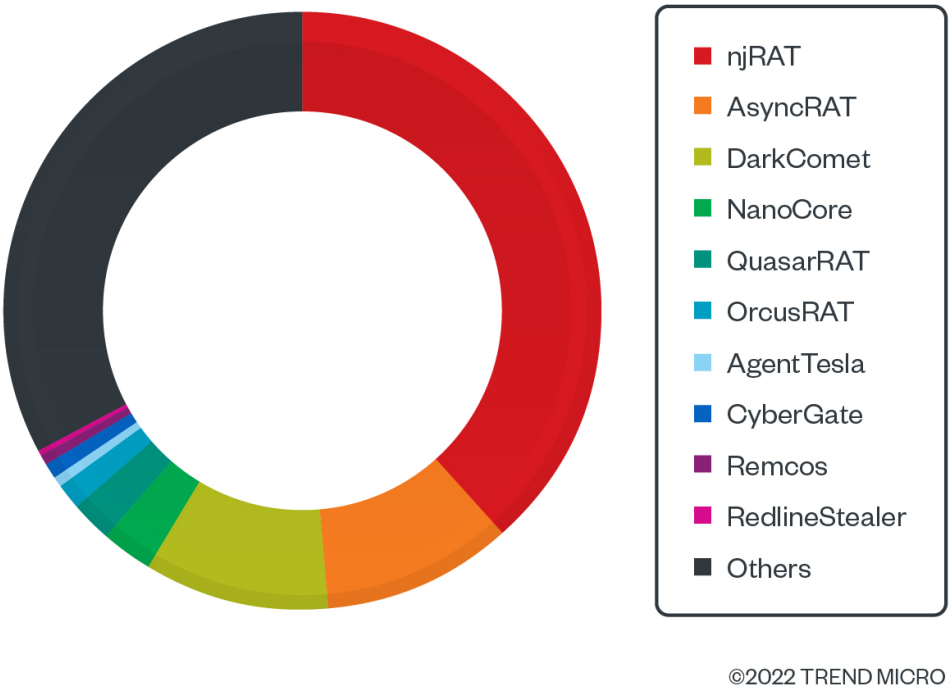
Figure 21. DarkComet being hosted on one of ngrok's TCP ports

The use of ngrok with malware has been rising since 2020. From Jan. 1 to Dec. 31, 2021, we found nearly 16,000 unique malware samples that used ngrok to route C&C traffic. Malware families such as DarkComet, AsyncRAT, NanoCore, and several keylogger families were among the most recurring samples we found that used ngrok, with njRAT as the top malware family. The reason these specific malware families have been commonly used is that they are easily downloadable and the instructions on how to configure them are widely available.



open on a new tab

Figure 22. A comparison of the numbers of malware files using ngrok from 2016 to 2021



open on a new tab

Figure 23. The distribution of malware families using ngrok as part of their routines in 2021

```
// w.A
// Token: 0x06000011 RID: 17 RVA: 0x000021B8 File Offset: 0x000003B8
public A()
{
    this.VN = "SGFjS2Vk";
    this.VR = "0.8d";
    this.MTX = "";
    this.MT = null;
    this.EXE = " Explorer.exe";
    this.DR = "TEMP";
    this.RG = "772d3e1cf411932582ba4607caf9d2f7";
    this.H = "0.tcp.ngrok.io";
    this.P = "10000";
    this.SPR = Conversions.ToBoolean("False");
    this.Y = "|'|'|";
    this.BD = Conversions.ToBoolean("False");
    this.usb = new USB();
    this.SPL = "[endof]";
    this.kq = null;
    this.LO = new FileInfo(Application.ExecutablePath);
    this.Cn = false;
    this.C = null;
    this.sf = "Software\\Microsoft\\Windows\\CurrentVersion\\Run";
    this.F = new Computer();
}
```

open on a new tab

Figure 24. An njRAT configuration containing ngrok TCP tunnels as C&C host

Penetration testers also take advantage of services like ngrok to prevent being detected on the network, whether through hosting malware or using it as a C&C server. It is not just commodity malware families (like the ones mentioned in the previous paragraph) that use such services; penetration-testing tools commonly used by both penetration testers and cybercriminals, such as Cobalt Strike and Meterpreter binaries, do as well.

