Rise of Oriental Gudgeon May 6, 2025

urlscan.io/blog/2025/05/06/oriental-gudgeon/

About

The urlscan.io Blog covers announcements, product news, and tutorials.

Contact

You can contact us at info@urlscan.io for general inquiries and at sales@urlscan.io for questions around our commercial products.

If you need technical support contact us at support@urlscan.io.

For matters of security, please see our Security Page.

Updates

For updates and announcements, subscribe to our newsletter.

Follow us on Twitter:

Follow @urlscanio

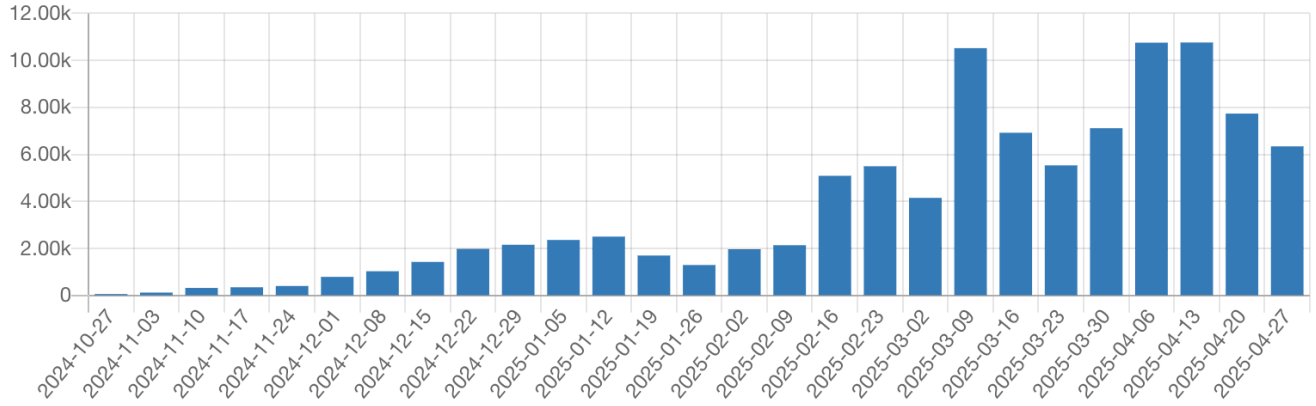
> This is the first ever research-oriented post on the urlscan blog. Our goal with these posts is to cover malicious activity we have not seen covered by other researchers. Will will also showcase how urlscan.io and urlscan Pro can be used to track the types of activities we cover in these posts.

Since October 2024, we have observed a phishing kit impersonating dozens of Japanese commercial entities, primarily companies in the financial services sector. The phishing kit will impersonate the website of these organizations and their brands with the goal of obtaining valid login credentials of legitimate users of these sites.

