Pixel-Perfect Trap: The Surge of SVG-Borne Phishing Attacks

By Bernard Bautista and Kevin Adriano

Published: 2025-04-10 · Archived: 2026-04-05 20:06:20 UTC

April 10, 2025 7 Minute Read

Ever thought an image file could be part of a cyber threat? The Trustwave SpiderLabs Email Security team has identified a major spike in SVG image-based attacks, where harmless-looking graphics are being used to hide dangerous links.

This blog post analyzes the various techniques cybercriminals are using to cleverly weaponize these image files in phishing attacks and what your organization can do to prevent these pixel-perfect tricks.

Background of Image-based Attacks

Cybercriminals have long leveraged image-based attacks to evade security defenses. One of the earliest examples of this was image spam in the early 2000s which emerged to bypass traditional text-based detections. Over the following years, attackers adopted new image-based techniques, notably QR-code phishing during the 2010s, which grew significantly and became a widespread threat by 2023.

Also during the 2010s, threat actors have used steganography - the practice of hiding data within another file or media — to conceal malicious code or stolen data within image files.

In 2017, Trustwave SpiderLabs identified another form of image-based attack via SVG files to embed scripts that will download Ursnif malware from a remote resource.

Rise in SVG-based Threats, Driven by PhaaS platforms

Figure 1. Monthly volume of email threats with SVG attachments from our spam traps over the past 12 months (April 2024 – March 2025).

SVG-based attacks have sharply pivoted toward phishing campaigns, with a staggering **1800%** increase in early 2025 compared to data collected since April 2024. A notable surge in campaigns was observed in Q1 of 2025, peaking in March. These are driven largely by the emergence of Attack-in-the-middle (AITM) Phishing-as-a-Service (PhaaS) platforms such as Tycoon2FA, which have significantly amplified the effectiveness and prevalence of these deceptive tactics.

What is an SVG File?

SVG (Scalable Vector Graphics) files are vector-based images commonly used for crisp logos, icons, and graphics due to their ability to scale without losing quality. Unlike typical image formats like JPEG or PNG, SVG files are based on XML (Extensible Markup Language), allowing them to contain interactive elements and scripts. This flexibility has made SVG files increasingly popular across websites, applications, and digital marketing platforms, serving diverse visual needs efficiently.

Example SVG:

The example code below is a benign SVG file that renders the Microsoft logo. It utilizes the element to define the shapes and colors of the logo's graphical components.

```
<svg xmlns="http://www.w3.org/2000/svg" aria-label="Microsoft" viewBox="0 0 512 512">
  <rect width="512" height="512" rx="15%" fill="#fff"/>
  <path d="M75 75v171h171V75z" fill="#f25022"/>
  <path d="M266 75v171h171V75z" fill="#7fba00"/>
  <path d="M75 266v171h171V266z" fill="#00a4ef"/>
  <path d="M266 266v171h171V266z" fill="#ffb900"/>
</svg>
```

Figure 2. Example benign SVG code that renders a Microsoft logo.

Rendered Microsoft logo in SVG:

Figure 3. The rendered SVG Microsoft logo.

How Cybercriminals Exploit SVG Files for Attacks

While SVG (Scalable Vector Graphics) files are widely used in web design and branding, their ability to embed JavaScript also introduces serious cybersecurity risks.

Cybercriminals exploit this feature by inserting malicious scripts directly into SVG files. These scripts can execute automatically upon opening the file, enabling a wide range of cyberattacks, including unauthorized system access, data theft, identity compromise, and leakage of sensitive information.

The primary cybersecurity risks posed by malicious SVG files include:

- Automatic execution of concealed malicious scripts without explicit user interaction.
- Difficulty for conventional security filters and antivirus tools to detect and block threats effectively.
- False sense of safety among users who typically view SVG files as harmless image content

File Comparison: SVG vs. PDF, DOC, HTML

To better understand the threat of SVG phishing, it is helpful to compare it with other common phishing file formats such as PDF, DOC, and HTML. This comparison helps evaluate the relative risks, delivery methods, and effectiveness of each format in bypassing security measures and deceiving users.

SVG Files

SVG phishing is highly effective because SVGs can embed JavaScript that executes automatically. Their harmless appearance and the lack of stringent security checks further heighten their appeal as phishing vectors.

PDF Files

PDF files are frequently employed in phishing attacks due to their ubiquity in business and official communications. Although PDFs can embed malicious links or scripts, executing these threats typically requires

user interaction, such as clicking a link or button within the document. Moreover, PDFs often undergo rigorous scanning by security software, diminishing their effectiveness compared to SVG files.