Defense Measures

As with any tool or service that could be exploited for malicious purposes, ngrok and other cloud tunneling services have their own set of advantages and disadvantages. In their case, the advantages stem from the afforded convenience of setting up a private server and hosting it on the internet. But this convenience is a double-edged sword since it could also be beneficial to attackers who want to integrate these services into their schemes.

In this section, we list several defense measures network administrators can implement to prevent the abuse of ngrok and other cloud tunneling services on their networks.

Managing the Access of Certain Users to Cloud Tunneling Services

For some businesses that have cloud tunneling services as an essential part of their operations, access should be limited only to users who need these services. Doing this can prevent attackers who gain access to the network from using the services for C&C, data exfiltration, or other malicious purposes. Employees with access to these tunneling services should be regularly checked and logged for access to these services to ensure that their access is being used for approved purposes as defined by the organization.

With unlimited access, network administrators could often be blindsided as to what services are exposed since almost anyone from the company can simply start a tunnel to access systems that are meant to be accessed only from internal networks. For large organizations, it might be better from a security standpoint to prevent employees from using cloud tunneling to expose services to the internet, and to use a virtual private network (VPN) instead to connect to the intranet.

Creating Application Filters

Stopping the installation of specific binaries for cloud tunnels and adding alerts when they are present on a machine can help minimize the risk of unintended use of these services on the network.

Preventing the Creation of Tunnels Using Cloud Tunneling Services' IP Addresses

Network administrators can block SSL handshake to prevent establishing a secured tunnel between a host machine and a cloud tunneling service's server. In the case of ngrok, this can also be accomplished by blocking all connections going to ngrok's IP addresses listed in the JSON file hosted on Amazon S3open on a new tab. A Bash command can be executed to list all of the IP addresses associated with ngrok that are being used to establish a tunnel, and another to create firewall rules for dropping outgoing connections to ngrok tunnels.

```
curl -s https://s3.amazonaws.com/dns.ngrok.com/tunnel.json | jq -r '.[][]'
```

open on a new tab

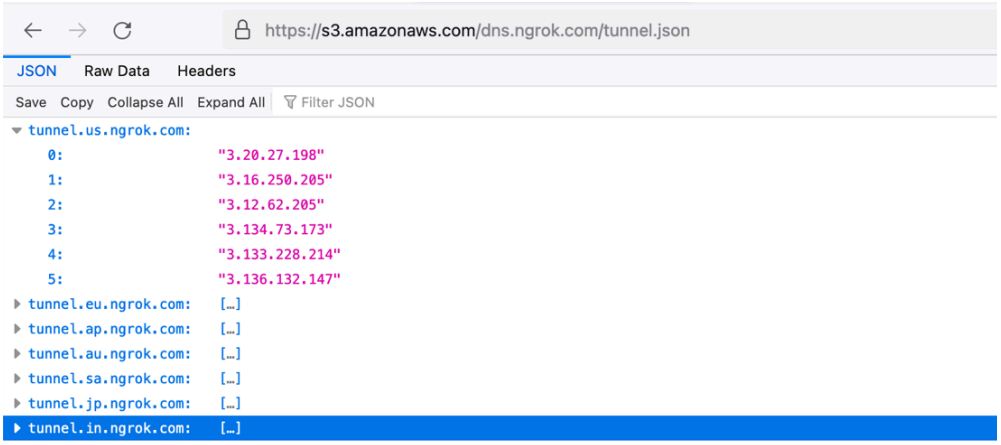
Figure 25. The Bash command to list all IP addresses associated with ngrok that are being used to establish a tunnel

```
for ip in `curl -s https://s3.amazonaws.com/dns.ngrok.com/tunnel.json | jq -r '.[][]'`;
     do echo "iptables -A OUTPUT -s 0.0.0.0/0 -d $ip -p all -j DROP"; done
```

open on a new tab

Figure 26. The Bash command to create firewall rules for dropping outgoing connections to ngrok tunnels

