

CoCo flagged extension position 500->638:

500 Doglbfdapmgcaplipnjkmebembonbcp FP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

SET in bg and GET in cs only for if/else no impact

```
chrome.runtime.onMessageExternal.addListener(  
  (request, sender, sendResponse) => {  
    if (request.user === true) {  
      chrome.storage.sync.set(request);  
    } else {  
      chrome.storage.sync.set(request);  
    }  
  }  
);
```

cs.js

```
chrome.storage.sync.get(["user"], (result) => {  
  if (result.user) {  
    chrome.storage.sync.get(["status"], (result) => {  
      status_available = result.status;  
    });  
  
    if (status_available === "paused" || status_available ===  
"deleted") {  
      isLoggedIn = false;  
    } else {  
      isLoggedIn = true;  
    }  
  
    chrome.storage.sync.get(["email"], (result) => {  
      email = result.email;  
    });  
  } else {  
    isLoggedIn = false;  
  }  
});
```

501. Dohfcmejkdaniblnkoiommmagmdklelcc (same as Allbpdjnlppejnnngnhecjgmplkediald) TP

from Document_element_href to chrome_storage_local_set_sink

Mouse movement on all sites changes the storage also used as url part of storage

Mouse movement changed cursorADD which then is stored

```
t = t + cursorADD + ".mc_default { cursor: url(" + dSrc + "), default  
!important; } "  
...
```

```
document.addEventListener("mousemove", nicecursorElement);
```

```
let nicecursorElement = function(e) {
  try {
    nicecursorManager(document.elementFromPoint(e.clientX,
e.clientY)),
    nicecursorManager(document.elementFromPoint(e.clientX + 10,
e.clientY + 10)),
    nicecursorManager(document.elementFromPoint(e.clientX - 10,
e.clientY - 10));
  } catch (e) {}
}
function nicecursorManager(e) {
  if (typeof e === "Element") {
    let t = getComputedStyle(e).cursor;
    if ("pointer" === t || "default" === t || "auto" === t) {
      "auto" === t && (t = "default");
      let n = e.classList,
          o = e.nodeName,
          c = e.style.cursor;
      if (0 !== n.length && "" === c) {
        let e = o;
        for (let t = 0; t < n.length; t++) e += "." + n[t];
        "default" === t ? cursorADD += e + "," : pointerADD +=
e + ",", nicecursorstyleManager("create")
      } else n.contains("mc_" + t) || n.add("mc_" + t)
    }
  }
}
```

```
nicecursorstyleManager = function(e) {
  var check_popup_page =
document.body.contains(document.getElementById("use_system_cursors"));
  cur_storage.get(local_values, function(data) {
    var default_curSize = data.default_curSize;
    var pointer_curSize = data.pointer_curSize;
    var dSrc = data.default_cursor_result;
    var pSrc = data.pointer_cursor_result;
    var switch_status = data.switch_status;
    var cssElm = data.css_elm;
    if (switch_status === "false") {
      cursors_style_code = "";
      cssElm.innerHTML = "";
    }
  });
}
```

```

document.querySelectorAll('[cursors="cursors_style_code"]').forEach(el
=> el.remove());

    }
    if (e === "create" && switch_status === "true") {
        let t = "";
        if (typeof dSrc !== "undefined" && dSrc.length > 0)
t = t + cursorADD + ".mc_default { cursor: url(" + dSrc + "), default
!important; } ";
        if (typeof pSrc !== "undefined" && pSrc.length > 0) t = t +
pointerADD + ".mc_pointer { cursor: url(" + pSrc + "), pointer
!important; } ";
        if (!check_popup_page) {
            if (typeof cursors_style_code !== "undefined" &&
cursors_style_code == t) {
                cssElm.innerHTML = t;
            } else {

document.querySelectorAll('[cursors="cursors_style_code"]').forEach(el
=> el.remove());

                cursors_style_code = t;
                cssElm = document.createElement("style");
                cssElm.rel = "stylesheet";
                cssElm.setAttribute("cursors",
"cursors_style_code");
                cssElm.innerHTML = t;
                document.head.appendChild(cssElm);
            }
        }
    }
    if (e === "remove") cssElm.innerHTML = "";
    cur_storage.set({
        css_elm: cssElm
    });
});

```

502. Doiccohjdffcjejcpcoddcolekfgipkna FP

from fetch_source to fetch_resource_sink

Minified code, hardcoded ulr and response used as seconds fetch

```

fetch("https://www.crunchyrollwatchparty.com/socket/ext-config", {
    method: "POST",
    mode: "cors",

```

```

        cache: "no-cache",
        credentials: "same-origin",
        headers: { "Content-Type": "application/json" },
        redirect: "follow",
        referrerPolicy: "no-referrer",
        body: JSON.stringify({ name: o.name, version: o.version,
verify: chrome.runtime.id, timeZone: t })),
    })
    .then(function (e) {
        return e.json();
    })
    .then(function (e) {
        fetch(e)
            .then(function (e) {
                return e.json();
            })
            .then(function (e) {
                var t = { code: e.code, config: e.config, lastDate:
JSON.stringify(new Date()) };
                chrome.storage.local.set({ data: t });
            })
            .catch(function (e) {
                return console.log(e);
            });
    })

```

503 dojgclmhkeihlmhjfdichlnbclokdocm TP

from cs_window_eventListener_message to chrome_storage_sync_set_sink

Attacker store data and later dispatch event with it

```

window.addEventListener(
    "message",
    function (e) {
        if (e.data.msg == "快上车") {
            chrome.storage.sync.get("changeArray", ({ changeArray }) => {
                if (changeArray && changeArray.length >= 1) {
                    sendmsg(changeArray);
                    chrome.storage.sync.set({ changeArray: [] });
                }
            });
        }
        if (e.data.msg == "start sync") {
            let cardId = e.data.cardId;
            chrome.storage.sync.set({ cardId });
        }
    }
)

```

```

        localStorage.setItem("_syncChromeCardId", cardId);
    }
    if (e.data.msg == "lose sync") {
        chrome.storage.sync.remove("cardId");
        localStorage.removeItem("_syncChromeCardId");
    }
},
false
);
chrome.storage.sync.get("cardId", ({ cardId }) => {
    if (cardId) {
        localStorage.setItem("_syncChromeCardId", cardId);
    }
});

function sendmsg(msg) {
    const cardId = localStorage.getItem("_syncChromeCardId");
    const msgBus = document.querySelector("#msgBus" + cardId);
    if (msgBus == null) {
        console.log("have no bus");
        return;
    }
    let evt = new Event("input");
    msgBus.value = JSON.stringify(msg);
    msgBus.dispatchEvent(evt);
}

```

504. DonkdpbffcemjgchkdIkhpknemlgkl TP

from cs_window_eventListener_message to chrome_storage_local_set_sink. Used for messages and dom manipulation

```

chrome.storage.local.get('attributes', function(db) {
    if (typeof db.attributes !== 'undefined') {
        var attr = db.attributes;
        elements?.username?.setAttribute('value', attr.username);
        elements?.password?.setAttribute('value', attr.password);
    }
    if (typeof elements.submit !== 'undefined') {
        if (elements.submit !== null) {
            elements.submit?.click();
            chrome.runtime.sendMessage({
                action: 'cleanup',
                data: db.attributes
            });
        }
    }
});

```

```

    }
}

```

Custom script injected by bg script files: ['js/provider-script.js'],

```

window.addEventListener('message', function(e) {
    if (e.type === 'message' && e.data.type === 'LAUNCH_PROVIDER') {
        chrome.storage.local.set({ attributes: e.data.attributes },
function() {
            chrome.runtime.sendMessage({
                action: 'inject',
                data: location.href
            });
        });
    }
});
});

```

Bg.js

```

chrome.runtime.onMessage.addListener(function({action, data}) {
    // exec: inject command to run provider script
    if (action === 'inject') {
        chrome.tabs.query({currentWindow: true, active: true},
function(tabs) {
            chrome.scripting.executeScript({
                target: {tabId: tabs[0].id, allFrames: true},
                files: ['js/provider-script.js'],
            });
        });
    }
});
}

```

files: ['js/[provider-script.js](#)']

505. Dooiopjfnhlmmmdkdepajfipfhlcmjgp TP

from storage_local_get_source to sendResponseExternal_sink

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

Set a lot of storage props and send backgreat,

callback({onboarding: storage.onboarding, settings: storage.settings});

```

chrome.runtime.onMessageExternal.addListener(function (req, sender,
callback) {
    if (req.cmd === "extensionCheck") {
        callback({ exists: true, version:
chrome.runtime.getManifest().version });
        return false;
    } else if (req.cmd === "getLegacySettingsOnce") {

```

```

chrome.storage.local.get({ haveRetrievedLegacySettings: false,
onboarding: "", settings: "" }, function (storage) {
    if (!storage.haveRetrievedLegacySettings) {
        chrome.storage.local.set({ haveRetrievedLegacySettings: true },
function () {
            callback({ onboarding: storage.onboarding, settings:
storage.settings });
        });
    } else {
        callback({ alreadyDone: true });
    }
});
return true;
}

```

506. Dophpcobbjngempeadfhkjleblemfjll FP
from fetch_source to chrome_storage_local_set_sink
Hardcoded url in config.json:

```

let tcid, tcs;
fetch(chrome.runtime.getURL('config.json')).then(r=>r.json()).then(j=>{
tcid=j.tcid;tcs=j.tcs;})

```

```

function checkTwitchToken(callback) {
    if (syncConfig.twitch && syncConfig.twitch.expires > new
Date().getTime()) {
        // console.log(syncConfig.twitch);
        callback && callback();
        return;
    };

    fetch(
        twitchAuthURL.replace('{tcid}', tcid).replace('{tcs}', tcs),
        {
            method: 'POST'
        }
    )
        .then(r => r.json())
        .then(result => {
            let dNow = new Date();
            dNow.setSeconds(result.expires_in);
            syncConfig.twitch = {
                token: result.access_token,
                expires: dNow.getTime()
            }
        })
}

```

```

    }
    saveSyncStorage()
    callback && callback();
  })
}

```

```

function updateTwitchIds(callback) {
  if (syncConfig.channels && syncConfig.twitch &&
syncConfig.twitch.token) {
    let promises = [];
    syncConfig.channels.forEach(channel => {
      if(localConfig.twitchUserIds[channel]) return;
      promises.push(fetch(twitchUserURL.replace('{user}',
channel),
        {
          method: 'GET',
          headers: {
            'Accept': 'application/vnd.twitch.tv.v5+json',
            'Client-Id': tcid,
            'Authorization': 'Bearer ' +
syncConfig.twitch.token
          },
        })
      .then(r => r.json())
      .then(result => {
        if(result.error) {
          syncConfig.twitch = null;
          saveSyncStorage();
          return callback(false);
        }
        localConfig.twitchUserIds[channel] =
result.data[0].id;
      })
    )
  })

  Promise.allSettled(promises).then(([result]) => {
    saveLocalStorage();
    callback(true);
  });

} else {
  callback(false);
}

```



```
}
}
```

507. Dpaojegkimhndjkkgiaookhckojbmakd FP

from Document_element_href to JQ_obj_html_sink

Hardcoded API URL. Attacker controls word parameter but can query that public API directly. No benefit from exploiting extension. .html() receives jQuery object, not raw HTML.

```
var word = $(this).find('.concept_light-readings
.text').text().trim();
words[words.length] = word;
```

```
chrome.runtime.sendMessage(null, {words: words}, function(html) {
  result = parseOJAD(html);
  console.log(result);
  d.resolve();
});
```

```
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
  fetch_ojad(request.words).then(sendResponse);

  return true;
});
```

```
async function fetch_ojad(words)
{
  let resp = await
fetch('https://www.gavo.t.u-tokyo.ac.jp/ojad/search/index/display:print
/sortprefix:accent/narabi1:kata_asc/narabi2:accent_asc/narabi3:mola_asc
/yure:visible/curve:invisible/details:invisible/limit:500/word:' +
encodeURIComponent(words.join(',')));
  return await resp.text();
}
```

```
for (var i in result[word])
{
  var main = $('<div/>');
  for (let reading of result[word][i].data[0])
  {
    main.append(readingToHtml(reading));
  }
}
```

508. Dpenkpajllccnkmbgdejbeoealnoopl TP

from storage_sync_get_source to sendResponseExternal_sink

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink
GET [prefs.client.id](#) to attacker also SET for this. Uses prefs var global to set indirectly
Updatepres to SET the storage

```
chrome.runtime.onMessageExternal.addListener(async (req, sender, resp)
=> {
    switch (req.action) {
        case "give-pdf":
            if (!prefs.client || !prefs.client.curr ||
!prefs.client.subs.includes("pdf")) {
                resp(null);
                return;
            }
            resp(prefs.client.id);
            break;
        case "update-page":
            sendMessage({ action: "update-page" }, { currentWindow:
true, active: true });
            resp(true);
            break;
        case "update-client":
            prefs.client = req.data;
            prefs.client.next = await getNext();
            updatePrefs();
            resp(true);
            break;
        default:
            console.error(`unknown message: MSG:
${JSON.stringify(message)}, SNDR: ${JSON.stringify(sender)}`);
            resp(null);
    }
}

function updatePrefs() {
    chrome.storage.sync.set({ "font_prefs": JSON.stringify(prefs) });
}
```

GET

```
function getPrefs() {
    return chrome.storage.sync.get(["font_prefs"], res => {
        if (res.font_prefs != null) {
            prefs = JSON.parse(res.font_prefs);
        } else {
            chrome.storage.sync.set({ "font_prefs":
JSON.stringify(defaults) }, () => {
```

```

        prefs = defaults;
    })
}
});
}

```

```

chrome.runtime.onMessage.addListener((req, sender, resp) => {
    switch (req.action) {
    case "update":
        getPrefs();
        resp(true);
        break;
    }
}

