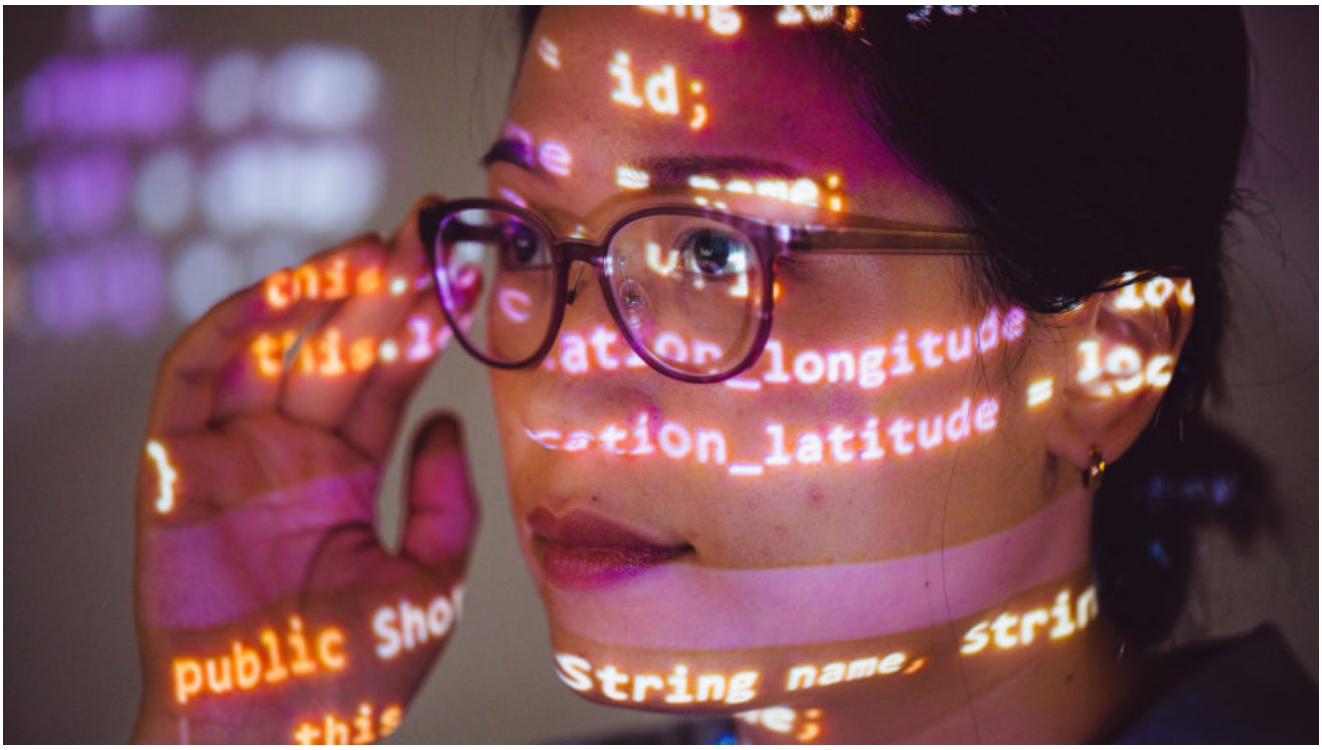
Evasive maneuvers: HTML smuggling explained

blog.malwarebytes.com/explained/2021/11/evasive-maneuvers-html-smuggling-explained/