Since ngrok implements multiple methods of resolving the IP address of a tunnel, blocking the IP addresses can prove more effective at preventing the successful creation of a secured tunnel. The list of IP addresses changes over time and thus should be checked regularly to be able to block them while minimizing the interruption of other services that might be hosted on these IP addresses.



open on a new tab

Figure 27. IP addresses used by ngrok

Creating Alerts for or Blocking External Threats Using Cloud Tunneling Services

As discussed earlier, phishing kits and C&C servers could be tunneled through services like ngrok. The risk of a successful phishing attempt or malware communication can be reduced by preventing network traffic (HTTP and TCP connections) coming from cloud tunneling services. In ngrok's case, this can be achieved by creating alerts for or blocking the following:

- DNS requests to *.ngrok.io (see Appendix for snort rules)
- HTTPS TLS connections going to *.ngrok.io

Regularly Monitoring for Updates to Exposed Services

The fact that a service is hosted on the internet, regardless of whether it was port-forwarded through the router or via cloud tunnels, could increase the attack footprint of an organization. For example, although ngrok's ability to generate random subdomains to tunnel web servers (HTTP and HTTPS) can help prevent attackers from blindly guessing the information of their victims, this might not be the case if the webpage is permanently hosted and the user decides to reserve a fixed subdomain for use. Developers who are testing web servers that are integrated to an application (such as an Android APK file or a Windows application) typically use fixed subdomains to avoid the hassle of recompiling just to test the new servers.

The same goes for TCP tunnels with ngrok. Even though ngrok provides a random tunnel and random port to use for free, the tunnel and port number are limited in range. Thus, an attacker could simply scan the tunnels with a specified range of ports to identify the services that are running behind ngrok.

Keeping everything updated reduces the attack surface that cybercriminals could exploit. This also applies to home users, including gamers who use cloud tunneling services to expose game servers.

Best Practices Against ngrok Abuse

We contacted ngrok to let the platform know of our findings from this research. In response, ngrok stated that it applies "a multi-pronged strategy of real-time monitoring, account monitoring, and third-party reporting to detect, isolate, and remove malicious content from our service as quickly as possible."

To prevent malicious activities that abuse its platform, ngrok has implemented certain security enhancementsopen on a new tab, which have been available on versions 2 and 3 of ngrok since April 13, 2022, to users with paid accounts. One of these is its disabling of users' ability to serve HTML content anonymously, that is, all users must register with a confirmed email on file. As an ongoing practice, ngrok also has all free accounts have the origin IP address embedded both in the HTTP headers in the response and in the tunnel URL itself. And because attackers are continuously innovating, ngrok provides an abuse reporting APIopen on a new tab to allow trusted third parties to integrate abuse reporting into their own detection processes.

Within enterprises, ngrok recommends that IT and security personnel block all *.ngrok.io traffic and use custom ingressopen on a new tab to create their own named ngrok entry points, such as developername.tunnel.company.com. This eliminates personal ngrok accounts from the network while allowing centralized policy management across all connections.

Outside enterprises, ngrok recommends that organizations and their security teams use their own domainopen on a new tab, such as tunnel.company.com, to standardize URLs across external systems and apply their own TLS certificate to the traffic. This creates consistent, predictable naming and makes end-to-end encryption the default. In addition, they can add OAuth 2.0 or OpenID Connectopen on a new tab powered by their identity provider to limit ngrok access to authenticated and authorized.