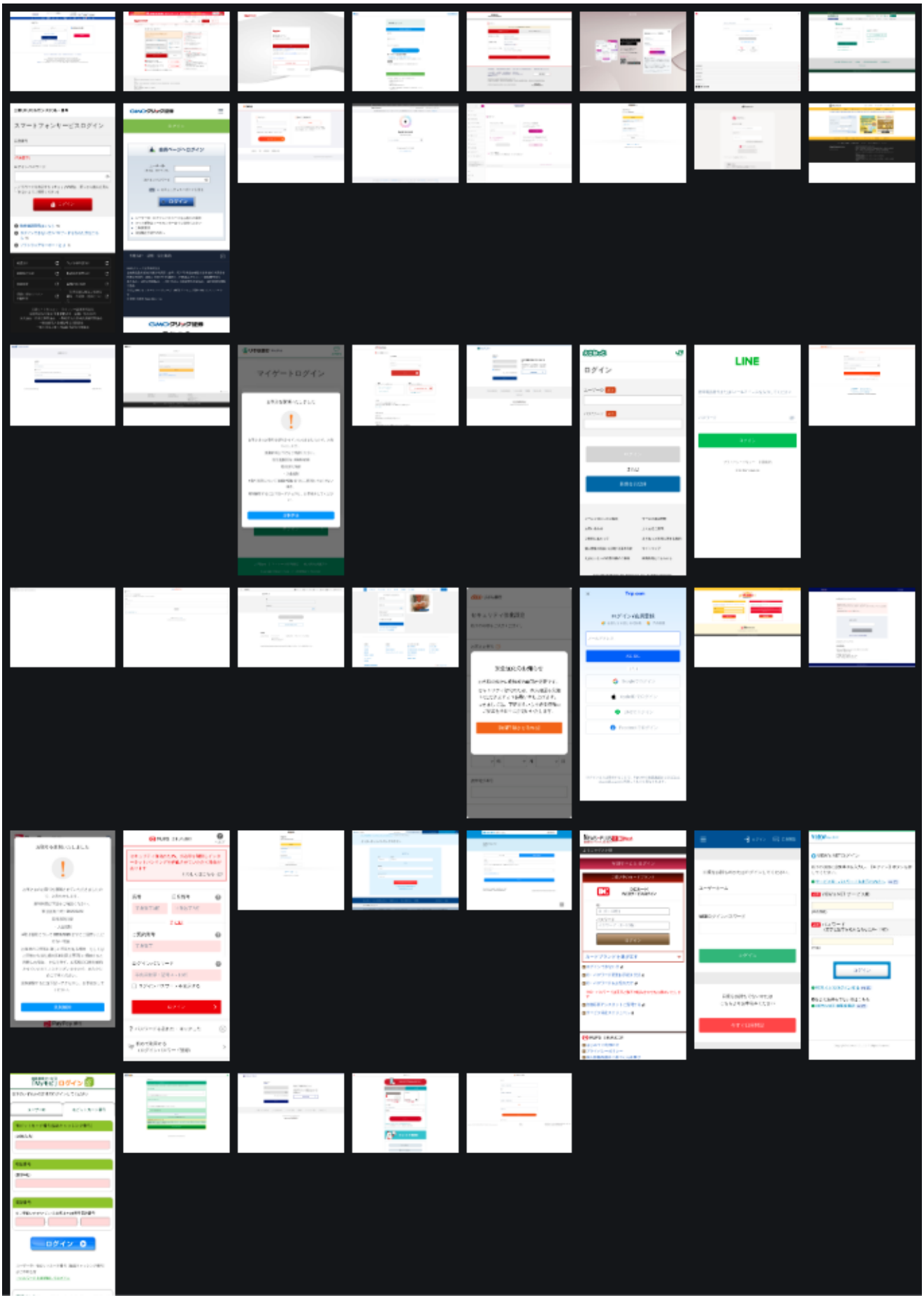
We are currently tracking this activity under the name *Oriental Gudgeon* due to its suspected Chinese origin. Oriental Gudgeon has recently expanded its targeting to include more than 40 Japanese companies.

In this blog post, we will highlight the timeline of Oriental Gudgeon's activity, the organizations being targeted, its attack flow, and how urlscan.io can be used to discover and analyze its activity.

Targeting and Timeline



Weekly website scans observed on urlscan Pro


ステートフォンモービスログイン
マイゲートログイン
ログイン
LINE

Screenshots of landing pages captured on urlscan Pro

A list of organizations that have been targeted by Oriental Gudgeon we've observed (as of 2025/05/02) is:

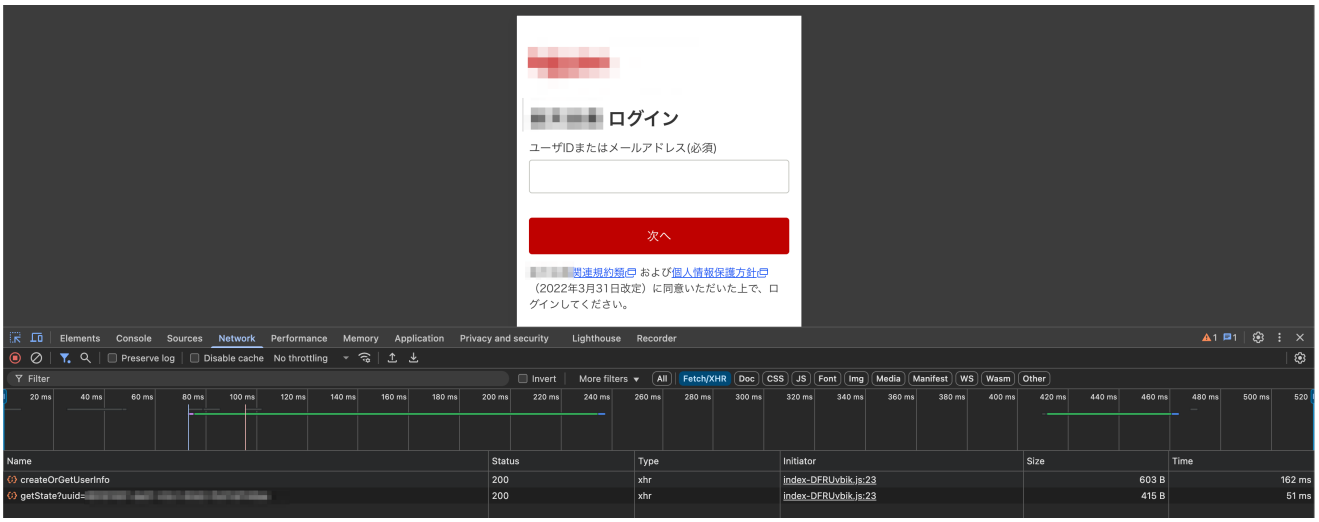
- AEON Card
- Aiful
- Amazon
- American Express
- ANA
- Apple
- au Jibun Bank
- au PAY Card
- DMM.com
- Epos Card
- GMO Click Securities
- Hokuyo Bank
- JACCS
- JAL Card
- JCB
- JR East
- LINE
- Mitsubishi UFJ Bank
- Mitsubishi UFJ Morgan Stanley Securities
- Mitsubishi UFJ Nicos
- Mizuho Bank
- Monex Securities
- N-Planet
- NHK
- Nomura Securities
- NTT Docomo
- Orico
- Paidy
- PayPay
- PayPay Bank
- Rakuten
- Rakuten Securities
- Resona Bank
- Saison Card
- SBI Net Bank
- SBI Securities
- SMBC Mobit