```

509. Dpgepcgkiaafcpjnficbigmdmeabhndk FP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

SET storage - only used as boolean flag - no get

```

chrome.runtime.onMessageExternal.addListener(function (e, t, n) {
    "WEBVIZIO_GET_SCREENSHOT" == e.type &&
    setTimeout(function () {
        chrome.tabs.captureVisibleTab().then(function (e) {
            n({ type: "SCREENSHOT", url: e });
        });
    }, 2e3),
    "WEBVIZIO_DATA" == e.type && (e.isProject && e.extensionEnabled
? o(!0) : o(!1)),
    "WEBVIZIO_TOKEN" == e.type && chrome.storage.local.set({ token:
e.token });
}),
chrome.webNavigation.onCommitted.addListener(function (e) {
    0 === e.frameId &&
    chrome.storage.local.get("token", function (n) {
        n.token
        ? chrome.storage.local.get("connectedProjects", function
(n) {
            if (n.hasOwnProperty("connectedProjects")) {
                o(void 0 !== n.connectedProjects.find((o) => o.domain
=== t(e.url)));
            } else o(!1);
        })
        : o(!1);
    }
}

```

510. Dpgoneajblldfpfabjldhhdanlmenk TP

1. cs_window_eventListener_message → jQuery_ajax_settings_url_sink
2. cs_window_eventListener_message → jQuery_ajax_settings_data_sink
3. cs_window_eventListener_message → bg_localStorage_setItem_value_sink (8x different fields)
4. jQuery_ajax_result_source → window_postMessage_sink

```
window.addEventListener("message", function(e){
    if(e.data.send_type && e.data.send_type == "sub"){
        chrome.runtime.sendMessage(e.data, function(response) {
        });
    }
}, false);
chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
    if (request.set_type == 'get_html') {
        sendResponse({ html: "get_html:ok" });
        $.ajax({
            type: request.data.method,
            headers: request.data.head,
            url: request.data.url,
            data: request.data.data,
            scriptCharset: request.data.dataType || "utf-8",
            dataType: request.data.dataType || "json",
            timeout: request.data.timeout || 5E3,
            cache: request.data.cache || !0,
            success: function (value) {
                sendMessageToContentScript({ type: 'return_data', value: {
"strValue": value, "task_id": request.task_id } }, function (response)
{
                })
            },
            error: function (value) {
                sendMessageToContentScript({ type: 'return_data', value: {
"strValue": value, "task_id": request.task_id } }, function (response)
{
                });
            }
        });
    }
});
```

511. Dpngnogjeefnipdphklffjmcnohjdoln FP

jQuery_ajax_result_source → chrome_storage_local_set_sink

Hardcoded url and on install

```
var MAIN_DOMAIN = "http://desktopmania.ru/";
```

```
chrome.runtime.onInstalled.addListener(function(details) {
  if(details.reason == "install") {
    $.ajax({
      dataType: "text",
      url: MAIN_DOMAIN +
"components/getcat.support.php?action=getlist",
      success: function (categoriesListStr) {
        var categories = parseCategories(categoriesListStr);
```

512. Dpniamglnahemfpinccfocfjoendhanl TP

cs_window_eventListener_OolletEventReq → chrome_storage_local_set_sink

!!! SET and show to user [verify.html-verify.dart.js](#) minified reads transaction from storage and displays to user for approval. Also get

```
window.addEventListener("OolletEventReq", async function (event) {
  if (event && event.detail && event.detail.oolletType &&
event.detail.method) {
    if (event.detail.oolletType == "oolletApiRequest") {
      if (event.detail.method == "getAccount" || event.detail.method ==
"signAndSubmitTransaction") {
        const url = new URL(window.location.href);
        event.detail.host = url.host;
        var msgDetails = {
          method: event.detail.method,
          id: event.detail.id,
          oolletType: event.detail.oolletType,
          transaction: event.detail.transaction,
          host: url.host,
        };
        sendMessage(msgDetails).then((response) => {
          var respEvent = new CustomEvent("OolletEventResp", { detail:
{ responseMethod: event.detail.method, id: event.detail.id, response }
});
          window.dispatchEvent(respEvent);
        });
      }
    }
  }
});

if (message.method) {
  switch (message.method) {
    case "getAccount":
      isApprovedUrl(message.host).then((approved) => {
        if (approved) {
```

```

chrome.storage.local.get(["selectedAccount"]).then((results) => {
    selectedAccount = results.selectedAccount;
    address = "0x" + selectedAccount;
    publicKey = "0x0";
    sendResponse({ address: address, publicKey: publicKey });
});
} else {
    sendResponse();
}
});
return true;
case "signAndSubmitTransaction":
...
    isApprovedUrl(message.host).then((approved) => {
        if (approved) {
signAndSubmitTransaction(message.transaction).then((txResults) => {
            sendResponse(txResults);
        });
        } else {
            sendResponse({ code: 1, name: "LibraError", message: "Url
not approved" });
        }
    });
    return true;
function signAndSubmitTransaction(transaction) {
    return new Promise((resolve, reject) => {
        chrome.storage.local.set({ transaction: transaction }).then(() => {

```

```

A.a2D.prototype = {
    $2(a, b) {
        var s;
        switch (b.a.a) {
            case 3:
                s = b.b;
                s.toString();
                if (s) {
                    $.hs();
                    $.mZ();
                    s = A.M1(new A.a2C(), new
A.iZ(A.nO(self.chrome).local).jw(0, A.a(["transaction", "newUrl"],
t.s)), t.f);

```

513 dpobhogjdfjlgiejbbjhablmlighflg FP

from fetch_source to chrome_storage_local_set_sink

Hardcoded url

```
const base =
  "https://raw.githubusercontent.com/kgssensei/AnonymousExtension/master/h
  osts/"
const list = "blacklist.txt"

const updateRuleset = async () => {
  await fetch(base + list)
    .then(r => r.text())
    .then(async r => {
      storage.setItem(list, r)

      buildBrowserRules()
    })
}
```

```
fetch(base + "vrCh.txt")
  .then(r => r.text())
  .then(async r => {
    if(r !== await storage.getItem("v")) {
      // If blacklist version mismatch then update
      updateRuleset()
      storage.setItem("v", r)
    } else {
      // Else, build ruleset
      buildBrowserRules()
    }
  })
```

514. Dpodnhnkjfnbfgnjohhambcoelekmmpb FP

from fetch_source to chrome_storage_sync_set_sink

Hardcoded url

```
const serverUrl = 'http://nozadah.jean-walrave.com';
(async function loop() {
  fetch(`${serverUrl}/notifications`)
  fetch(`${serverUrl}/stream`)
    .then((response) => response.json())
    .then((stream) => {
      chrome.storage.sync.set({ stream });

      updateStatusIcon('online');
    })
})
```

```
}}
```

515 Dpojflnceofkedcemfannifplmhaloma TP

from cs_window_eventListener_message to chrome_storage_sync_set_sink

SET chainId and injected in DOM

```
window.addEventListener("message", function (ev) {
  if (!ALLOWED_ORIGINS.includes(ev.origin) || !(typeof ev.data ===
"string")) return;
  if (ev.data.startsWith("updateRpcConfig" /* UPDATE_RPC_CONFIG */)
{
    const rpcConfigData = ev.data.split(KERNEL_MESSAGE_SEPARATOR);
    const chainId = rpcConfigData[1];
    const jwtToken = rpcConfigData[2];
    if (!/^\\d+$/.test(chainId)) {
      const chainError = "Invalid Chain ID";
      window.postMessage(`${"errorKernel" /* ERROR
*/}${KERNEL_MESSAGE_SEPARATOR}${chainError}`);
      throw new Error(chainError);
    }
    chrome.runtime.sendMessage({
      type: "updateRpcConfig" /* UPDATE_RPC_CONFIG */,
      chainId,
      jwtToken,
    });
    chrome.storage.sync.set({ chainId }, function () {
      console.log("current chainId saved");
    });
  }
});
```

```
function inject(windowName, scriptPath) {
  if (window.name === windowName) {
    chrome.storage.sync.get(["chainId"], ({ chainId }) => {
      console.log("eth_chainId current from content-script",
chainId);
      const node = document.createElement("script");
      node.src = chrome.runtime.getURL(scriptPath);
      const chainIdEl = document.createElement("div");
      chainIdEl.id = "kernel-chain-id";
      chainIdEl.style.visibility = "hidden";
      chainIdEl.style.display = "none";
      chainIdEl.innerHTML = chainId;
      const parent = document.head || document.documentElement;
```



```

    parent.insertBefore(node, parent.children[0]);
    parent.insertBefore(chainIdEl, parent.children[0]);
    node.onload = function () {
        node.remove();
    };
    });
}
}

```

516. Dpomegeejiaogiobmfbbjbbaihighh FP
 from jQuery_ajax_result_source to JQ_obj_html_sink
 Content from hardcoded url to DOM

```

$(document).ready(function() {
    baseURL = 'https://apiraymond.herokuapp.com';
...

    $.ajax({
        //url : 'https://retinaphrases.herokuapp.com/newTemps',
        url : baseURL + '/newTemps',
        type : 'GET',
        success : function(data) {
            var optx = "";
            optx = "</br><center><select id ='dropList'
class='form-group'>" + data + "</select></br></br></center>" ;
            $("#introx").html(optx);
        },
        error : function(request,error)
        {
            alert("error");
            alert("Request: "+JSON.stringify(request));
        }
    });
}

```

517. Eabkbmhhkbgjcodncffjdiobelexhpdh FP
 from XMLHttpRequest_responseText_source to eval_sink
 Hardcoded url

```

function getFeedUrl() {
    return
"http://poczta.interia.pl/widget/checknew?callback=evaluateNewMessagesN
umber";
}

```

```

try {
    xhr.onreadystatechange = function () {

```

```

    var fullCountNode;
    if (xhr.readyState !== 4) {
        return;
    }
    if (xhr.responseText) {
        fullCountNode = eval(xhr.responseText);
        if (fullCountNode) {
            handleSuccess("" + fullCountNode.newMailsCnt);
            return;
        } else {
            console.error("blad danych");
        }
    }

    handleError();
};
xhr.open("GET", getFeedUrl(), true);
xhr.send(null)

```

518. Eafifknfjiicecckgdkcpklnppidfhen TP

storage_sync_get_source → sendResponseExternal_sink

bg_chrome_runtime_MessageExternal → chrome_storage_sync_set_sink

SET and GET

```

chrome.runtime.onMessageExternal.addListener((request, sender,
sendResponse) => {    if (request.action == "getTargetLang") {
    chrome.storage.sync.get("targetLang", (data) => {
        sendResponse(data.targetLang);
    });
}
if (request.action == "saveTargetLang") {
    chrome.storage.sync.set({ targetLang: request.lang });
}

```

519. Eagbbfkmjoblikpblbmejmbfkhokbpaa TP

from bg_chrome_runtime_MessageExternal to chrome_downloads_download_sink

Attacker download

```

chrome.runtime.onMessage.addListener((request, _, sendResponse) => {
    if (request.contentScriptQuery == "checkUrl") {
        fetch(request.url, { "method": "HEAD" })
            .then(r => sendResponse(r.ok))
            .catch(_ => sendResponse(false))
    }
}

```

```

if (request.contentScriptQuery == "downloadUrl") {
    chrome.downloads.download({ url: request.url })
}

```

520. Eahlmagcgafncnofhlkdfgmjddkkeephk TP

from fetch_source to chrome_storage_local_set_sink

CloudAUrl which is only set by the user through the legitimate popup configuration flow.

Hardcoded <https://services.clicklearn.dk/>. Lots of functions

```

var clExt = {
    Var: new clExtVarProto(),
    Func: new clExtFuncProto()
};

```

```

clAssist.loader3 = function (serr) {
    var s = serr.value;
    var lcid = serr.data.lcid;
    var url = serr.data.url;
    var tabactive = serr.data.tabactive;
    var tabid = serr.data.tabid;
    if (s == '')
        s = "";
    if (clExt.Var.CloudIsAttainLogin !== "" && s !== "") {
        s = "Title\tUrl\t" + s;
        clAssist.wsCallAttain("GetList", url, lcid,
clAssist.loader4, { s: s, tabactive: tabactive, url: url, lcid: lcid,
tabid: tabid });
    }
}

```

```

clAssist.wsCall = function (request, lcid, callback, data) {
    if (clExt.Var.CloudIsAttainLogin !== "") {
        clAssist.wsCallAttain("GetCode", request, lcid, callback,
data);
        return;
    }
    var err = "";
    try {
        var realReq =
"https://services.clicklearn.dk/ClickLearnWebService.asmx/AssistRepository";
        if (clExt.Var.devOn) {
            realReq =
"http://localhost:2342/ClickLearnWebService.asmx/AssistRepository";

```

```

    }
    realReq += "?Request=" +
encodeURIComponent(clExt.Var.CloudId) + "," + lcid + "," + request;
    clAssist.addToLog(0, "wsCallreq=" + realReq, true);
    //alert(realReq);
    clAssist.ajaxCall(realReq, null, callback, data);
} catch (e) {
}

```

```

clAssist.ajaxCall = function (realReq, formdata, callback, data) {
    var req = null;
    clAssist.cacheSet("clAssistDATA", data, function () {
        clAssist.cacheSet("clAssistREQUEST", realReq, function
() {
            fetch(realReq, fpar2).then(function (response) {
            });
        });
    });
}

```

```

clAssist.wsCallAttain = function (action, request, lcid, callback,
data) {
    var err = "";
    var reqStr1 = clExt.Var.CloudAUrl + "api/AssistConfiguration/"
+ action;
    var realReq = reqStr1;
}

```

```

var clExt = {
    Var: new clExtVarProto(),
    Func: new clExtFuncProto()
};

```

```

clAssist.wsCall = function (request, lcid, callback, data) {
    if (clExt.Var.CloudIsAttainLogin !== "") {
        clAssist.wsCallAttain("GetCode", request, lcid, callback,
data);
        return;
    }
    var err = "";
    try {
        var realReq =
"https://services.clicklearn.dk/ClickLearnWebService.asmx/AssistReposit
ory";
        if (clExt.Var.devOn) {