DOC Files

Microsoft Word documents (DOC) commonly feature phishing attempts using embedded hyperlinks paired with enticing text or images. These links are crafted to appear legitimate and urge users to click, directing them toward phishing websites designed to capture credentials. Unlike SVGs, DOC files do not inherently execute scripts automatically unless part of a macro-based attack and rely heavily on user engagement and trust.

HTML Files

HTML phishing leverages embedded scripts that execute directly in browsers, often involving complex obfuscation techniques. Despite their direct threat potential, users are usually more cautious with HTML attachments due to increased awareness about their associated risks.

Table 1. Summary of Comparison

FEATURE	SVG	PDF	DOC (WORD)	HTML
SCRIPT CAPABILITY	Yes, easily embedded	Limited, typically user-activated	Yes, through macro, but most phishing relies primarily on hyperlinks	Yes, easily embedded
USER TRUST LEVEL	High (perceived as simple images)	High (professional documents)	High (trusted document format)	Medium (users wary of HTML files)
EXECUTION METHOD	Automatic upon opening	Usually requires interaction	Click-based hyperlinks	Automatic or click-based execution

Trustwave MailMarshal

Figure 4. Table featuring Summary of Comparison between SVG, PDF, Doc and HTML.

Browser and Email Client Handling of SVG Files

Web browsers such as Chrome, Firefox, Safari, and Edge natively handle SVG files and automatically execute embedded JavaScript without issuing security alerts. This makes SVG phishing highly effective, as users receive minimal warning about the potential risks.

In contrast, desktop email clients like Outlook and Thunderbird generally do not execute scripts within SVG files. Instead, they prompt users to open these files in an external browser, inadvertently increasing phishing risks by transferring the attack vector to a less secure environment.

Figure 5. SVG-Based Phishing Attack Flow.

In-the-Wild Campaigns

While HTML and PDF attachments remain popular in phishing campaigns due to their versatility, recent activity reveals a notable shift toward the use of SVG files as an alternative delivery mechanism. This lightweight, text-based image format is increasingly exploited by threat actors to embed JavaScript-based redirection, allowing them to act as stealthy intermediaries that funnel victims to credential-harvesting pages while evading traditional security filters.

This technique has been observed across campaigns linked to AiTM PhaaS platforms such as Tycoon2FA, Mamba2FA, and Sneaky2FA—all of which specialize in intercepting credentials and bypassing multi-factor authentication.

In one observed campaign (figure 6), attackers mimic a Microsoft Teams voicemail notification to lure victims into downloading a suspicious attachment. The phishing email carries a subject and body text resembling a legitimate Teams alert.