Jovi Umawing November 15, 2021

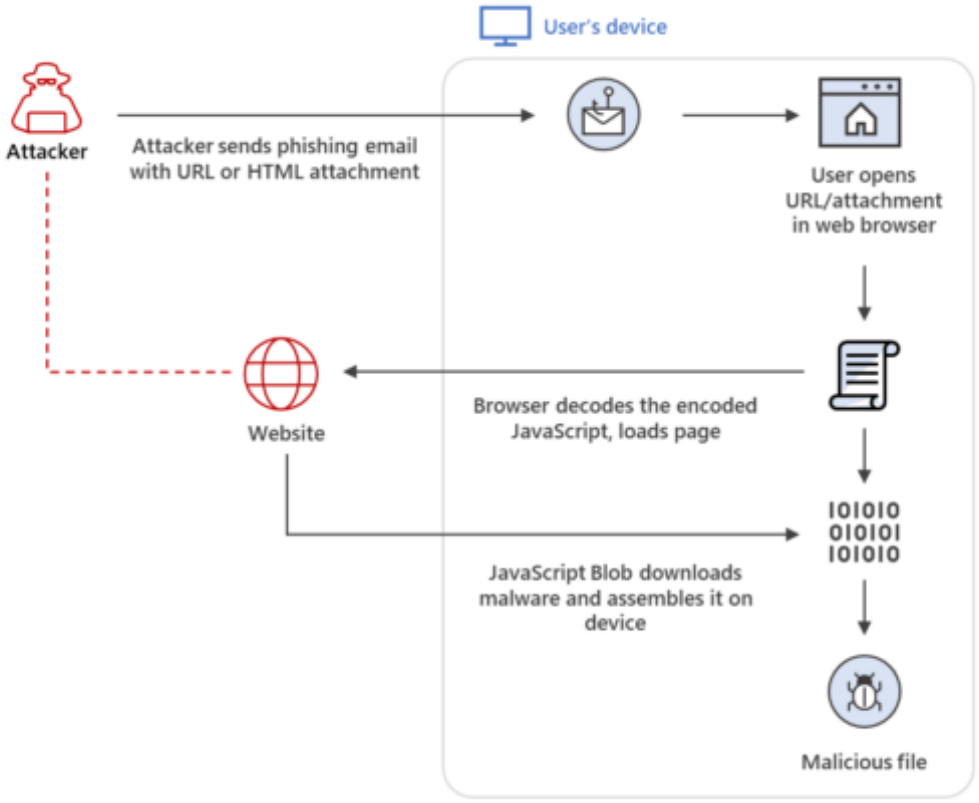


Microsoft Threat Intelligence Center (MSTIC) last week disclosed "a highly evasive malware delivery technique that leverages legitimate HTML5 and JavaScript features" that it calls HTML smuggling.

HTML smuggling has been used in targeted, spear-phishing email campaigns that deliver banking Trojans (such as Mekotio), remote access Trojans (RATs) like AsyncRAT/NJRAT, and Trickbot. These are malware that aid threat actors in gaining control of affected devices and delivering ransomware or other payloads.

MSTIC said the technique was used in a spear-phishing attack by the notorious NOBELIUM, the threat actor behind the noteworthy, nation-state cyberattack on SolarWinds.

How HTML smuggling works



An overview of HTML smuggling (Source: Microsoft)

What is HTML smuggling?

HTML smuggling got its name from the way attackers smuggle in or hide an encoded malicious JavaScript blob within an HTML email attachment. Once a user receives the email and opens this attachment, their browser decodes the malformed script, which then assembles the malware payload onto the affected computer or host device.