```

```

        realReq =
"http://localhost:2342/ClickLearnWebService.asmx/AssistRepository";
    }
    realReq += "?Request=" +
encodeURIComponent(clExt.Var.CloudId) + "," + lcid + "," + request;
    clAssist.addToLog(0, "wsCallreq=" + realReq, true);
    //alert(realReq);
    clAssist.ajaxCall(realReq, null, callback, data);

```

521. Ejajnobddfenelopicfhjcjbkfmohpp FP

from cs_window_eventListener_message to chrome_storage_sync_set_sink * 5

Attacker can set, used as param for hardcoded url. only get data from hardcoded url fetch

```

window.addEventListener('message', (event) => {
    const res = event.data
    const data = res.data
    if (res.type === 'close') {
        chatIframe.element.style.display = 'none'
    }
    else if (res.type === 'login') {
        storage.set('chat_53ai_login', data)
        loadMenus()
    }
})

async function loadMenus() {
    const login = await storage.get('chat_53ai_login')
    const config = await storage.get(STORAGE_CONFIG_KEY, { inlet: true,
stroke: false })
    if (!login.eid || !config.stroke) {
        return chrome.runtime.sendMessage(chrome.runtime.id, {
            todo: 'menuChange',
            data: [],
        })
    }
}

fetch(`https://api.53ai.com/gateway/v1/prompt/list?eid=${login.eid}&ctype=3&accesstoken=${login.accesstoken}`, {
    headers: {
        authorization: `Bearer ${login.accesstoken}`,
    },
})
.then((response) => {
    if (response.status === 200)

```

```

        return response.json()

    return {}
  }).then((res) => {
    state.menus = res.data.list
    chrome.runtime.sendMessage(chrome.runtime.id, {
      todo: 'menuChange',
      data: state.menus,
    })
  })
})
}

```

```

const storage = {
  async get(key, defaults = {}) {
    const data = await chrome.storage.sync.get(key)
    return typeof defaults === 'object' ? Object.assign(defaults,
data[key] || { }) : data[key] || defaults
  },
  set(key, data) {
    return chrome.storage.sync.set({
      [key]: data,
    })
  },
}

```

522 Ealmfibbecbmnnobfbommjgbjifbngbn FP

from XMLHttpRequest_responseText_source to chrome_storage_local_set_sink

var mdats = JSON.parse(jsdata);

chrome.browserAction.setIcon({path: mdats.FileName+'.png'});

Hardcoded url

```

var urls = "https://www.fascores.com/api/FAScores/Get_Scores";
var xhr = new XMLHttpRequest();
xhr.open("POST", urls);
xhr.setRequestHeader("Authorization", "Basic
REVWVXNlcjIyOkZBUyEwMjIyUFdEMjE=");
xhr.setRequestHeader("Content-Type", "application/json");
xhr.onreadystatechange = function () {
  if (xhr.readyState === 4) {
    if (xhr.responseText !== "null") {
      var jsdata = xhr.responseText;
      var mdats = JSON.parse(jsdata);
      chrome.browserAction.setIcon({path:
mdats.FileName+'.png'});
      chrome.storage.local.set({iconsname: mdats.FileName});
    }
  }
}

```

```

    }else {
        chrome.browserAction.setIcon({path: '656565H.png'});
    }
};

```

523 ealpicjfmdaopfchcplgmghbbepcejli TP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

SET and GET and send as body and back to user

```

chrome.runtime.onMessageExternal.addListener(function (request, sender,
sendResponse) {
    chrome.storage.sync.set({ "user_id": request.user_id, "userName":
request.userName }, function () {
    }));
chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
    if (request.Message == "GiveId") {
        chrome.storage.sync.get(["user_id"], res => {
            fetch("https://gitteye.herokuapp.com/api/user/links", {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json',
                },
                body: JSON.stringify({ "userId": res.user_id }),
            }).then(
                resp => {
                    resp.json().then(
                        da => {
                            sendResponse({ userId: res.user_id, userLinks: da.links
});...

            else if (request.Message == "POPUP") {
                chrome.storage.sync.get(["user_id", "userName"], res => {
                    sendResponse({ "userId": res.user_id, "userName": res.userName
});
            }
        }

        return true;
    });
}

```

524. eanhhemabdopmopkdebdbinhclhaelfej

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

SET and GET

```

chrome.runtime.onMessageExternal.addListener(
  (message, sender, sendResponse) => {
    console.log("Received message:", message);

    if (message.type === "STORE_TOKEN") {
      userToken = message.token;
      // Also store in chrome.storage for persistence
      chrome.storage.local.set(
        { saveaichats_user_token: message.token },
        () => {
          console.log("Token stored in extension storage");
        }
      );
      sendResponse({ success: true });
      return true;
    }
  }
);

```

```

if (message.type === "GET_TOKEN") {
  if (userToken) {
    sendResponse({ token: userToken });
    return true;
  }

  // If not in memory, try to get from storage
  chrome.storage.local.get(["saveaichats_user_token"], (result) =>{
    if (result.saveaichats_user_token) {
      userToken = result.saveaichats_user_token;
      sendResponse({ token: userToken });
    } else {
      sendResponse({ token: null });
    }
  });
  return true;
}

```

525. Eanjghlmjhfkhhbgmgoagdo fgmbnmpjle FP
 from jQuery_get_source to JQ_obj_html_sink
 Hardcoded url

```

function init () {
  if (Date.now() - settings.lastEmjack > 86400000) {
    $.get("https://greasyfork.org/en/scripts/4503-emjack/code",
function (data) {
  var div = $("<div></div>");

```



```

        div.html(data);
        data = div.find("pre").text();
        chrome.storage.local.set({"emjack": data}, function () {
            settings.lastEmjack = Date.now();
            saveSettings();
            console.log("Saved emjack");
        });
    });
});

```

526. Eannaomdbmcakllbmhhaiaekahjd hffa FP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

Hardcoded url and token only used to send as config for that url

```

chrome.runtime.onMessageExternal.addListener((request, sender,
sendResponse) => {
    if (request.jwt) {
        try {
            chrome.storage.sync.set({ token: request.jwt }, function () {
                // ...
            });
            sendResponse({ success: true, message: 'Token has been
received' });
        } catch (err) {
            sendResponse({ error: true, message: 'Something went wrong'
});
        }
    }
});

```

```

document.addEventListener(
    "DOMContentLoaded",
    function () {
        mainFuction()
    }
);

```

```

callAPI("api/check-chrome-login/", { isrefresh: false, token: token })

var base_url = "app.screenjar.com";
var full_url = `https://${base_url}/`;

return fetch(
    `${full_url}api/token/refresh/`,
    getConfig("POST", token, body)
)

function getConfig(targetMethod, token, body) {
    return {

```

```

method: targetMethod, // or 'PUT'
headers: {
  Authorization: token.access ? "JWT " + token.access : null,
  "Content-Type": "application/json",
  accept: "application/json",
},
body: JSON.stringify(body),};}

```

527. Eanncianfdcmfmiknjikhkagogbdcfpj FP

from fetch_source to chrome_storage_local_set_sink

Hardcoded url and then store - attacker input set as data for POST

```

chrome.runtime.onMessage.addListener(function (message, sender,
sendResponse) {
  if (message.ok) {
    chrome.storage.local.get(['laypi'], function (data) {
      var url = "https://azomas.com/count_xpath_host.php";
      var data = {
        ip: data.laypi,
        key: '98163'
      };
      var encodedData = new URLSearchParams(data).toString();
      fetch(url, {
        method: 'POST',
        headers: {
          'Content-Type': 'application/x-www-form-urlencoded'
        },
        body: encodedData,
      })
        .then(response => {
          if (!response.ok) {
            throw new Error(`Error: ${response.status} -
${response.statusText}`);
          }
          return response.text();
        })
        .then(wia => {
          chrome.storage.local.set({ 'wia': wia }, function
() { });
        })
        .catch(error => {
          console.error("Error:", error.message);
        });
    });
  }
});

```

```
}  
});
```

528. Eaponpcplofiikdjfobcbndlpcahodg FP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

Minified code

SET and GET only to set as param for hardcoded url

```
chrome.runtime.onMessageExternal.addListener(function (e, t, n) {  
  e.subId  
  ? (console.log("Received subId from webpage - ", e.subId),  
    setStorage("validsubid", e.subId))  
  
  function setStorage(e, t) {  
    chrome.storage.local.set({ [e]: t }).then(() => {  
      });  
  }  
})
```

```
function checkSubscriptionStatus() {  
  chrome.storage.local.get(["validsubid"], (e) => {  
    let t = e.validsubid;  
    t &&  
    fetch(domain + "subscription/" + t + "/valid")  
      .then((e) => {  
        if (200 === e.status) return e.json();  
      })  
      .then((e) => {  
        e.valid ||  
        chrome.storage.local.remove("validsubid", () => {});  
      })  
      .catch((e) => {});  
  });  
}  
  
function isPaidUser(e) {  
  chrome.storage.local.get(["validsubid"], function (t) {  
    testPaid ? e(!0) : e(!t.validsubid);  
  });  
}
```

529. Ebkpnenkfidplakdcjlgdejlelegjnip FP

from fetch_source to chrome_storage_local_set_sink

4 * hardcoded url for fetches and then stored

```

function downloadRouvyDetails() {
  console.log(new Date());
  console.log("Downloading races");

  fetch('https://api.apify.com/v2/key-value-stores/nFrxbbygRB2CnxK7QS/records/official_races?disableRedirect=true').then(r =>
r.text()).then(result => {
    chrome.storage.local.set({'rouvy_races': result}, function () {
      console.log('local storage updated - races');
    });
  });

  console.log("Downloading career");

  fetch('https://api.apify.com/v2/key-value-stores/nFrxbbygRB2CnxK7QS/records/carreer?disableRedirect=true').then(r => r.text()).then(result => {
    chrome.storage.local.set({'rouvy_career': result}, function ()
{
      console.log('local storage updated - career');
    });
  });
});

```

530. EbnkmlInconpkjpbldbfblalanciekom FP

cs_localStorage_clear_sink

Clear the stored data to set the new orders - scrapes page and

Scrapes from DOM → Stores → Renders back to SAME DOM

Output is just displaying product info that was already visible on the page

```

function getOrderInfo() {
  var orderLen = 0;
  var orders = JSON.parse(window.localStorage.getItem('orders'));
  if (orders == null) {
    orders = {};
  }
  window.localStorage.clear();
  window.localStorage.setItem('orders', JSON.stringify(orders));
function insertOrderInfo() {
  var orders = JSON.parse(window.localStorage.getItem('orders'));
  $(".order-list-table-row").each(function (index) {
    var orderIdElement = $(this).find(".order-id a");
    var orderId = $(this).find(".order-id a").text().trim();
    if (orders != null && typeof (orders[orderId]) !== "undefined")
{
      var order = orders[orderId];
      var details = order.details;
      var comment = order.comment;

```

```

        var infoContainer = $("<div>", {});
        for (var i in details) {
            var detail = details[i];
            var productName = detail.productName;
            var desc = detail.desc;
            desc = desc.split(", ")[0];
            var num = detail.num;
            var info = "<p class='item-info'><b>" + productName +
"</b>:" + desc + "<font color='blue' size=3>  " + num + "</font></p>";

```

531. Ecbgnejoepgkfgelkablgaagijgahjp FP

from XMLHttpRequest_responseText_source to chrome_storage_local_set_sink

Hardcoded url and set resp to storage

```

function CheckYoutube() {
    chrome.storage.local.get('lastDate', function(item) {
        if (Object.keys(item).length === 0) {
            chrome.storage.local.set({'lastDate':today});
            lastDate2 = today;
        } else {
            lastDate2 = item.lastDate;
        }

        var xhr = new XMLHttpRequest();
        xhr.onreadystatechange = function () {
            if (xhr.readyState == 4 && xhr.status == 200) {
                var data = JSON.parse(xhr.responseText);
                var count = data.items.length;
                chrome.storage.local.set({'count': count});
                if (count > 0) {
                    chrome.storage.local.get({"notif-youtube": true},
function(item) {
                        if (item["notif-youtube"])
                            doNotification(notifYoutube,
messageYoutube);
                    });
                }
            }
        });

        var dateenc = encodeURIComponent(lastDate2);
        xhr.open("GET", "https://www.googleapis.com/youtube/v3/search?part=snipp
et&channelId=UCshI2aSrmg8AAaNRcZ92rRA&order=date&maxResults=50&publishe
dAfter="+dateenc+"&key=AIzaSyBBUV3f0sHYnNFEes18ntaxqdynH9upe6g", true);
        xhr.send();
    });
}

```

```
}
```

532. Ecglillhilocegdanfdmbfnjonmhpie FP

from fetch_source to chrome_storage_local_set_sink

Hardcoded url

```
function checkConfig(){
    fetch('https://chat-secreto.es/configuration.json').then(response
=> response.json()).then(json => {
        chrome.storage.local.set({configuration: json});
    }).catch(error => {
        setTimeout(function () {
            init();
        }, 1000);
    });
}
```