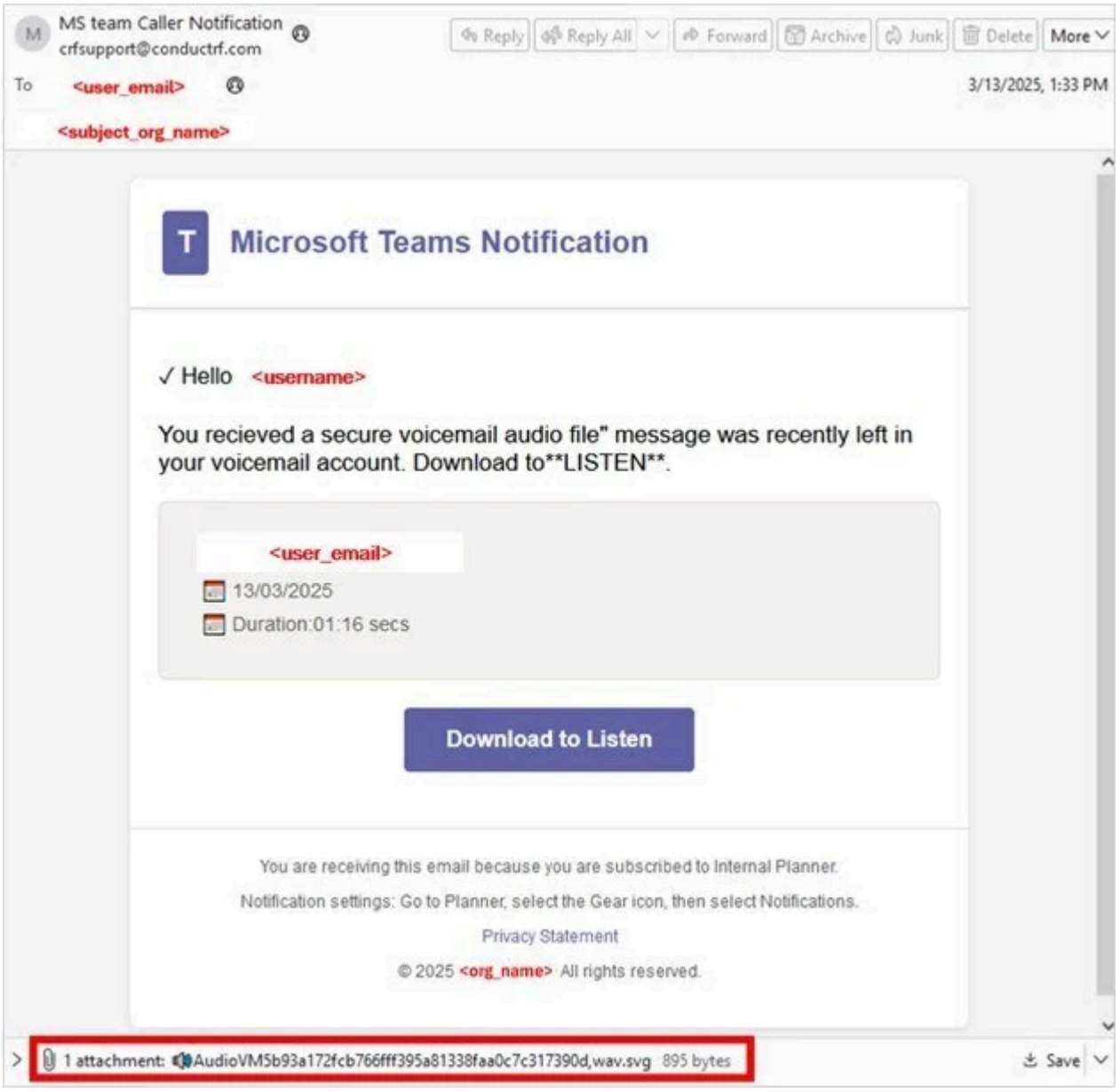

Figure 6. Fake Microsoft Teams alert email with a deceptive voicemail message prompt.

The attachment is deceptively named to appear like an audio file. Despite its .svg extension, the file is crafted to appear like a voice message. When clicked, it executes an embedded redirection code that leads users to a fake Office 365 login page.

This redirection is achieved through the abuse of the SVG element, which allows HTML and JavaScript to run inside the image. The SVG includes obfuscated script content encoded in base64, making it harder for traditional email security tools to detect.

Figure 7. Obfuscated JavaScript inside an SVG file is used to redirect victims to a phishing site.

As soon as the SVG loads, the script dynamically decodes the URL and appends the victim's email as a fragment identifier. This method helps bypass email filters by hiding suspicious code inside an innocent-looking .svg file.

The spoofed landing page is convincingly designed to steal user credentials under the guise of M365 login.

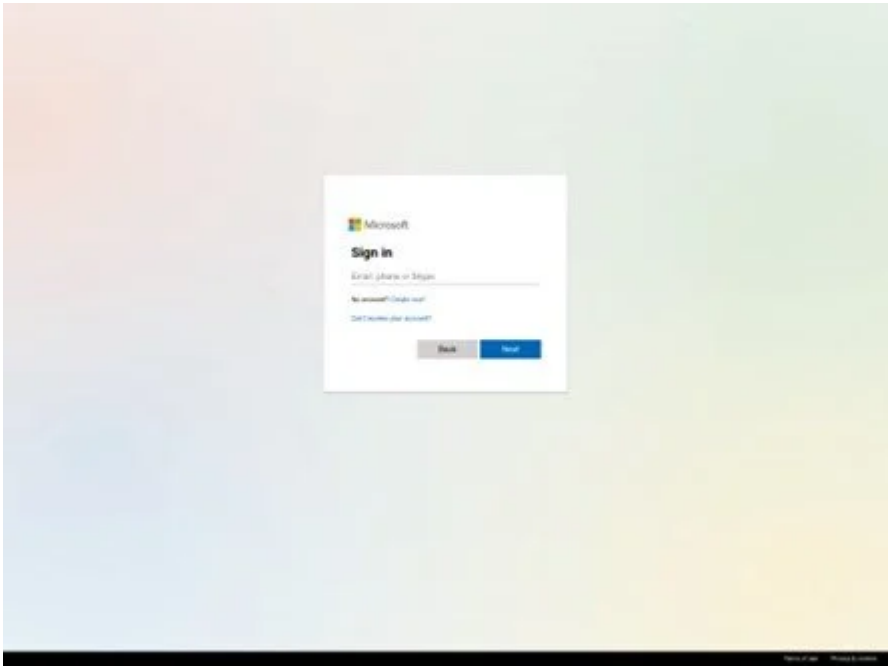

Figure 8. Fake Office 365 login page designed to steal user credentials.