- Sumitomo Mitsui Card
- Trip.com
- UC Card
- Yodobashi Camera
- Yokohama Bank

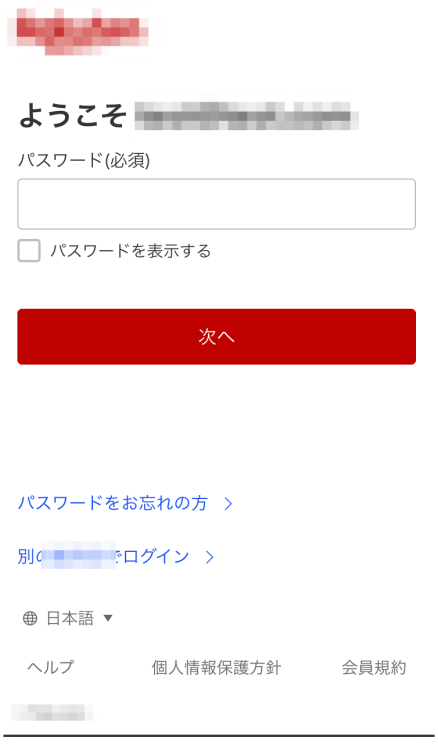
Attack Flow

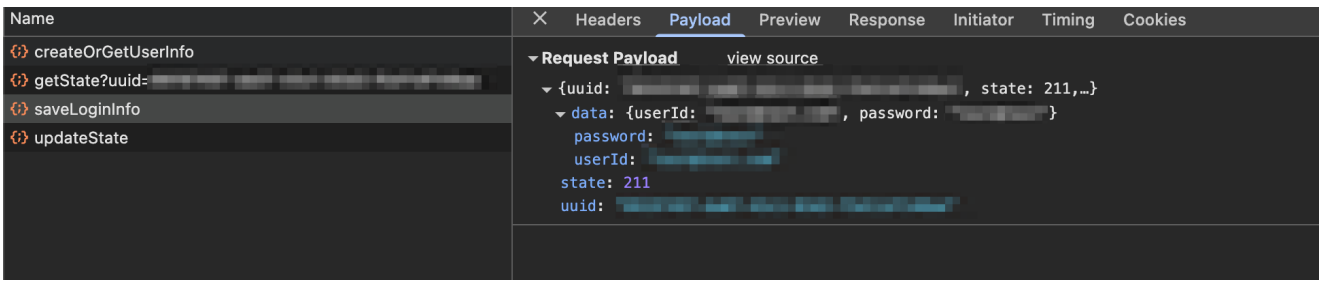
- Targeting starts with a phishing e-mail:
  A typical subject line might contain `アカウントへの不審アクセスに関する重要なお知らせ` which translates to `Important notice regarding suspicious access to your account`.
- Phishing website anatomy:
  - A phishing website is a Vue based SPA and it has two parts, a landing component, and phishing components.
  - Landing component: controls access to the site based on several visitor attributes (see the following Unveiling Cloaks section for details). Only allows Japanese victims to access the site.
    - `POST /visitors/info/createOrGetUserInfo`: creates or gets a victim's information in their backend. Assigns a UUID per victim.
    - `GET /visitors/info/getState?uuid={UUID}`
  - Phishing components: steal credentials, PIN code, etc.
    - `POST /visitors/info/saveLoginInfo`: saves victim's login information.
    - `POST /visitors/info/saveAccountInfo`: saves victim's account information. (Optional)
    - `POST /visitors/info/saveVerificationData`: saves victims' verification data. (Optional)
    - `POST /visitors/info/savePinCode`: saves victims' pin code. (Optional)
    - `POST /visitors/info/savePasscode`: saves victims' passcode. (Optional)
    - `POST /visitors/info/saveCreditCardInfo`: saves victims' credit card information. (Optional)
    - Etc.

For example, a phishing email like this navigates a victim to a phishing website impersonates a Japanese credit card service:



Landing component: `POST /visitors/info/createOrGetUserInfo`





Phishing component: `POST /visitors/info/saveLoginInfo`

Phishing component: `POST /visitors/info/saveCreditCardInfo`

Unveiling Cloaks

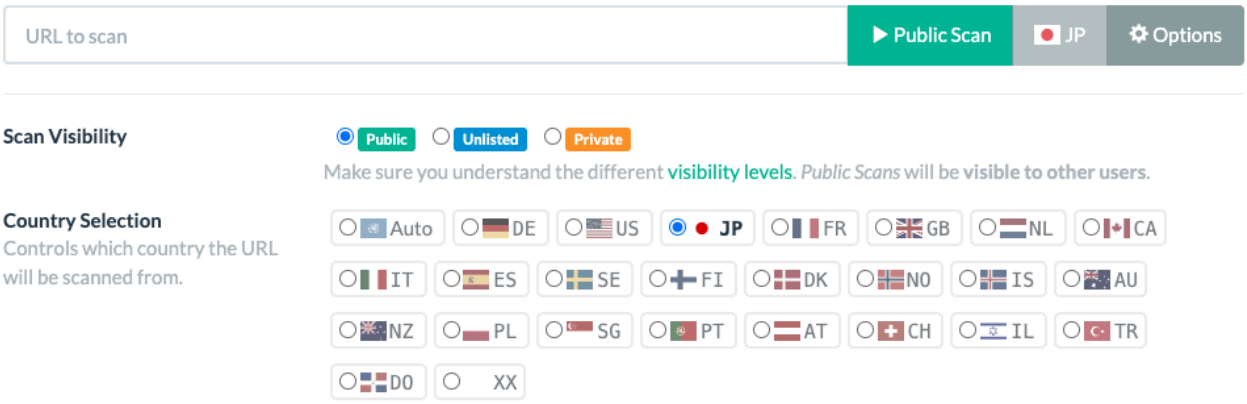
Oriental Gudgeon uses several cloaking techniques:

- IP based geo blocking: redirects the user to a benign website if the user is not located within Japan.
- User-info blocking: cloaks itself (shows 404 page) by closing the `/visitors/info/createOrGetUserInfo` API endpoint.

Fortunately, the urlscan.io platform contains multiple features to bypass these blocking techniques.