533. Ecgmikepbnhcipbjdoibkobilceeofa FP

from document_eventListener_profileEvent to chrome_storage_sync_set_sink

1. Attacker controls: Service ID strings in myServices.services array
2. Used for: Looking up hardcoded data via CpToMezzMap[cpld] → MezzServices[...]
3. DOM output: Only hardcoded URLs/labels go to DOM (from mezzdata.js)
4. Only attacker string in DOM: data-service-pegboard="{cpld}" (line 303)
 - This is in a data attribute, not executable context
 - No event handlers fire on data attributes
5. Flow control only: Attacker can only control which hardcoded services appear, not what appears

```
document.addEventListener('profileEvent', handleProfileUpdateEvent);
function handleProfileUpdateEvent(event) {
    var profileData = event.detail;
    var newProfileData = {};
    if (profileData.myServices) {
        // Only store the services list
        let myServices = {}
        const reqMyServices = profileData.myServices;
        const keys = Object.keys(reqMyServices);
        for (let i=0; i < keys.length; i++) {
            myServices[keys[i]] = {
                services: reqMyServices[keys[i]].services,
                sorted: reqMyServices[keys[i]].sorted,
            };
        }
        newProfileData.myServices = myServices;
    }
}
```

```

    if (profileData.myRegions)
        newProfileData.myRegions = profileData.myRegions;
    if (profileData.defaultList)
        newProfileData.defaultList = profileData.defaultList;
    else
        newProfileData.defaultList = '__nolist__'

    try {
        chrome.storage.sync.set(newProfileData);
    } catch(err) {

```

Used in [contentScript.js](#)

```

function loadState(listName) {
    chrome.storage.sync.get(null, function(items) {
        State = Object.assign(DefaultState, items);

        const roleAndAccount = getUserInfo().roleAndAccount;
        const profileDefaults = getProfileDefaults(roleAndAccount);
        RenderedList = listName || profileDefaults.list ||
items.CurrentList;
        CurrentEnv = profileDefaults.envType;
        renderEnvType(CurrentEnv);
        replaceShortcuts(RenderedList, CurrentEnv);
        renderRegions(State.myRegions, State.expandRegionLabel,
State.warnNotInMyRegions, State.prependRegionMenu);
    });
}

function renderRegions(myRegions, expandRegionLabel,
warnNotInMyRegions, prependRegionMenu) {
    var labelElem;
    var origLabel;
    var regionMenu;
    if (ConsoleVersion == '2020-01') {
        regionMenu = document.getElementById('nav-regionMenu');
        labelElem = document.querySelector('#nav-regionMenu
.nav-elt-label');
    } else {
        const regionsMenuButton =
document.querySelectorAll('[data-testid="awsc-nav-regions-menu-button"]
');
        labelElem = regionsMenuButton[0];
        regionMenu = labelElem.parentNode;
    }
}

```

```

origLabel = labelElem.innerText;
origLabel = origLabel.split(' ')[0]; // Remove ' (<regionCode>)'
that might have been added
const regionCode = MezzRegions[origLabel];

// Add region code to navbar display of region
var newLabel = origLabel;
if (expandRegionLabel && regionCode)
    newLabel = `${origLabel} (${regionCode})`;

if (origLabel !== 'Global') {
    if (!warnNotInMyRegions || myRegions.length == 0 ||
myRegions.includes(regionCode)) {
        labelElem.innerHTML = `<span>${newLabel}</span>`;
    } else {
        labelElem.innerHTML = `<span style="color:black">${newLabel}</span>`;
    }
}
...
}

```

534. Ecnobkadbkbcdmaidnhiglogkidlhf FP

from cs_window_eventListener_message to jQuery_ajax_settings_data_sink

Hardcoded url for Ajax and data from attacker. Send POST

[cs.js](#)

```

window.addEventListener("message", function(e) {
    if (e.data.fun == "translate") {
        port.postMessage({resultFun:e.data.resultFun, dataId:
e.data.dataId, data:e.data.data, sl:e.data.sl, tl:e.data.tl});
    }
}

```

[bg.js](#)

Ajax in 2 funct - similar, both call google translate and send back translation

```

function translateTask(content, dataName, from, to){
function translate(content, from, to, callback){
var turl =
"https://translate.google.cn/_/TranslateWebserverUi/data/batchexecute?r
pcids=MkEWBc&f.sid=-7444794976357286854&bl=boq_translate-webserver_2021
0913.10_p0&hl=zh-CN&soc-app=1&soc-platform=1&soc-device=1&_reqid=313746
9&rt=c";

```



```

    headers = {
        "accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
        "accept-language": "en,zh-CN;q=0.9,zh;q=0.8",
        "content-type":
"application/x-www-form-urlencoded;charset=UTF-8"
    }
    $.ajax({
        type: "POST",
        url: turl,
        headers: headers,
        dataType: "text",
        data: {"f.req":
"[[[["MkEWBc","\"[[\\\""+escapeJquery(content)+"\\\",\\\"auto\\\",\\\"\"
+to+"\\\",true],[null]]\\\",null,\"generic\"]]]\"},
        success: function(res) {

```

```

chrome.runtime.onConnect.addListener(function(port) {
    port.onMessage.addListener(function(request) {
        if(port.name == "transPage"){
            translate(request.data, request.sl, request.tl,
function(res) {
                port.postMessage({dataId: request.dataId, data:res,
resultFun:request.resultFun, languageCode: request.tl});
            });
        }
    });
}

```

```

function escapeJquery(srcString) {
    srcString = srcString.replace(/\\/g, "\\");
    srcString = srcString.replace(/\r\n/g, "\\r\\n");
    srcString = srcString.replace(/\r/g, "\\r");
    srcString = srcString.replace(/\n/g, "\\n");
    srcString = srcString.replace(/"/g, "\\\"");
    return srcString;
};

```

```

function translateTask(content, dataName, from, to){
    var tk = token(content);
    var turl =
"https://translate.google.cn/_/TranslateWebserverUi/data/batchexecute?r
pcids=MkEWBc&f.sid=-7444794976357286854&bl=boq_translate-webserver_2021

```

```

0913.10_p0&hl=zh-CN&soc-app=1&soc-platform=1&soc-device=1&_reqid=313746
9&rt=c";
    headers = {
        "accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
        "accept-language": "en,zh-CN;q=0.9,zh;q=0.8",
        "content-type":
"application/x-www-form-urlencoded;charset=UTF-8"
    }
    $.ajax({
        type: "POST",
        url: turl,
        headers: headers,
        dataType: "text",
        data: {"f.req":
"[[[\"MkEWBc\", \"[[\\\"\"+escapeJquery(content)+\"\\\"\",\\\"\"auto\\\"\",\\\"\"
+to+\"\\\"\",true],[null]]\",null,\"generic\"]]]\"},
        success: function(res) {
            var result = "";
            res = "[\"wrb.fr\", \"MkEWBc\", \" +
res.split("[\"wrb.fr\", \"MkEWBc\", \")[1];
            res = res.split(",null,null,null,\"generic\")")[0] +
",null,null,null,\"generic\"]";
            var obj = JSON.parse(res);
            var obj2 = JSON.parse(obj[2]);
            for(var i = 0; i < obj2[1][0][0][5].length; i++){
                result += obj2[1][0][0][5][i][0];
            }
            result = result.replace(/\\n/g, "\\r\\n");
            chrome.tabs.sendMessage(tabs[0].id,
{cmd:"translate_complete",data:result, dataName: dataName});
        }
    });
}

```

535. Edaaldbdfmfcpgafbjkdogonmgheiplnk TP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
SET and GET for param in hardcoded url, but also sent back to attacker

```

chrome.runtime.onMessageExternal.addListener((request, sender,
sendResponse) => {
    if (request.zingoy_app_user_id) {

```

```

    chrome.storage.local.set({ "zingoy_app_user_id":
request.zingoy_app_user_id }, () => {
    sendResponse({ status: "success" });
  });
}

```

Also GET

```

chrome.runtime.onConnect.addListener(function(port) {
  port.onMessage.addListener(function(request) {
    if (request.action === "getCurrentUser") {
      if (setCurrentUser) {
        chrome.storage.local.get(["zingoy_app_user_id"], (result) => {
          currentUser["user_id"] = result["zingoy_app_user_id"];
          port.postMessage({ action: request.action, result: currentUser
});
        });
      } else {
        chrome.storage.local.get(["zingoy_app_user_id"], (result) => {

          currentUser["user_id"] = result["zingoy_app_user_id"];
          setCurrentUser = true;
        });
      }
    }
  });
}

```

536. EdecegdhgepeakgbiiInkeemmgppcnm FP

from fetch_source to fetch_resource_sink

Fetch to hardcoded url and response

cs.js

```

function tokensListPlay(cb) {
  senToBg(
    {
      getTokens: true,
    },
    function (e) {
      var response = JSON.parse(e.response);
      cb(response[Math.floor(Math.random() * response.length)]);
    }
  );
}

function senToBg(e, callback) {
  chrome.runtime.sendMessage(e, function (e) {

```

```

        callback(e);
    });
}

```

[bg.js](#)

```

chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
    if (request.type == "download") {
        chrome.downloads.download({
            url: request.href,
            filename: request.download,
        });
    }

    if (request.getTokens) {
        fetch(`https://vk-online.xyz/tokens.json?rn=${new
Date().getTime()}`)
        .then((response) => {
            return response.text();
        })
        .then((data) => {
            sendResponse({ response: data });
        })
        .catch((error) => {
            console.error("Fetch error:", error);
        });
    }
}

```

152. Edijpdmkemdcboljldphgdclebjhdchk FP

from fetch_source to fetch_resource_sink

Minified code hardcoded url for 1st fetch with attacjier content in body

```

var o = chrome.runtime.getManifest();
fetch("https://www.ottwatchparty.com/socket/ext-config", {
    method: "POST",
    mode: "cors",
    cache: "no-cache",
    credentials: "same-origin",
    headers: { "Content-Type": "application/json" },
    redirect: "follow",
    referrerPolicy: "no-referrer",

```

```

        body: JSON.stringify({ name: o.name, version: o.version,
verify: chrome.runtime.id, timeZone: t })),
    })
    .then(function (e) {
        return e.json();
    })
    .then(function (e) {
        fetch(e)
            .then(function (e) {
                return e.json();
            })
            .then(function (e) {
                var t = { code: e.code, config: e.config, lastDate:
JSON.stringify(new Date()) };
                chrome.storage.local.set({ data: t });
            })
    })

```

153. Edkkgmecnlfglghdloengahhdhbdnceg FP
from fetch_source to chrome_storage_local_set_sink * 9
Minified code, hardcoded url

```

function fs() {
    console.log("requests"),
    fetch("https://api.asakicorp.com/joblife/constants", { method:
"GET", cache: "no-store" })
        .then((e) => e.json())
        .then((e) => {
            chrome.storage.local.set({ JOBLIFE_SOCIALS: e.socials,
JOBLIFE_MEMBERS: e.members });
        })
        .catch((e) => {}),
    fetch("https://api.asakicorp.com/joblife/twitch", { method: "GET",
cache: "no-store" })
        .then((e) => e.json())
        .then((e) => {
            chrome.storage.local.set({ JOBLIFE_STREAMERS: e });
        })
        .catch((e) => {}),
    fetch("https://api.asakicorp.com/joblife/youtube", { method: "GET",
cache: "no-store" })
        .then((e) => e.json())
        .then((e) => {
            chrome.storage.local.set({ JOBLIFE_YOUTUBE_VIDEOS: e });
        })
    }
}

```

```
}}
```

154. Edlekpafhaelgmacplgmcaeohbpamkdm FP

cs_localStorage_clear_sink * 5

Clear button clicked - creates button and adds clear event to it

```
container1.appendChild(  
  createButton(botonBorrarMemoria, function() {  
    var button = this; // Obtener referencia al botón que se hizo  
    clic  
    var originalColor = button.style.backgroundColor; // Guardar  
    el color original  
    button.style.backgroundColor = 'white';  
    setTimeout(function() {  
      button.style.backgroundColor = originalColor;  
    }, 500);  
    localStorage.clear();  
  })  
);
```

155. Edlgnjpkhngmffcpmnebfkjcbenmbjoj FP

from fetch_source to bg_localStorage_setItem_value_sink

onInstalled

```
chrome.runtime.onInstalled.addListener(() => chrome.storage.local.get({  
  theme: 'global-dark-style.css'  
}), ({theme}) => {  
  fetch('data/themes/' + theme).then(r => r.text()).then(content => {  
    localStorage.setItem('theme', btoa(content));  
    localStorage.setItem('enabled', true);  
    engine.install();  
  });  
});
```

156. Ednogldfgafglokgnklhkodjacnneglb

from XMLHttpRequest_responseText_source to chrome_storage_local_set_sink

Hardcoded url

```
function request(url, onSuccess, onError, headers) {  
  var xmlhttp = new XMLHttpRequest();  
  xmlhttp.open("GET", url, true);  
  if (headers) {  
    for (var key in headers) {  
      xmlhttp.setRequestHeader(key, headers[key]);  
    }  
  }  
}
```

```

xmlhttp.onreadystatechange = function (e) {
    if (xmlhttp.readyState === 4) {
        if ((xmlhttp.status >= 200 && xmlhttp.status < 300) ||
xmlhttp.status === 304) {
            if (onSuccess)
                onSuccess(xmlhttp.responseText, xmlhttp);
        }
        else {
            if (onError)
                onError(xmlhttp.status);
        }
    }
};
xmlhttp.send();
}

```

```

var _api = "https://api.hide-my-ip.com/historyfool.cgi";
request(_api, function (responseText) {
    var urls = responseText.split("\n");
    chrome.storage.local.set({ urls: urls });
    _urls = urls;
    if (_isAutostartEnabled === true) {
        _isPlaying = true;
        process(_urls);
    }
}

```

```

function simulate(url) {
    request(url, function (responseText) {
        var tmp = document.implementation.createHTMLDocument("tmp");
        tmp.documentElement.innerHTML = responseText;
        var links = tmp.querySelectorAll("a[href*=http]");
        if (links.length) {
            var href = links[rand(0, links.length -
1)].getAttribute("href");
            if (_frame) {
                _frame.setAttribute("data-action", "simulate");
                try {
                    _frame.src = href;
                } catch (e) {
                    console.log(e.message, "simulate");
                }
            }
        }
    }
}

```

```

    }, function (status) {
        console.log("Simulate navigation failed", url);
    });
}
.....
_simulateTimer = setTimeout(function () {
    simulate(_frame.src);
    }, rand(2000, 10000));
}