Further investigation linked this campaign to the Mamba2FA Phishing-as-a-Service (PhaaS) platform. Mamba2FA is known for its advanced phishing kits and MFA-bypass capabilities, offering phishing operators all-in-one solutions to intercept and steal credentials even in protected environments.

Other Variants Observed in the Wild

While the core delivery method remains consistent, embedding JavaScript within SVG files to trigger redirects, the implementations vary widely.

Some campaigns use deceptive SVG icons, such as logos or cloud document previews, to lure clicks. Others employ different obfuscation layers, including base64 encoding, character fragmentation, JavaScript encoding tricks, and junk comments inserted throughout the code to evade detection.

The redirect destinations also differ, ranging from fake login pages to credential-harvesting gateways tied to various phishing kits and infrastructure.

Variant 1. Obfuscated SVG script used by Tycoon2FA Phishing-as-a-Service

This SVG phishing variant conceals its malicious URL through multiple obfuscation layers. Appearing as a standard vector graphic with dimensions of 400×250, it contains embedded JavaScript within a CDATA-wrapped block.

CDATA (Character Data) is an XML construct that allows raw text, including special characters like angle brackets and ampersands, to be embedded without being parsed. In this case, it enables attackers to insert executable JavaScript directly into the SVG without breaking the structure, helping hide the payload from basic inspection.

Figure 9. Embedded SVG script with multiple layers of encoding designed to conceal its phishing URL from detection.

The script employs ROT13 encryption, Base64 decoding, and XOR encryption with a specific key to ensure the phishing URL remains hidden until execution. Upon execution, it redirects users to a phishing destination, automatically appending the target email as immediate input on the phishing page.

Variant 2. Logo-Based Staging and Redirect Technique

This SVG file combines graphical content with an embedded redirection script, creating a seemingly harmless image that covertly navigates users to another website.

The file defines an SVG image with specific dimensions *(234×48 pixels)* and a defined *viewBox*. It uses multiple elements to form shapes or text, all rendered in a uniform color.

Figure 10. SVG file combining both the graphical content and embedded redirection script.

Embedded within a CDATA-wrapped block, the JavaScript code defines a redirect function that sets a specific URL target. When the SVG loads in the browser (triggered by *window.onload*), the script automatically executes, immediately redirecting users to the intended phishing destination, which in this case, is a Google Drawings page.

Figure 11 shows an SVG test file we created to demonstrate browser behavior when an SVG file with a redirection script is clicked. Via browser, it initially renders the Microsoft logo and then automatically redirects to a Google Drawings page like figure 10.

Figure 11. Clip showing SVG test file with logo graphics and automatic URL redirection.

This SVG is a prime example of how visual content can serve as a cover for JavaScript-based redirection. While the image renders correctly, its hidden behavior underlines the importance of scrutinizing files that incorporate script elements.

Conclusion

The rise in SVG phishing suggests that threat actors are continuously expanding their tactics to bypass security measures beyond QR codes and the traditional methods, including links, HTML, and document-based attacks. Many of these campaigns are facilitated by phishing kits that operate as PhaaS platforms, making them more

accessible and scalable for cybercriminals. Awareness and proactive security measures are vital in combating this subtle yet increasingly prevalent threat.

To effectively combat this increasingly prevalent threat, users and organizations should:

- **Consider blocking or flagging SVG attachments: e**valuate the option of blocking emails with SVG attachments or, at a minimum, flagging them with a warning.
- **Be cautious with attachments and links:** treat unexpected files and embedded links with suspicion, especially if they come from unknown or unverified sources.
- **Verify authenticity:** double-check senders and content, especially with urgent or unsolicited messages.
- **Train employees regularly:** provide ongoing education on phishing trends and techniques to help users recognize and respond to threats.
- **Use advanced protection:** implement robust filtering and threat-detection systems to proactively block malicious threats like SVG-based attachments. Tools like Trustwave MailMarshal offer layered protection against email threats.
- **Implement MFA methods with extra layers: s**trengthen defenses with phishing-resistant methods like FIDO2 and implement conditional access, continuous authentication, and session monitoring.

Indicators of Compromise

```
hxxps[://]ut[.]sxbmjefh[.]ru/I6wx84s/
```

```
hxxps[://]docs[.]google[.]com/drawings/d/1e6oBFLaz3YRncI8qZ--Mg7yh8Uzw8XK0uW5l-z-khKc/preview?pli=1
```

```
hxxps[://]grado33closet[.]com/n/?
c3Y9bzM2NV8xX25vbSZyYW5kPVl6WlpSVGs9JnVpZD1VU0VSMDQwMzIwMjVVNDEwMzA0MDM=#
```

Source: https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/pixel-perfect-trap-the-surge-of-svg-borne-phishing-attacks/