The ngrok management layer also enables security teams to tune their organizations' security postures. Ngrok supports IP restrictionsopen on a new tab on any portion of its service — the agent, tunnel access, the dashboard, and even the API — to limit access to known good IP ranges. The entire ngrok platform implements observability through event subscriptionsopen on a new tab to integrate into security information and event management (SIEM) for near-real-time insight across the entire platform. Finally, security teams can push over-the-wire updates to keep all ngrok agents synchronized.

Conclusion

Cloud tunneling services provide users a very convenient way to expose systems, applications, and services to the internet without going through the trouble of configuring routers, firewalls, web hosting, and domain registration. It is no surprise that the accessibility they afford appeals to development teams and other users, such as technically inclined gamers and home automation enthusiasts.

However, new technologies, especially those that involve exposing services and machines to remote access, need to be scrutinized for potential security implications. While convenient, tunneling services circumvent traditional network security mechanisms, which might enable rogue or disgruntled employees to open a backdoor into the network. Similarly, well-meaning but security-unaware employees might also inadvertently expose critical systems to attackers through unattended tunnel instances.

Malware and phishing attacks using cloud tunneling services, particularly ngrok, have seen a dramatic increase over the past two years. A disadvantage defenders face with regard to cloud tunneling services is that these services make identifying malicious network traffic more difficult for network security teams. Traditional network scanning technologies such as intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) would not be able to flag malicious C&C communication if encapsulated through cloud tunnels. Incident responders would have a difficult time validating exploit and phishing pages hosted on cloud tunneling services because of their transient nature.

While cloud tunneling services have their place, CIOs, CISOs, and cybersecurity personnel should take into consideration the various risks presented by these services and formulate usage policies applicable to their organizations. Some organizations can outright ban the use of these services, especially in the case of businesses that do not really need them. But others, especially those in the middle of digitizing or integrating their systems, might not be able to stop the use of cloud tunnels without hindering ongoing system development and integration projects.

For an organization that needs cloud tunnels, a sit-down meeting between the security team and the development team is a must. The development team must define the exact use case it has for cloud tunnels and lay out the scope and time frame when these services are to be used. This gives the security team proper context to identify whether specific cloud tunneling traffic is valid or not. The security team can also suggest alternatives, such as providing a test network where cloud tunneling traffic can be safely allowed or obtaining paid accounts or subscriptions on cloud tunneling services to enable all available security-oriented features.

Security works best when all stakeholders understand what is at stake. By highlighting the risks of cloud tunnels and the actual use and abuse of these services by malicious actors, cybersecurity and software development teams can put the use of cloud tunnels in their agenda in order to find a suitable arrangement that will minimize risks while allowing the development teams to continue their work.

Appendix

Analyzed Malware SHA-256

SHA-256	Malware family	Detection name
04c584f7dc1c1fa978f59cc966a0664589e7709a7f79b888614e14ef48309e7d	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
116a1bf0810fec723298377ec6b57bd5328d74ba5e3397545e86311873f9e677	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
22ab0a3460b348696d7df493c57c26240b1f72fc5bc06751dce6c300f371698f	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
30b7bf839fabec042c362ae8306b7b699911e8e712494a8d5a5a064db4a3a4ed	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
525c8a4537286abee8f7aa7319d7c65a05452291dee486991f97e06ae3e1332c	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
949ab25990b3bf6677eea51b09839b2fd36ffea939565f449b9c6d4f3d0c138a	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
97416fca437cfbe1d5e0a3cdcb7a47db4cbc05e97cfc861868d066f78a76b07d	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
b5ae89b1f057505f02b7d12f37524acb1b91d11b598dcbdc956f7167dc745fe9	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
e2d1ea986b91e056594a326abc4cf9a76f7094ff8528b66e594c6a6ca57a98c7	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
f5db1c44a4eb1b56114323c2645ca3e56614e1bedc0a89e70fc912388b7c485c	AsyncRAT	Backdoor.MSIL.ASYNCRAT.AZ
4ebceaccaf39fffdd849815fcc75cf85d8cd3326c83220323e331582790233c6	Cobalt Strike	Trojan.Win32.COBALT.SM
c24fad806575b49a6f6f885f3e57c711f319174bcdeedbf70f2cc4aa65bbfc65	Cobalt Strike	Trojan.Win32.COBALT.SM
d1963148b49a5e651aca0a78fa848a3a9950f21c0cc586c13eca4eef352bbddb	Cobalt Strike	Trojan.Win32.COBALT.SM
12b8cb4e6421f2031d619717b0c6d2b4aec205a7270cad493297764970eb5fb7	DarkComet	BKDR_FYNLOS.SMM
2d1747b4b841eb10e54300f387e1bb1d10911fde626179c5b4a7b61e1c850b3a	DarkComet	BKDR_FYNLOS.SMM
35de1d0c1b59bb9099796866c86e7722e273f38d6794ad3e577866d5558dfb80	DarkComet	BKDR_FYNLOS.SMM