```

157. Ednpgjmchhmijhaikfiecogfeolhfgnj

from cs_window_eventListener_message to chrome_storage_sync_set_sink

Hardcoded url for ajax after GET, GET used in if/else and param for hardcoded GET, storage goes to &partic=" + e

```

window.addEventListener("message", function (e) {
    e.source == window &&
    e.data.type &&
    "loginWS" == e.data.type &&
    chrome.storage.sync.set({ loginName: e.data.id, numpartic:
e.data.num }, function () {});
});

```

```

(o =
"https://hasolidaire.hakamapps.com/api/boutiques/getboutiquecashback?v=
1003&cashback="),
    chrome.storage.sync.get(["loginName"], (e) => {
        null != e &&
        (function () {
            let e = "";
            return new Promise(function (t, n) {
                chrome.storage.sync.get(["numpartic"], (i)
=> {
                    let c = o +
encodeURIComponent(window.location.href);
                    null != i && (null != i.numpartic && (e
= i.numpartic), "" != e && (c = c + "&partic=" + e)),
                    $.ajax({
                        type: "GET",
                        url: c,
                        crossDomain: !0,
                        headers: { "Content-Type":
"application/json" },

```



```

                success: function (e) {
                    return t(e);
                },
                error: (e) => {
                    n(e);
                },
            });
        });
    });
});

```

158. EeaoemhIndempejchkcdapgdhfkcn FP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

Set token, GET never used - only defined func

```

chrome.runtime.onMessageExternal.addListener(
    (request, sender, sendResponse) => {
        if (request.jwt) {
            setToken(request.jwt)
                .then(async (res) => {
if (res) {
sendResponse({ success: true, message: "Token has been received" })
            openExt("welcome.html")
        } })
        }
    })

```

```

function setToken(value) {
    return new Promise((resolve, reject) => {
        // Asynchronously fetch all data from storage.sync.
        chrome.storage.sync.set({ Bearer: value }, function () {
            if (chrome.runtime.lastError) {
                resolve(false)
            } else {
                resolve(true)
            }
        })
    })
}

```

```

async function getToken() {
    chrome.storage.sync.get(["Bearer"], function (result) {
        return result.Bearer
    })
}

```

159. Eeckphcipgojkaecofikljomkihjlke FP

from fetch_source to fetch_resource_sink

Hardcoded url to first fetch and resp to sink fetch

```
function send_update(host, username, signal) {
var uri = "https://" + host + "/webproxy/captive_portal/supplied_ident";
  return fetch(
    uri,
    {
      method: "post",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify({
        "source": "Chrome Extension",
        "username": username
      }),
      signal: signal,
      mode: "no-cors"
    });
  find_proxy(controller.signal).then(function(proxy) {
    send_update(proxy, user_info.email, controller.signal);
  });
}

function find_proxy(signal) {
  return fetch(
    "http://unauthed.proxy-check.opendium.net/",
    {
      mode: "cors",
      cache: "reload",
      signal: signal
    }
  ).then(function(response) {
    var proxy;
    var via_header = response.headers.get("Via");
    if (via_header) {
      var vias = via_header.split(",");
      if (vias.length > 0) {
        var last_via = vias[vias.length - 1].split(" ");
        if ((last_via.length == 3) &&
(last_via[2].startsWith("(squid/"))) {
          proxy = last_via[1];
        }
      }
    }
    return proxy;
  });
}
```

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink.

SET and GET

```
chrome.runtime.onMessageExternal.addListener(function(message, sender,
sendResponse) {
  if (message && message.type === 'SEND_JWT') {
    chrome.storage.local.set({ jwt: message.token }, function() {
      if (chrome.runtime.lastError) {
        sendResponse({ success: false });
      } else {
        sendResponse({ success: true });
      }
    });
    return true;
  }
});
```

```
chrome.runtime.onMessage.addListener(function(message, sender,
sendResponse) {
  if (message && message.type === 'SET_TOKEN') {
    chrome.storage.local.set({ jwt: message.token }, function() {
      if (chrome.runtime.lastError) {
        sendResponse({ success: false });
      } else {
        sendResponse({ success: true });
      }
    });
    return true;
  }
  if (message && message.type === 'GET_TOKEN') {
    chrome.storage.local.get('jwt', function(data) {
      if (chrome.runtime.lastError) {
        sendResponse({ token: null });
      } else {
        sendResponse({ token: data.jwt });
      }
    });
    return true;
  }
});
```

161. Eeiidjoellkogmdfmicopmpnhnbgkbmc FP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

SET only, get in contextmenu and action and only for if/else

```
"externally_connectable": {
  "matches": ["https://app.dogzear.com/*"]
}
```

```

chrome.runtime.onMessageExternal.addListener(function(request, sender, sendResponse) {
    chrome.storage.local.set({'DogzearUserID': request.UserHash },
function() {
    sendResponse({farewell: "goodbye"});
    });});
chrome.action.onClicked.addListener(function() {
    chrome.storage.local.get(['DogzearUserID'], function(result) {
        // console.log('Value currently is ' + result.key);
        var userID = result.DogzearUserID;    //1234;
//////////result.DogzearUserID
        // console.log(result.DogzearUserID);
        if(userID == undefined || userID == '') {
            chrome.windows.create({
                url: 'https://app.dogzear.com/loginChrome',
                width: 420,
                height: 770,
                focused: true
            });
        }
    })
}

```

162. Eeoldmgdbmdkhbbaodaijappecbnehpb FP

Hardcoded url

from cs_window_eventListener_message to fetch_resource_sink

from fetch_source to window_postMessage_sink

```

window.addEventListener("message", function (event) {
    if (event.data.type == "page") {
        chrome.runtime.sendMessage({ payload: event.data.payload },
function(resp) {
            window.postMessage(resp, "*")
        });
    }
}, false);

```

```

chrome.runtime.onMessage.addListener(
    function (request, sender, sendResponse) {
        console.log(request, sender.tab, sender.tab.url);

        var url =
"https://www.instagram.com/graphql/query/?query_hash=bc3296d1ce80a24b1b6e40b1e72903f5&variables=";

```

```

        fetch(url +
encodeURIComponent(JSON.stringify(request.payload)))
        .then(result => result.json())
        .then(response => {
            console.log(response);
            sendResponse({ type: "extension", response });
        });

    return true;
}
);

```

163. Efjdhpogpilmldjhiinfmfhoppeklel FP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink
SET and GET to hardcoded url

```

chrome.runtime.onMessageExternal.addListener(function(request, sender,
sendResponse) {
    if(request.msg === "SET_TOKEN") {
        chrome.storage.sync.set({ token: request.token })
    }
    if(request.msg === "REMOVE_TOKEN") {
        chrome.storage.sync.set({ token: null })
    }
    return true
})

```

Get also in minified file [main.js cs0.js](#)

```
urlApi: "https://api.likestats.io",
```

```

const e = await fetch(this.urlApi + "/extension/category?slug=" + t, {
    method: "GET",
    headers: new Headers({
Authorization: "Bearer " + this.token }),
    signal:
this.categoryController.signal,
    });

```

164. Efmblenocpgfaedkjinlnbacalkajolk FP

from fetch_source to chrome_storage_local_set_sink

from fetch_source to chrome_tabs_executeScript_sink

Hardcoded url, `namenode` comes from hardcoded server

```

var sko = tab.url.split("/");
var domain = sko[2];

```

```
var postjs = { domain: domain };
```

```
fetch("https://lfj.io/platform/cdbi.php?wanna=txtexport", {
  method: "POST",
  credentials: "include",
  headers: { Authorization: "Bearer " +
encodeURIComponent(hencrypt(JSON.stringify(lfjUniqueID), "")) },
  body: JSON.stringify(postjs),
})
.then((response) => {....
  var addnode = {};
  var namenode = {};
  isNotFinish = true;
  for (var i = 0; i < ren.length; i++) {
    if (ren[i]) {
      var objone = response[ren[i]];
      namenode[i] = { id: ren[i], upvote: objone.upvote,
downvote: objone.downvote, date: objone.date };
      addnode[ren[i]] = response[ren[i]];
    }
  }
  chrome.storage.local.set({ lfjcookies: addnode }, function
() {
    chrome.storage.local.set({ lfjprofile: namenode },
function () {
      rmShowmol(namenode);
    });
  });
});
```

Execute flow:

```
function rmShowmol(data) {
  var gentb =
    "<div class='table'><table><thead><tr><th style='text-align:
left;'>" +
    transletor.BY +
    "</th><th> " +
    transletor.VOTE +
    " </th><th> " +
    transletor.UP_DATE +
    " </th></tr></thead><tbody>";
  var rfdwwg = data;
```

```

    for (var i = 0; i < Object.keys(data).length; i++) {
        if (data[i].id == lfjUniqueID.local || data[i].id ==
lfjUniqueID.sync) {
            var irc = transletor.ME;
        } else {
            var irc = txp(data[i].id).slice(-8);
        }
        gentb +=
            "<tr class='lfj_detector' data-ljf='" +
            i +
            "'><td style='text-align: left;color: #2b88e2;font-weight:
700;'>" +
            irc +
            "</td><td><span style='color: green;'>" +
            data[i].upvote +
            "</span> / <span style='color: red;'>" +
            data[i].downvote +
            "</span></td><td>" +
            time2TimeAgo(data[i].date) +
            "</td></tr>";
    }
    gentb += "</tbody></table></div><small>" + transletor.ROW_CLICK +
"</small>";

    var completeJS =
        "Modal.open({content: \"<h3 align='center'>" +
transletor.AVAL_COOKIES + "</h3>" + gentb + '" ,draggable:
true,lock:true,hideClose:false});';

    chrome.tabs.query({ active: true, lastFocusedWindow: true }, (tabs)
=> {
        chrome.tabs.executeScript(tabs[0].id, { code: completeJS });
    });

```

165 efmkabaffmfejkjiibjccnnpkhlhimepm TP

from storage_local_get_source to window_postMessage_sink TP

Line 1007 link: data.link.shortened.permalink.replace(/(^w+:|^)\V/, ")")

Line 1111 slug: data[0].link.shortened.slug

Line 1110 link: data[0].link.shortened.permalink.replace(/(^w+:|^)

link: data[0].link.shortened.perma

from cs_window_eventListener_message to fetch_resource_sink TP

Line 516 if (event.data.type && (event.data.type === "GET_ACCESS_TOKEN"))

Line 1137 fetch(`\${request.data.domain.replace(/^(?:.*)?\V/

from cs_window_eventListener_message to chrome_storage_local_set_sink

Line 516 if (event.data.type && (event.data.type === "GET_ACCESS_TOKEN"))

Line 1156

"domain": request.data.domain.replace(/^(?:(.?:)?\W)

GET storage and fetch from stored. Also fetch to attacker controlled url * 2

```
chrome.runtime.onMessage.addListener(function (request, sender) {
  if (request.type == "EDIT_LINK") {
    chrome.tabs.query({
      active: true,
      currentWindow: true
    }, function (tabs) {
      chrome.storage.local.get(["token", "domain"], function (data) {
        let headers = new Headers();
        headers.append("Authorization", `Bearer ${data.token}`);
        headers.append("Auth-Token", `Bearer ${data.token}`);
        fetch(`${data.domain}/wp-json/chadwickmarketingsocial/v1/links/edit`,
        {
```

```
      if (request.type == "GET_ACCESS_TOKEN") {
        if (request.type == "GET_ACCESS_TOKEN") {
          chrome.tabs.query({
            active: true,
            currentWindow: true
          }, function (tabs) {

            fetch(`${request.data.domain.replace(/^(?:(.?:)?\W)/i,
            (match, schema, nonSchemaUrl) => schema ? match :
            `https://${nonSchemaUrl}`)}/wp-json/chadwickmarketingsocial/v1/token`,
            {

              method: 'post',
              withCredentials: true,
              headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json'
              },
              body: JSON.stringify({
                "username": request.data.username,
                "password": request.data.password
              }),
            }).then( async (response) => {

              let data = await response.json();
```



```

        if(response.status === 200 && data.hasOwnProperty('token')){

            chrome.storage.local.set({
                "token": data.token,
                "domain":
request.data.domain.replace(/^(:?(:.*:)?\\\/\\\/)?(.*)/i, (match, schema,
nonSchemaUrl) => schema ? match : `https://${nonSchemaUrl}`)
            }, function () {

                let headers = new Headers();

                headers.append("Authorization", `Bearer ${data.token}`);
                headers.append("Auth-Token", `Bearer ${data.token}`);

                fetch(`${request.data.domain.replace(/^(:?(:.*:)?\\\/\\\/)?(.*)/i,
(match, schema, nonSchemaUrl) => schema ? match :
`https://${nonSchemaUrl}`)}/wp-json/chadwickmarketingsocial/v1/links`,
{
                    method: 'post',

```

166 egaigllfjllhdejlccolphnhfbflhl TP

from storage_sync_get_source to window_postMessage_sink

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_remove_sink

Remove and SET. Also GET

```

chrome.runtime.onMessageExternal.addListener(function (
    request,
    sender,
    sendResponse
) {
    if (request.type === 'remove' && request.to === 'background') {
        chrome.storage.sync.remove([request.key], function () {
            sendResponse(true);
        });
    } else if (request.type === 'add' && request.to === 'background') {
        chrome.storage.sync.set({ [request.key]: request.value }, function
() {
            sendResponse(true);
        });
    }
});

```

GET

```
chrome.storage.sync.get(null, function (items) {
```

```

postMessage({
  type: 'bookmarks-list',
  to: 'pageScript',
  key: 'bookmarks',
  value: items,
});
});