Geo Blocking

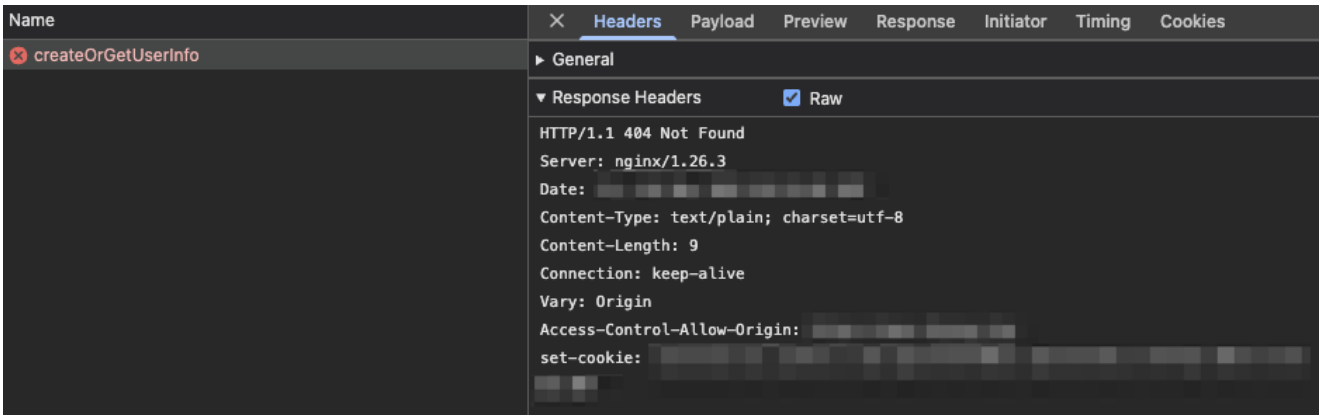
For scanning websites on urlscan.io, you can choose the scanner country:



To bypass Oriental Gudgeon's geofence, you should select the Japan (JP) scanning location. This feature is available for all urlscan.io users. Customers of the urlscan Pro platform are also able to use residential IP addresses to analyze suspicious URLs.

User-info Blocking

Oriental Gudgeon closes its operation windows by disabling the `POST /visitors/info/createOrGetUserInfo` API endpoint. More specifically, it changes to return 404 from the endpoint:



It leads to showing a blank 404 page:

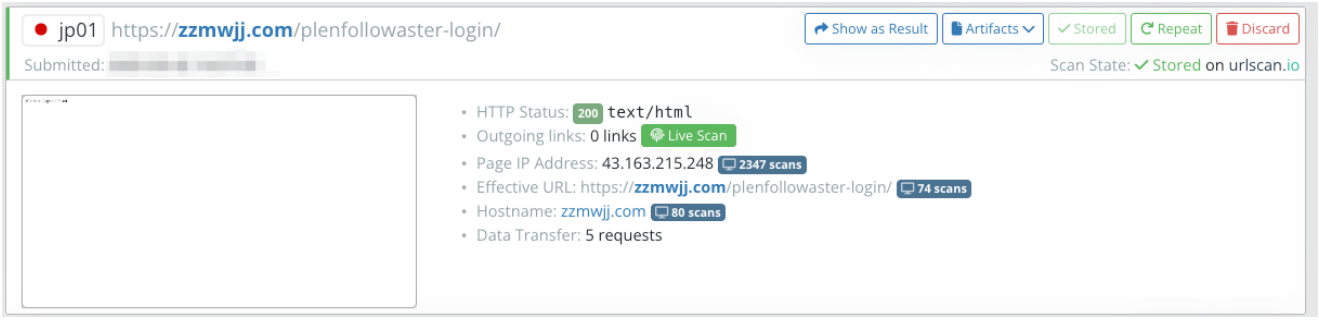
urlscan Pro recently introduced a new feature that allows Custom Javascript injection for Live Scanning, which is available for customers on the urlscan Pro platform. Using our custom Javascript snippet we can bypass the block because of flaws in Oriental Gudgeon's access control management.

```
async function Qi() {
  try {
    const e = localStorage.getItem("uuid");
    if (e) {
      const t = new Gi({
        type: "user",
        uuid: e,
        isNewUser: !1,
      });
      if (!t.isConnected())
        t.connect()
          .then(() => {
            const s = Ls(Fs);
            s.provide("socketClient", t), s.use(Ns()), s.mount("#app");
          })
          .catch((s) => {
            console.error("WebSocket连接失败", s);
          });
      else {
        const s = Ls(Fs);
        s.provide("socketClient", t), s.use(Ns()), s.mount("#app");
      }
    } else {
      const t = await le.post("/visitors/info/createOrGetUserInfo", {
          currentState: 2,
          browserInfo: ip(),
          domain: window.location.hostname,
          codeName: "日本rakuten乐天证券",
          buttons: {
            skip: ["2", "31", "5", "7", "9", "14", "53"],
            reject: ["2", "31", "5", "7", "9", "14", "53"],
          },
          views: [],
          extraData: {
            phonemessage: "客户电话|验证电话/或只输入客户电话",
          },
        }),
        { code: s, data: n } = t;
      if (s === 1e3) {
        localStorage.setItem("uuid", n.uuid),
          localStorage.setItem("deductionStatement", n.deductionStatement),
          localStorage.setItem("price", n.price),
          localStorage.setItem("code", n.code),
          n.extraData &&
            n.extraData.customCaptchaOptions &&
            JSON.stringify(n.extraData.customCaptchaOptions) !== "{}" &&
            localStorage.setItem(
              "customCaptchaOptions",
              JSON.stringify(n.extraData.customCaptchaOptions)
            );
        const o = new Gi({
          type: "user",
```