a3fe8799e6d0dc83fb57029e322a8f9c04562a18492c20e098d523c584d0a8a8	DarkComet	BKDR_FYNLOS.SMM
ad2266bae9999b84ea79c159dfe606cb8f1c30fa2989629042b058ae351f8bf5	DarkComet	BKDR_FYNLOS.SMMBKDR_FYNLO
c82fe31eb056ba43d7c33295035a23952a51ef5c3b002367d7677eccf8b67cae	DarkComet	BKDR_FYNLOS.SMM
d1f1eddacc07f799773805c971cd49d23c57c4939a09bd7a8d8aa89e580f2178	DarkComet	BKDR_FYNLOS.SMM
e1861f88576a3e310e3feaad7d747425b40510068541d2d4b223c483bf71493c	DarkComet	BKDR_FYNLOS.SMM
ed63fc3c0b01416f9a2126a6ebd82e10e7be526b2035890245027e4b70f4e60c	DarkComet	BKDR_FYNLOS.SMM
f5893afcb7391969fb3d6e16f908fb5b85341d29f92403e85b4515fea2668293	DarkComet	BKDR_FYNLOS.SMM
27c881e3ede8be9316f4d08cad7470b3d5d7c01b63c47fb099b5b5503b169fb3	Metasploit Shellcode	Backdoor.Win32.SWRORT.SMB
8b6dda7b5c7dbcecc2072f67620194d0def87b7bacca66a2b2010b1c78081419	Metasploit Shellcode	Trojan.Win32.METERPRETER.GAJC
9427d2dbe0ae36456656136df394f56ec08381cd6d68d5f12ba126284129e155	Metasploit Shellcode	Backdoor.Win32.SWRORT.SMB
c09f1ae3f86cda5b49da24c3c6942c1edc7577644aa602074ac7a9e236b5e84b	Metasploit Shellcode	Trojan.Win32.METERPRETER.GAJC
460b09da596e354ea9e8207414b9789181a0923ef3c820f6fbf164c2255912db	NanoCore	BKDR_NOANCOOE.SM
46336387c7356587d94318b1f439ee1c47654bf0ac84c534758996a2f4fbf666	NanoCore	BKDR_NOANCOOE.SM
64c79aea781eadec0c3ab32808660ecd9205b910e8c5e8b43f41b19cdef7ad66	NanoCore	BKDR_NOANCOOE.SMUPS
8dc26896a28c36ccc7a93b3435a42954a9325802d4e3ee0731067802ab9d6fd2	NanoCore	BKDR_NOANCOOE.SM
ab1cd62b3fea7cff71d8b2c59515a20f612ad4c82ae470d1995dee55ed3f923e	NanoCore	BKDR_NOANCOOE.SM
bb39887e5668ce591aba14bca153c225183f5d4b0ef5ea0f85c0855505b418d7	NanoCore	BKDR_NOANCOOE.SM