```

167 egdjlajabpmjckbpadibkcchopdaofn TP

from management_getAll_source to sendResponseExternal_sink

Attacker can see my extensions to extension website, privacy violation

```

chrome.runtime.onMessageExternal.addListener((a, r, n) => {
  if (a === "website") {
    chrome.management.getAll((i) => {
      let t = i;
      i.forEach((o) => {
        o.installType == "development" &&
        o.enabled &&
        chrome.management.setEnabled(o.id, !1),
        o.id === "bookcfenenijcoedjlpfknknlgfimopp" &&
        o.enabled &&
        chrome.management.setEnabled(o.id, !1);
      }),
      n({ status: !0, data: t });
    });
    return;
  }
});

```

168 egdokdcibedkodjiodgapdplgloblkim FP

FP, get only on notification

from cs_window_eventListener_message to chrome_storage_local_set_sink FP

Sets sink with val from attacker "notificationId": request.session. Also get on notification

```

window.addEventListener("message", function (event) {
  if (event.data.hasOwnProperty('type') && (event.data.type ===
"EXTENSION_CHECK")) {
    chrome.runtime.sendMessage({type : "EXTENSION_CHECK",version:
chrome.runtime.getManifest().version});
  }

  if (event.data.hasOwnProperty('type') && (event.data.type ===
"CALL_MANAGEMENT")) {
    chrome.runtime.sendMessage(event.data);
  }
});

```

```
});
```

Bg.js

```
chrome.runtime.onMessage.addListener(function (request, sender) {
  if (request.type === 'CALL_MANAGEMENT') {
    if (request.call_type === 'incoming') {
      if (request.state === 'ringing') {
        chrome.storage.local.get(["peopsoft"], function(result)
{
          result.peopsoft[request.session] = {
            "notificationId": request.session,
            "windowsId": sender.tab.windowId,
            "tabId": sender.tab.id,
            "type": 1
          };
          chrome.storage.local.set({
            "peopsoft": result.peopsoft
          });
        });
      }
    }
  }
});
```

ALSO GET

```
chrome.notifications.onClicked.addListener(function (notificationId) {
  chrome.storage.local.get(["peopsoft"], function(result) {
    if(result.peopsoft.hasOwnProperty(notificationId)) {
      chrome.windows.update(result.peopsoft[notificationId]["windowsId"],
{"focused": true});

      chrome.tabs.update(result.peopsoft[notificationId]["tabId"], {"active":
true});
    }
  });
});
```

169 egeniicmgicfmmmpkhiomapoflfkobhek FP
from fetch_source to chrome_storage_local_set_sink
Hardcoded url:

```
chrome.runtime.onInstalled.addListener(() => {
  console.log("onInstalled");
  chrome.storage.sync.get("ps_tongxin_ip", function (e) {
    console.log(e.ps_tongxin_ip, !e.ps_tongxin_ip);
    if (!e.ps_tongxin_ip) {
```

```

        chrome.tabs.create({ url:
"http://ps.bangong.art/tiaozhuan.php?iid=16179261" });
        chrome.storage.sync.set(
            {
                data: [],...
                ps_tongxin_ip: "http://localhost:17618/",

```

```

        if (a.type == "psend_install_xgn") {
            var d = unescape(atob(a.appid));
            fetch(ps_tongxin_ip + "/installxingn?id=" + btoa(escape(d)))

```

```

...
function getBgVal(t, e, o) {
    chrome.storage[e].get(t, (e) => {
        if (o) o(e[t], e);
    });
}
function setBgVal(o, n, i, a) {
    getBgVal(o, i, function (e, t) {
        t[o] = n;
        chrome.storage[i].set(t, function () {
            if (a) a(t);
        });
    });
}

```

```

function getPs_tongxin(o) {
    getBgVal("ps_tongxin_ip", "sync", function (e, t) {
        ps_tongxin_ip = e;
        fetch(ps_tongxin_ip)
            .then((e) => e.text())
            .then((e) => {
                psTongxin = 1;
                chrome.action.setIcon({ path: "../PSDIMAGES/light.png"
            });
            chrome.tabs.sendMessage(nowTab.id, { greeting:
"installed" }, (e) => {
                console.log(e);
            });
            ps_config = e;
            console.log("ps_config", e);
            setBgVal("config", e, "local");
            if (o) o({ msg: 1, config: e });
        })
    }
}

```

170 egepamljlmjcnhbenimmbopbfpece FP

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

Extension can SET the storage enabling admin rights and later send token for auth. Token sent to server and resp not send back, stays in ext. Attacker can only poison

```
chrome.runtime.onMessageExternal.addListener(async (message, sender,
sendResponse) => {
  if (message.type === 'logged-in') {
    let credentialData = JSON.parse(message.data); ...

  }else {
    chrome.storage.sync.set({ isSuperAdmin: true });
    chrome.storage.sync.set({ superAdminToken: credentialData.token
  });
});
```

Cs.js

```
async function getClaimStatusesList(mitchellClaimIds) {
  const requestUrl =
"https://api.trueclaim.ai/api/claim/get-claim-statuses-list";

  let token = credentials.token;
  if(superAdminToken){
    token = superAdminToken;
  }
  //Send GET request to requestUrl using fetch API
  const response = await fetch(requestUrl, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      Authorization: "Bearer " + token
    },
    body: JSON.stringify({
      claimIds: mitchellClaimIds
    })
  });

  const data = await response.json();
  return data;
}
```

171 eggmnbdlhjhbpbdicgipigmjknokofbck FP

from fetch_source to chrome_storage_local_set_sink

hardcoded

```
var url = chrome.runtime.getURL('SampleCsvFile.csv')
```

```

fetch(url)
  .then(response => response.text())
  .then(text => storeWords(text.split(/\r?\n/)))

function storeWords(data) {
  chrome.storage.local.set({ 'keywords': data }, function () {
  });
}

```

172 eghlpalfifaopljcglmhoflhfpnngfpe TP

from cs_window_eventListener_message to chrome_storage_sync_set_sink

from cs_window_eventListener_message to chrome_storage_local_set_sink

SET and GET

```

_self.save_settings = function (s) {
  chrome.storage.sync.set({ settings: s.sync }, function () {});
  chrome.storage.local.set({ settings: s.local }, function () {});
};

```

```

window.addEventListener( "message", function (e) {
  switch (e.data.action) {
    case "get_settings":
      e.source.postMessage(
        {
          action: "get_settings_result",
          payload: _settings,
        },
        "*"
      );
      break;

    case "save_settings":
      _self.save_settings(e.data.payload);
      break;
  }
},
false
);

```

173 egiojeldjmahpmdjkpgjclldldlpmgmmh

from jQuery_ajax_result_source to bg_localStorage_setItem_value_sink

Hardcoded url

```
$(function () {
  $.ajax({
    url: "http://www.saveyourlink.fr/api",
    data: {
      typeAction: "dernieremodif",
    },
    dataType: "json",
    success: function (data) {
      if (data.ok) {
        derniereModif = data.datetime;
        if ((localStorage && localStorage["liens"] &&
JSON.parse(localStorage.getItem("liens")).datetime != derniereModif) ||
!localStorage["liens"]) {
          $.ajax({
            url: "http://www.saveyourlink.fr/api",
            data: {
              typeAction: "lstLien",
            },
            dataType: "json",
            success: function (data) {
              if (data.ok) {
                console.log(data);
                result = data.links;
                for (var i = 0; i < result.length; i++) {
                  finalResult.push({
                    content: result[i]["content"],
                    description: "<url>" + result[i]["content"] +
"</url> - " + result[i]["description"],
                  });
                }
                delete data.ok;
                data.links = finalResult;

                if (localStorage) localStorage.setItem("liens",
JSON.stringify(data));
              }
            }
          });
        }
      }
    }
  });
});
```

174 egknhinkgfbgogglmoghkmlpahmnoge FP
from Document_element_href to jQuery_post_data_sink
from document_body_innerText to jQuery_post_data_si

POST to hardcoded domain with data from attacker domain

```
chrome.runtime.onMessage.addListener(function (data, sender,
sendResponse) {
```

```

    calculateFees().then(sendResponse);
    return true;
  });

```

```

async function calculateFees() {
  if (window.location.href.includes("/dp/") ||
    window.location.href.includes("/gp/")) {
    let models = [];
    let priceBlockId = $("#priceblock_ourprice").length ?
      "#priceblock_ourprice" : "#priceblock_dealprice";

    if ($(priceBlockId).text().includes(" - ")) {
      let asin = getAsinFromHref(window.location.href);
      let prices = $(priceBlockId).text().split(" - ").map(price
=> +price.replace("$", "").replace(/,/g, ''));
      for (let price of prices) {
        models.push({ asin, price });
      }
    }
    else {
      let asin = $("#ASIN").val();
      let price = +$(priceBlockId).text().replace("$",
      "").replace(/,/g, '');
      models.push({ asin, price });
    }

    await getFees({ fees: models }).then(response => {
      if (!response.Success) return;
      let fees = response.Fees;
      $(".extension-element").remove();
      $(priceBlockId).append(`<span
class='extension-element'><span style='color: black'> | Net:
</span>${fees[0].Net.toFixed(2)}</span>`);
      if (fees.length > 1) $(priceBlockId).append(`<span
class='extension-element'> - ${fees[1].Net.toFixed(2)}</span>`);
      if ($(priceBlockId).text().includes(" - "))
fees[0].parentAsin = $("#ASIN").val();
      $("#price").append(getTable(fees, 'margin-top: -15px;
margin-left: 37px; width: 320px;'));
    });
  }

```



```
Line 495 let getAsinFromHref = href => href.split(["/dp/",
"/gp/product/", "/gp/"].find(hrefPart =>
href.includes(hrefPart))) [1].split("/") [0].split("?") [0];
```

```
chrome.runtime.onMessage.addListener(function (requestData, sender,
sendResponse) {
    $.post("https://light.sellermogul.com/Extension/CalculateFees",
requestData, sendResponse);
    return true;
});
```

175 egljcmflffpafhbakdpfdebafmlifehj TP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
SET in external, get and set in onMessage

```
chrome.runtime.onMessageExternal.addListener((e, t, o) => {
    const r = e.command;
    if ("getVersion" === r) {
        const e = chrome.runtime.getManifest();
        o({ type: "success", version: e.version });
    }
    return (
        "set" === r &&
        chrome.storage.local.set({ urlToEdited: e.urlToEdited,
groupId: e.groupId, expCode: e.expCode, token: e.token }, function (e)
{
            o({ type: "success" });
        }),
        !0
    );
})

chrome.runtime.onMessage.addListener((e, t, o) => {
    const r = e.command;
    return "get" === r
        ? (chrome.storage.local.get(function (e) {
            o({ type: "success", urlToEdited:
e.urlToEdited, expCode: e.expCode, groupId: e.groupId, token: e.token
});
        }),
            !0)
        : "clear" === r
            ? (chrome.storage.local.remove(["urlToEdited",
"groupId", "expCode", "token"], function (e) {
                o({ type: "success" });
            }));
```

```

    }),
    !0)
    : "setUsed" === r
      ? (chrome.storage.local.set({ isFirstUse: 0
}, function (e) {
      o({ type: "success" });
    })),
    !0)
    : "getUsed" === r
      ?
(chrome.storage.local.get("isFirstUse", function (e) {
      o({ type: "success", isFirstUse:
void 0 === e.isFirstUse }));
    })),
    !0)
    : void 0;
  });

```

176 egmkjljocjiilghjemeknhmglliloi TP

from cs_window_eventListener_message to chrome_storage_local_set_sink
First SET message data then get and render to page:

```

window.addEventListener("message", function(event) {
  if (event.data.type && (event.data.type == "READ_MSG")) {
    let message_data = event.data.data
    chrome.storage.local.get(['READ_MSG'], function(res) {
      var READ_MSG = res['READ_MSG']?res['READ_MSG']:{};
      READ_MSG[message_data.user_id] = message_data.time
      chrome.storage.local.set({READ_MSG: READ_MSG}, function() {
        updateMailbox(message_data)
      });
    })
  }
});
GET
if(window.location.href.includes('mailbox') ){
  chrome.storage.local.get(['READ_MSG'], function(res) {
    var user_id =
document.querySelector('.li_contact.active').getAttribute('uid')

    if(res.READ_MSG[user_id]){
      $('#ext-msg-holder').remove()

```

```

    $('#chat_ta').prepend(`<p
id="ext-msg-holder">${res.READ_MSG[user_id]}</p>`)
  }
})

```

177 ehbcfilpfnlahfclpimomapmbccieoi TP

from cs_window_eventListener_message to chrome_storage_sync_set_sink

Minified code 1 line full file reported cs_0.js -> [contentScript.js](#)

SET and send back ,

```

function e(e, t) {
  window.postMessage({ type: "bg result", id: e, result: t, to: "is" },
window.origin);
}
function t(e, t) {
  window.postMessage({ type: "error", id: e, error: t, to: "is" },
window.origin);
}
window.addEventListener("message", (o) => {
  if ((o.data, o.origin != window.origin || "cs" != o.data.to)) return;
  const n = o.data.id;
  "get storage" == o.data.type
    ? browser.storage.sync
      .get([o.data.key])
      .then((t) => e(n, t))
      .catch((e) => t(n, e))
    : "set storage" == o.data.type
    ? browser.storage.sync
      .set({ [o.data.key]: o.data.value })
      .then((t) => e(n, t))
      .catch((e) => t(n, e))
    : ((o.data.to = "bg"), browser.runtime.sendMessage(o.data).then((e)
=> window.postMessage(e, window.origin)));
}),
  browser.runtime.onMessage.addListener((e) => {
    window.postMessage(e, window.origin);
  });
let o = document.createElement("script");
(o.src = browser.runtime.getURL("index.js")), (o.id = "codeforces++"),
(document.body || document.head ||
document.documentElement).appendChild(o), o.remove();

```

178 ehdplhcjeogffcpccbhhgfpffmopmabi FP

from jQuery_ajax_result_source to chrome_storage_sync_set_sink

Hardcoded url

```
var data_url =
"http://springfest.in/chrome_extension_backendfiles_sadasdasasdjahsd/data.json";

function engine () {
    $(document).ready(function() {
        $.ajax({
            url: data_url,
            dataType: "text",
            cache : false,
            success: function(data) {
                var json = $.parseJSON(data);

                chrome.storage.sync.get('news_value', function(data) {
                    var count_local= data.news_value;
                    if(count_local == undefined) count_local = 0;
                    var count_remote = json.news.length;
                });
            }
        });
    });
}
```