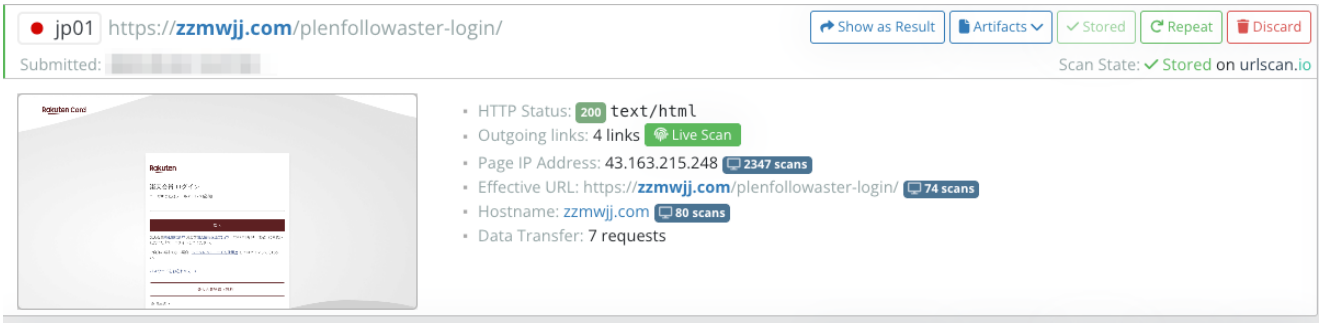
```
          uuid: n.uuid,
          isNewUser: !0,
        });
        if (!o.isConnected())
          o.connect()
            .then(() => {
              const i = Ls(Fs);
              i.provide("socketClient", o), i.use(Ns()), i.mount("#app");
            })
            .catch((i) => {
              console.error("WebSocket连接失败", i);
            });
        else {
          const i = Ls(Fs);
          i.provide("socketClient", o), i.use(Ns()), i.mount("#app");
        }
      }
    }
  } catch (e) {
    console.error("应用初始化失败:", e);
  }
}
```

A part of unminified JavaScript code we analyzed

It uses the local storage to persistent UUID (`const e = localStorage.getItem("uuid");`) and calls `POST /visitors/info/createOrGetUserInfo` if there is no UUID in there. By setting a crafted UUID in the browser's local storage via the custom eval script injection, we can make the check pass.



urlscan Pro: Live scan of a malicious site without any extra options



urlscan Pro: Live scan of the same site using a custom evalScript injection

Attribution

Our current working theory is that the Oriental Gudgeon threat actor is a native Chinese speaker. This theory is based on several artifacts we discovered while analyzing their phishing kit:

- Their JavaScript code (see snippet above) contains the error message 应用初始化失败 which is written in Simplified Chinese. Request payloads also contain Simplified Chinese, for example 客户电话.
- The initial phishing email was sent from Chinese networks. There has been a recent shift towards using residential proxies though. (Spamhaus observes the same trend)

Tracking - urlscan Pro

We have shared our tracking rules for Oriental Gudgeon within the urlscan Pro portal through a Saved Search. Customers of the urlscan Pro platform can get updated results using our hunting rules: urlscan Pro - Oriental Gudgeon.

Appendix 1: IOCs

To date we have observed the following domains as part of this activity.