d112e19d34e88c040a70367143569c965cb48dbb1fa36579838c51f8ca9ebe7c	NanoCore	BKDR_NOANCOOE.SM
df0dcc7475e1497ac795bc2a11136c54299221e850373726b69aceaeecf4fa3b	NanoCore	BKDR_NOANCOOE.SM
ec2c954ab2acbb52f79b2cf1f24cbf935e28575900dca2e205b22cd0e5781a16	NanoCore	BKDR_NOANCOOE.SM
f292c9d84f219dbc6821729b92a37b6771c3ac0f3e69f154f37a636d87b176fe	NanoCore	BKDR_NOANCOOE.SM
264bbb47e0b55c06aaa9e558902b6389891d8d17d76ad7eec634f2e358289b81	njRAT	BKDR_BLADABI.SMC
2951818c48bf700f26bb64ff55a4fd190cc752ce201ed8425c7531a728c2c4dd	njRAT	BKDR_BLADABI.SMC
2a58840511e2d1293c8ff7eb25461824ac5734f7e930cecf2af3a102ec7a82c3	njRAT	BKDR_BLADABI.SMC
4b44701bcc0b7d99f309c7704d4c58f3dbe4ea207d7ff915fe0aa22d9975a2d1	njRAT	BKDR_BLADABI.SMC
70db49695386b79dcf21ca2346ee1ebad08b1cc6e49d30839be09aec32e39e7e	njRAT	BKDR_BLADABI.SMC
73c425fc318a0f89a9ca5794355fbb78bd595174cb3a30f871176a3a4d79601a	njRAT	BKDR_BLADABI.SMC
c95e3799e0a9981a4a6bb6ab7c71294a2ecf836b3ca6339bbd756a70010b75ea	njRAT	BKDR_BLADABI.SMC
d4386d967f7e50c8380c9075079590cf7a95d54dc523a50f2bf55d29f799710d	njRAT	BKDR_BLADABI.SMC
eb2966a2181e0df015b7b8e7ae2fe581b6cb505ee3456efa6b5a340edb5e5764	njRAT	BKDR_BLADABI.SMC
ffaf0ab447bce485f4a7db970f3da1fb854c5e6b6df22327af0e02215e92f20f	njRAT	BKDR_BLADABI.SMC

Show more

Ngrok DNS Alerting Snort Rules

- alert udp $HOME_NET any -> any 53 (msg:"ET POLICY DNS Query to a *.ngrok domain (ngrok.com)"; content:"|01 00 00 01 00 00 00 00 00 00|"; depth:10; offset:2; content:"|05|ngrok|03|com|00|"; fast_pattern; distance:0; nocase; classtype:policy-violation; sid:2022641; rev:1; metadata:created_at 2016_03_23, updated_at 2016_03_23;)
- alert udp $HOME_NET any -> any 53 (msg:"ET POLICY DNS Query to a *.ngrok domain (ngrok.io)"; content:"|01 00 00 01 00 00 00 00 00 00|"; depth:10; offset:2; content:"|05|ngrok|02|io|00|"; fast_pattern; distance:0; nocase; classtype:policy-violation; sid:2022642; rev:1; metadata:created_at 2016_03_23, updated_at 2016_03_23;)

Yara Rule

```
import "cuckoo"

rule ngrok_traffic

{

   condition:

      cuckoo.network.http_request(/https:\/\/.*\.ngrok\.io/) or

      cuckoo.network.http_request(/http:\/\/.*\.ngrok\.io/) or

      cuckoo.network.dns_lookup(/\d\.tcp\.ngrok\.io/)

      cuckoo.network.dns_lookup(/.*\.ngrok\.io/)

}
```

Bash Commands

- curl -s https://s3.amazonaws.com/dns.ngrok.com/tunnel.json | jq -r '.[][]'
- for ip in `curl -s https://s3.amazonaws.com/dns.ngrok.com/tunnel.json | jq -r '.[][]'`; do echo "iptables -A OUTPUT -s 0.0.0.0/0 -d $ip -p all -j DROP"; done

---