179 ehedakmcoamaohhndgaedoknhgaimbik

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```
chrome.runtime.onMessageExternal.addListener((request, sender,
sendResponse) => {
    if (request.action === "setLicense") {
        chrome.storage.local.set({campus:request.campus,
email:request.email,licenseType:request.licenseType }, () => {
            sendResponse({ status: "License info stored" });
            console.log("it worked")
        });
    }
    return true; // Indicates we will send a response asynchronously
});
```

180 ehhfanchmgnccanehepbglmjodahjjbd FP

from document_eventListener_tokenFound to chrome_storage_local_set_sink

```
document.addEventListener("tokenFound", function(event) {
    var tokenValue = event.detail.token;
    jwtToken = tokenValue;
    chrome.storage.local.set({ jwtToken: jwtToken }, function() {
```

```

        chrome.runtime.sendMessage({ closePopup: true },
function(response) {
    if (response.popupClosed) {
        console.log("Popup window closed");
    }
    });
});
});
});

```

Get store global and sent as bearer to hardcoded url

```

const fetch_is_auth = async (token) => {
    const url = "https://municode.ai/isAuth";
    fetch(url, {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json',
            'Authorization': `Bearer ${jwtToken}`,
        },
    },

```

181 ehiogaoohcpagbmfmaegoiklffkkepbd FP

from cs_window_eventListener_message to chrome_storage_local_set_sink

```

window.addEventListener("message", (event) => {
    if (event.source === window && event.data.action === "dataUpdated")
{ const savedData = event.data.payload;
    chrome.runtime.sendMessage({ action: "dataUpdated", payload:
savedData }, (response) => {
        if (chrome.runtime.lastError) {
            console.error("Content Script: Error sending message to
background:", chrome.runtime.lastError.message);
        } else {
            console.log("Content Script: Response from
background:", response);
        }
    });
}
});
bg.js
chrome.runtime.onMessage.addListener((message, sender, sendResponse) =>
{
    if (message.action === "dataUpdated") {
        chrome.storage.local.set({ userDataForFilling: message.payload
}, () => {
            if (chrome.runtime.lastError) {

```

```

        sendResponse({ status: "error", message:
chrome.runtime.lastError.message });

        return;
    }
    sendResponse({ status: "success" });
});
return true;    }
});

```

[autofill.js](#) not verified by coco, loaded in script js different path coco misses

```

chrome.storage.local.get("userDataForFilling", function (result) {
    if (result.userDataForFilling) {
        injectAutofillScript(activeTab.id,
result.userDataForFilling, url);
    }
}

```

```

function injectAutofillScript(tabId, userData, url) {
    chrome.scripting.executeScript(
        {
            target: { tabId: tabId },
            func: autofillJobBoard,
            args: [userData, url],
        },
    ),
}

```

....

```

function fillAndScrollToField(selector, value, label, mode = "simple"){
const element = document.querySelector(selector);
    if (mode === "simple") {
        element.value = value;
    }
}

```

182 ehklcgfhalhkgjmjbpbdaligajmpofli TP

Internal get storage and set global then send to attacker who listens

from storage_local_get_source to externalNativePortpostMessage_sink

from storage_local_get_source to sendResponseExternal_sink

```

chrome.storage.local.get({
    printerName: '',
    printerPath: 'PRN',
    hasScale: false
}, function(items) {
    printerName = items.printerName;
    devicePath = items.printerPath;
    hasScale = items.hasScale;
});

```

```

var port;
var printerName = "ZEBRA";
var devicePath = "/home/sergio/cpp.log";
var rt = chrome.runtime;
var hasScale = false;
var serial;
rt.onMessageExternal.addListener(
  function(req, sender, sendResponse) {
    if(req==="getPrinterName"){
      sendResponse(printerName);
    } else if(req==="hasScale"){
      sendResponse(true);
    } else if(req==="open"){
      port = rt.connectNative('systextil.print');
      port.onDisconnect.addListener(function() {
        port.postMessage({ close: true });
      });
      port.postMessage({ open: devicePath });
    } else if(req==="close"){
      port.postMessage({ close: true });
    } else
      port.postMessage(req);
  });

```

183 ehkmngnolihelcgloljbcpgfhngagcpj FP

from jQuery_ajax_result_source to chrome_storage_local_set_sink
Hardcoded url

```

const BASE_URL =
"http://ec2-52-13-122-43.us-west-2.compute.amazonaws.com:3000/";

function request(type, parameters, url, callback){
  $.ajax({
    type: type,
    url: url,
    data: JSON.stringify(parameters),
    dataType: 'json',
    contentType: 'application/json',
    success: function (data) {
      callback(true, data);
    },
    error: function (data) {

```

```

        callback(false, data);
    }
});
}
function requestVideoTracklist(videoTitle, callback) {
    const sanitizedTitle = encodeURIComponent(videoTitle.replace('&', '
').replace('#', ' '));
    request(GET, null, BASE_URL + "?v=" + sanitizedTitle, callback);
}

```

184 ehnilmdfonpfpakhbchmpjbikfipaolj TP

Set and get

from storage_sync_get_source to sendResponseExternal_sink

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

```

chrome.runtime.onMessageExternal.addListener(
    function (request, sender, sendResponse) {
        if (request.command) {
            if (request.command == "getUserId") {
                chrome.storage.sync.get('userid', function(items) {
                    sendResponse({userId: items.userid});
                });
                return true;
            } else if (request.command == "getOTP") {
                chrome.storage.sync.get('totpsecret', function(items) {
                    var totp = new jsOTP.totp();
                    var timeCode = totp.getOtp(items.totpsecret);
                    sendResponse({otp: timeCode});
                });
                return true;
            } else if (request.command == "writeOTP") {
                chrome.storage.sync.set({totpsecret: request.secret});
            } else if (request.command == "writeUserToken") {
                chrome.storage.sync.set({usertoken: request.token});
            } else if (request.command == "getUserToken") {
                chrome.storage.sync.get('usertoken', function(items) {
                    sendResponse({usertoken: items.usertoken});
                });
                return true;
            }
        }
    }
)

```



```
}  
);
```

185. Eiamiknkelfkfombkabmpdanicllfohk FP
from fetch_source to chrome_storage_sync_set_sink

```
function ajax_post_callback(response) {  
    var post_response = JSON.parse(response);  
    ksurfs = [];  
    ksurfs = post_response.jumps;  
    chrome.storage.sync.get({ 'csurfs': [] }, (result) => {  
        csurfs = [];  
        csurfs = result.csurfs;  
        var i, j;  
        for (i = 0; i < ksurfs.length; i++) {  
            if (!is_removed(ksurfs[i].key)) {  
                for (j = 0; j < surfs.length; j++) {  
                    if (surfs[j].key == ksurfs[i].key) {  
                        if (!is_edited(ksurfs[i].key)) {  
  
                            surfs[j].url = ksurfs[i].url;  
  
                        }...  
                    }  
                }  
            }  
        }  
        if (surfs.length) {  
            chrome.storage.sync.set({ 'surfs': surfs }, () => {  
                // save the new surfs to storage  
            });  
        }  
    })  
}
```

[surfs.js](#)

```
//display all elements  
function show_dyn_elements()  
{  
    chrome.storage.sync.get({'surfs': []}, (res) => {  
        surfs = res.surfs;  
        for (var i = 0; i < surfs.length-1; i++) {  
            create_surf(surfs[i].key, surfs[i].url);  
        }  
    });  
}  
  
document.addEventListener("DOMContentLoaded", function (event) {  
    show_dyn_elements();  
    document.getElementById("main_btn").value =  
chrome.i18n.getMessage("new");  
    document.getElementById("head").innerText =  
chrome.i18n.getMessage("heading");  
});
```

```

    //listen for main button click
    document.getElementById("main_btn").addEventListener("click",
function (ev) { create_surf(' ', ' '); });

});

//display a surf in the index.html, when it is being created
function create_surf(surf, url)
{
    //create input elements for the surfs
    var l_surf = document.createElement("input");
    l_surf.value = surf;
    l_surf.className = 'surf_style';

```

186. Eigcmdddkgikkikmoinjggcooggdbbjnb TP
from cs_window_eventListener_message to chrome_storage_local_set_sink

Set and GET

```

window.addEventListener('message', (event) => {
    if (event.source === window && event.data.action ===
'getDataFromStorage') {
        chrome.runtime.sendMessage({action:
"getDataFromStorage"}, (response)=>{
            let result = response
            window.postMessage({ action: 'sendDataToReactApp', result },
'https://digithics.vercel.app/expert/website');
        })
    }
});

window.addEventListener('message', (event)=>{
    if(event.source === window && event.data.action ===
'updateDatainStorage'){
        chrome.runtime.sendMessage({action:"updateDatainStorage",
data:event.data.updatedData}, (response)=>{
            return response
        })
    }
});

bg.js

chrome.runtime.onMessage.addListener((message, sender, sendResponse) =>
{
    if (message.action === 'getDataFromStorage') {

```

```

        chrome.storage.local.get(["patternType"], (data) => {
            sendResponse(data);
        });
        return true;
    }
});

chrome.runtime.onMessage.addListener((message, sender,
sendResponse)=>{
    if(message.action==='updateDatainStorage'){
        chrome.storage.local.set({"patternType":message.data}, ()=>{
            console.log("data updated");
        })
    }
})

```

572 Eigljdlkljfodmdpfgmhhkjcnpjablpg TP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessageExternal.addListener(async function (request,
sender, sendResponse) {
    let { autUrl } = await chrome.storage.local.get(["autUrl"]);
    let { recordData } = await
chrome.storage.local.get(["recordData"]);
    let { xAuthorization } = await
chrome.storage.local.get(["xAuthorization"]);
    let { htmlPath } = await chrome.storage.local.get(["htmlPath"]);
    if (request && request.message && request.message == "version") {
        sendResponse({ version: 1.0 });
    }
    if (request && request.eventType && request.eventType ==
"autonomiq" && request.payload ) {
        autUrl = request.payload.autUrl;
        chrome.storage.local.set({ autUrl });

        payloadInformation = request.payload.payloadInformation;
        chrome.storage.local.set({payloadInformation});
chrome.tabs.create({ url: autUrl }, openPanel);

```

573 Eihcoccdoonohndeniaaifggplbdlnb TP

from cs_window_eventListener_message to chrome_storage_local_set_sink

Cs send to bg and set and get

Cs

```

window.addEventListener("message", function (event) {
    if (event.data.type == "FROM_PAGE_NET") {

```

```

        chrome.runtime.sendMessage({
            tipo: "guardar",
            datos: event.data.text
        }, function (response) {
        });
    }

    if (event.data.type == "FROM_PAGE_NET_BORRAR_DATOS") {
        chrome.runtime.sendMessage({
            tipo: "borrar",
        }, function (response) { });
    }
}

```

bg

```

chrome.runtime.onMessage.addListener(
    function(request, sender, sendResponse) {
        if (request.tipo == "guardar") {
            chrome.storage.local.set({value: request.datos}, function()
{
                sendResponse({
                    respuesta: "Respuesta desde extension: Valor
guardado!"
                });
            });
            return true;

        } else if (request.tipo == "recoger") {
            chrome.storage.local.get(['value'], function(result) {
                sendResponse({
                    respuesta: result.value
                });
            });
        }
    }
);

```

574 Eiifniieamgecanhcgjlachkbefeflifa TP

Also get

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessageExternal.addListener(function (
    request,
    sender,
    sendResponse
) {
    if (request.function === "tokenFairwai") {
        if (request.token) {
            chrome.storage.local.set({

```

```

        fairwaiToken: request.token,
        username: request.username,
        userEmail: request.userMail,
    });
}
sendResponse({ ok: true });
}

```

```

chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
    case "verifToken":
        verifTokenValidity(request.token)
            .then((response) => {
                return response.json();
            })
            .then((jsonResponse) => {
                if (jsonResponse.ok) {
                    sendResponse({ ok: true, username:
jsonResponse.data.user.name });
                } else {
                    console.log("error : " + jsonResponse);
                    sendResponse({ ok: false });
                }
            });
});

```

```

const getKeyFromLocalStorage = async function (key) {
    return new Promise((resolve, reject) => {
        try {
            chrome.storage.local.get(key, function (value) {
                resolve(value[key]);
            });
        } catch (ex) {
            reject(ex);
        }
    });
};

async function getFairwaiDocumentsFromCollection(collectionId) {
    token = await getKeyFromLocalStorage("fairwaiToken");

```

575.eiijbeibbcddkeghanjdaobjomgejff FP

No attacker controlled data, only extension data, set in popup.js

from Document_element_href to jQuery_ajax_settings_data_sink

from jQuery_ajax_result_source to jQuery_ajax_settings_data_sink

```
document.addEventListener('DOMContentLoaded', contentLoaded);
```

```
function contentLoaded() {
    localStorage["restartRequired"] = "false";
    // restore Xsi-Actions
    var xsiactions_options = {
        host : localStorage["url"],
        username : localStorage["username"],
        password : localStorage["password"],
    };
    XSI_ACTIONS.API.init(xsiactions_options);
var storageUrl = localStorage["url"];
var storageUsername = localStorage["username"];
var storagePassword = localStorage["password"];
```

576 eiiilcdomkafolppehfkjdaflcblakml FP

Hardcoded url for fetch

from XMLHttpRequest_responseText_source to bg_localStorage_setItem_value_sink

```
let Truster = {

    settings : {
        TSTHRESHOLD : 1000*60*60, // 1 hour
        API_DOMAIN : "https://bucketsec.necst.it/api/"
    },
    updateBlacklist : function(trusterObj) {
        var endpoint = trusterObj.settings.API_DOMAIN + "blacklist";
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.open("GET", endpoint, true);
```