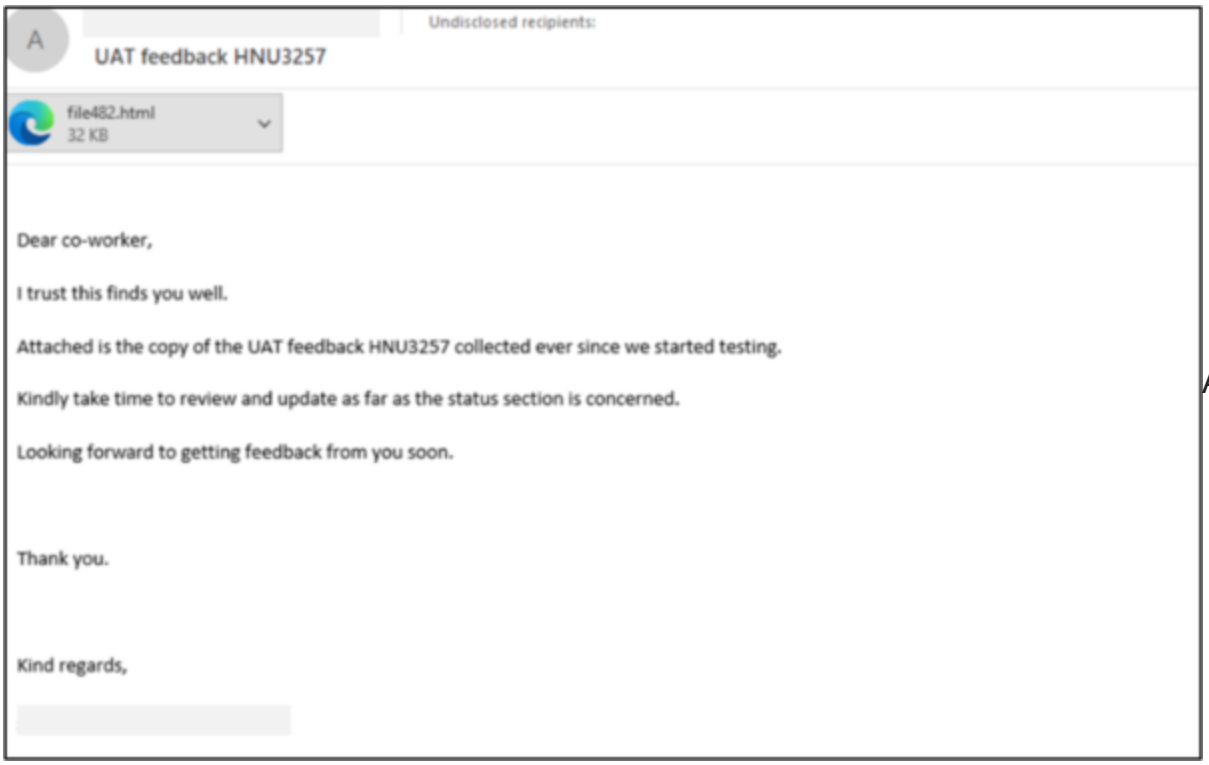
Usually, malware payloads go through the network when someone opens a malicious attachment or clicks a malicious link. In this case, the malware payload is created within the host. This means that it bypasses email filters, which usually look for malicious attachments.

HTML smuggling is a particular threat to an organization's network because it bypasses customary security mitigation settings aimed at filtering content. Even if, for example, an organization has disabled the automatic execution of JavaScript within its environment—this could stop the JavaScript blob from running—it can still be affected by HTML smuggling as there are multiple ways to implement it. According to MSTIC, obfuscation and the many ways JavaScript can be coded could evade conventional JavaScript filters.

HTML smuggling isn't new, but MSTIC notes that many cybercriminals are embracing its use in their own attack campaigns. "Such adoption shows how tactics, techniques, and procedures (TTPs) trickle down from cybercrime gangs to malicious threat actors and vice

versa … It also reinforces the current state of the underground economy, where such TTPs get commoditized when deemed effective."

Some ransomware gangs have already started using this new delivery mechanism, and this could be early signs of a fledgling trend. Even organizations confident with their perimeter security are called to double back and take mitigation steps to detect and block phishing attempts that could involve HTML smuggling. As we can see, disabling JavaScript is no longer enough.



A sample of an email that uses HTML smuggling. This is part of a Trickbot spear-phishing campaign. (Source: Microsoft)

Staying secure against HTML smuggling attacks

A layered approach to security is needed to successfully defend against HTML smuggling. Microsoft suggests killing the attack chain before it even begins. Start off by checking for common characteristics of HTML smuggling campaigns by applying behavior rules that look for:

- an HTML file containing suspicious script
- an HTML file that obfuscates a JS
- an HTML file that decodes a Base64 JS script
- a ZIP file email attachment containing JS
- a password-protected attachment

Organizations should also configure their endpoint security products to block:

- JavaScript or VBScript from automatically running a downloaded executable file
- Running potentially obfuscated scripts
- Executable files from running “unless they meet a prevalence, age, or trusted list criterion”

BleepingComputer recommends other mitigating steps, such as associating JavaScript files with a text editor like Notepad. This prevents the script from actually running but would let the user view its code safely instead.

Finally, organizations must educate their employees about HTML smuggling and train them on how to respond to it properly when encountered. Instruct them to never run a file that ends in either *.js* or *.jse* as these are JavaScript files. They should be deleted immediately.

Stay safe!