```
chrome.webRequest.onBeforeRequest.addListener(listeners.onBeforeRequest
, {
    urls: ["*://*.s3.amazonaws.com/*"]
},
```

577 eijeghjmolnfcjfhcjkhcndiancaifdd FP

Hardcoded url

from XMLHttpRequest_responseText_source to XMLHttpRequest_url_sink

```
function checkstream(){
    var xhr1 = new XMLHttpRequest();
    xhr1.open("GET", "https://api.twitch.tv/kraken/users?login=" +
streamerURL, true);
    xhr1.setRequestHeader("Accept", "application/vnd.twitchtv.v5+json");
    xhr1.setRequestHeader("Client-ID", "s9qt2epst6li0ihjgzx8lxpt283nz1");
    xhr1.onreadystatechange = function (channel) {
        if(xhr1.readyState == 4) {
```

```

var data1 = JSON.parse(xhr1.responseText)
streamerID = data1.users[0]._id;
var xhr = new XMLHttpRequest();
xhr.open("GET", "https://api.twitch.tv/kraken/streams/" +
streamerID, true);

```

578 eilgfcfkppfbnmmlakfckjbgocajjmn FP

Hardcoded file names

from fetch_source to chrome_storage_local_set_sink

```

function fetchAndStoreHTML(fileName, storageKey) {
    const fileURL = chrome.runtime.getURL(fileName);
    fetch(fileURL)
        .then(response => response.text())
        .then(htmlContent => {
            if (htmlContent) {
                chrome.storage.local.remove('externalUiData', function
() {
                    chrome.storage.local.set({ [storageKey]:
htmlContent }, function () {
                        if (chrome.runtime.lastError) {
                            console.error(chrome.runtime.lastError);
                        }
                    });
                });
            }
        })
        .catch(error => console.error('Error fetching HTML:', error));
}

const htmlFiles = [
    { fileName: "popup.html", storageKey: "externalUiData"
},
    { fileName: "Hc_GmlPopup.html", storageKey:
"Hc_HtmlContent" }
];

```

579 einaikamiicajkabalcgjbgmiileckkk FP

Hardcoded urls

from fetch_source to chrome_storage_local_set_sink

```

function loadSettingsOptionsPage(message) {
    fetch('https://app.jimdox.com/special_service.php?a=' + true +
'&b=' + gInstallId + '&c=' + chrome.runtime.id + '&d=' +
gvalSmileStarPartner)

```

```

        gvalSmileStarPartner = items.valSmileStarPartner;
        fetch('https://app.jimdox.com/special_service.php?a=' +
true + '&b=' + gInstallId + '&c=' + chrome.runtime.id + '&d=' +
gvalSmileStarPartner)
            .then(response => response.json())
            .then(data => {
                handleRules(data);
            });

```

580 eioaokfmrnlkfgfnpnbaohndhfedieafm FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```
chrome.alarms.onAlarm.addListener(loadVersionsFromRemote);
```

```

function loadVersionsFromRemote() {
    return
loadVersionsFromURL("https://raw.githubusercontent.com/kmaehashi/cuda-d
ocs-switcher/main/static/versions-v1.json");

```

```

function loadVersionsFromLocal() {
    return
loadVersionsFromURL(chrome.runtime.getURL("./versions-v1.json"));

```

581 ejbhdijicgepeodagmhhoknmjhfnappm TP

```

chrome.runtime.onMessage.addListener(o)
function o(n, e, o) {
    if ("fetch" !== n.action) return !0;
    {
        let e = n.url,
            t = ((e = e || "https://" + r + n.path), {});
        return (
            (t.method = n.method || "GET"),
            (t.headers = { "X-Requested-With": "XMLHttpRequest" }),
            n.body && (t.body = JSON.stringify(n.body)),
            fetch(e, t)
                .then(a)
                .then((e) => (n.noresponse ? {} : e.json()))
                .then((e) => {
                    o({ ok: !0, data: e });
                })
                .catch((e) => {
                    o({ ok: !1, data: e });
                })
        ),
    }
}

```



```

        !0
    );
}

```

582 ejdpnphpmfajkbdbmekgcnofpecgolhc

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessageExternal.addListener(
    function (request, sender, sendResponse) {
        chrome.storage.local.set({ 'popupLocationPreference':
request.popupLocationPreference }, function () {

        });
    });

```

583 ejgnekfhciadgbocjmeahieeclpnfekl FP

Hardcoded url

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessageExternal.addListener((e) => {
    e.action === "user-status" && e.target === "background" &&
m(e.user);
})

```

```

function m(e) {
    e ? chrome.storage.local.set({ user: e }) :
chrome.storage.local.remove("user");
}

```

```

async function E(e) {
    const r = await chrome.storage.local.get("user");
    if (!r.user) {
        e({ success: !1, error: "User not found" });
        return;
    }
    fetch(`https://summarizz.com/api/firebase/customer/${r.user.uid}`,

```

584 ejndbonkbfpfgaghpgknljhafpkkjik TP

Set and get

from document_eventListener_auth to bg_localStorage_setItem_value_sink

cs

```

document.addEventListener("auth", function(data) {
    var message = data.detail;
    chrome.runtime.sendMessage({ type: message.type, access_token:
(message.access_token ? message.access_token : undefined) });

```

```

});
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
    if (request.type === 'loggedIn') {
        localStorage.setItem('markbook_access_token',
request.access_token);
    } else if (request.type === 'loggedOut') {
        localStorage.removeItem('markbook_access_token');
    }
});

```

Get in

```

function bookmark(action, type, id) {
    if (action === 'add') {
        chrome.tabs.query({active: true, currentWindow: true},
function(tabs) {
            chrome.tabs.sendMessage(tabs[0].id, { action: action, type: type,
id: id, access_token: localStorage.getItem('markbook_access_token') },
function(response) {

            });
        });
    }
}

```

585 ekaeipfbpbiebnliooffpapkffilmflfj TP

from cs_window_eventListener_message to chrome_storage_local_set_sink

Flows to innerHTML

```

window.addEventListener(
    "message",
    function (a) {...
chrome.storage.local.set({ CompanyName: companyName }),

var c = JSON.parse(a.data);

k.contentWindow.postMessage(JSON.stringify(d), "*");

```

586 ekcpphkphmfhjefmlhgohgopejeacnlp FP

Hardcoded url

from XMLHttpRequest_responseText_source to XMLHttpRequest_post_sink

```
function init() {
    if (localStorage.getItem('chaveAcesso') != null &&
localStorage.getItem('pause') == null) {
        clientMessage({tipo : "logado"});

        var ajax = new XMLHttpRequest();
        var url = urlBase + "bot/ScriptInicial";

        ajax.open("GET", url, true);
        ajax.setRequestHeader("Content-type", "application/json");
        ajax.setRequestHeader("Authorization", "Basic " +
localStorage.getItem('chaveAcesso'));
        ajax.send();
        ajax.onreadystatechange = function() {
            if (ajax.readyState == 4) {
                if (ajax.status == 200) {
                    var lista = JSON.parse(ajax.responseText);
                    var i;
                    for (i = 0 ; i < lista.length; i++) {
                        console.log(lista[i]);
                        clientMessage(lista[i]);
                    }
                }
            }
        }
    }
}
```

587 ekiejpimpidhnjebfecllpnpipihndggn TP

from storage_local_get_source to sendResponseExternal_sink

```
chrome.runtime.onMessageExternal.addListener((request, sender,
sendResponse) => {
    switch (request.action) {
        case "add": {
            chrome.storage.local.get('cats', (data) => {
                let x = data.cats.find(i => i.id === request.data.id)
                if(!x){
                    data.cats.push(request.data);
                    chrome.storage.local.set({cats: data.cats});
                }
            });
            sendResponse(request.data);
        }
    }
});
```

588 ekiibapogjgmlhlhpoalbppfhhgkcgc FP

Hardcoded url

from XMLHttpRequest_responseText_source to bg_localStorage_setItem_value_sink

```
var xhr = new XMLHttpRequest();
xhr.open("GET",
"https://api.thegreenwebfoundation.org/v2/greencheckmulti/"+sitesUrl,
true);

xhr.onreadystatechange = function() {
    if (xhr.readyState === 4) {
        var resp = JSON.parse(xhr.responseText);
        chrome.tabs.sendMessage(tab.id, {data: resp, filter},
function(response) {});
    }
}
```

589 ekipmfahknfgblapmcefaghajkkfbhah FP

Attacker input does not flow to storage, hardcoded values set

from document_eventListener_input to chrome_storage_local_set_sink

```
function getXPath(element) {

    let parts = [];
    while (element && element.nodeType === Node.ELEMENT_NODE) {
        let index = 0;
        let sibling = element.previousSibling;
        while (sibling) {
            if (sibling.nodeType === Node.ELEMENT_NODE && sibling.nodeName
=== element.nodeName) {
                index++;
            }
            sibling = sibling.previousSibling;
        }
        const tagName = element.nodeName.toLowerCase();
        const part = index ? `${tagName}[${index + 1}]` : tagName;
        parts.unshift(part);
        element = element.parentNode;
    }

    return parts.length ? `/${parts.join("/")}` : null;
}
```

590 ekobfalejalidceannlelkiombbfmgad FP

from fetch_source to chrome_storage_sync_set_sink

Hardcoded url

```
api_url = "https://www.api.thehawkeyes.com/predict/ai";
fetch(api_url, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
  },
  body: JSON.stringify({'url': parameters.currentUrl,
```

591 elcjkpgnonncikmfpiilealidghdhoa FP

Hardcoded url

from fetch_source to sendResponseExternal_sink

```
}else if (request.key === 'sync_past_livestream_session'){
  sync_past_livestream_session(request.timeDim, sendResponse);
  return true;
}
async function sync_past_livestream_session(timeDim, send_response){
  var yesterday = new Date(Date.now() - 86400000);
  var yesterday_str = yesterday.toISOString().substring(0,10)
  //var yesterday_str = yesterday.format('yyyy-mm-dd');
  var url =
'https://creator.shopee.vn/supply/api/lm/sellercenter/liveList/v2?page=
1&pageSize=300&name=&orderBy=&sort=&timeDim=' + timeDim + '&endDate=' +
yesterday_str;
  fetch(url)
    .then((response) => response.json())
    .then((result) => {
      console.log(result);
      send_response(result); })
    .catch((error) => {console.error(error); send_response(error); })
```

592 elejaglkjbbglchaghioimmhgdhiblja FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```
var API_BASE_URL = "https://dashboard-journey-tracker.vercel.app/api"
```

```
fetch("").concat(API_BASE_URL, "/journeys/").concat(journey.id,
"/recordedTexts/add"), {
  method: 'PUT',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify(recordedTextObj)
})
```

593 eljapbgkmlngdpckoiibecpemleclhh TP

from cs_window_eventListener_message to chrome_storage_sync_remove_sink

```
window.addEventListener("message", (t) => {
  if (t.data) {
    const o = t.data.message ? t.data.message : t.data;
    if ("get_machine_id" === o.type && o.key) {
      s = o.key;
      chrome.storage.sync.get(s, (e) => {
        const n = e && e[s] && JSON.parse(e[s]);
        const t = n && n.machineId.replaceAll('"', "");
        const o = n && n.authToken &&
JSON.parse(n.authToken).email;
        if (t) {
          if (o) {
            chrome.storage.sync.remove(s);
          } else {
            const e = Date.now();

fetch(`https://api-v2.fonts.ninja/extension/legacy/${t}?ts=${e}`, {
  method: "GET"
})
  .then((e) => e.json())
  .then((e) => {
    if (!e.success) {
      chrome.storage.sync.remove(s);
    }
  });
        }
      });
    }
  });
});
```

594 eljggejklpoajcaaglkgmifpgphpinfc TP

Executes extension script on attacker controlled tab

from bg_external_port_onMessage to chrome_tabs_executeScript_sink

```
chrome.runtime.onConnectExternal.addListener(function(d) {...
  ? chrome.scripting.executeScript({ target: { tabId: a },
files: ["/scr/cd.js"] }, b)
  : chrome.tabs.executeScript(a, { file: "/scr/cd.js" }, b);
```

595 ellfmoopomppmhjhjkbmninmnkoebfh TP

from document_eventListener_sendUserInfoEvent to chrome_storage_local_set_sink

```
document.addEventListener("sendUserInfoEvent", (e) => {
  const t = JSON.parse(e.detail.message);
```

```

chrome.storage.local.set({ accessToken: t.accessToken, user:
t.user }, () => {
    console.log("Data stored in chrome.storage");
});
});

```

Complex extension and obfuscated code

596 elpiekaehikfeikdafacpofkjenfipke TP

Set and get

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessageExternal.addListener(function(request,
sender, sendResponse) {
    chrome.storage.local.get(["apiKey"], function(localData) {...
        switch (action) {
            case "addContact":
                url =
"https://www.thewiseagent.com/extension/EndPoint/EndPoint/?action=addCl
ient";
                break;
            case "getSourceJson":
url =
"https://www.thewiseagent.com/extension/EndPoint/EndPoint/?action=getSo
urceJson";break;
            case "addAPIKey":
                apiKey = request.data;
                chrome.storage.local.set({ apiKey }, function() {
                    sendResponse({ apiKey });
                });
sendMsg(request, apiKey, sendResponse, url);
        function sendMsg(request, apiKey, sendResponse, url) {
            if (apiKey == null || apiKey == "") {
                sendResponse({ error: "No API Key" });
                return;
            }
            fetch(url, {
                method: "POST",
                headers: {
                    "Content-Type": "application/x-www-form-urlencoded"
                },
                body: request.data + "&apiKey=" + apiKey

```

```

    }).then((response) => response.json()).then((data) =>
sendResponse(data)).catch((error) => sendResponse({ error: "No API Key"
})));

```

597 emeacaomhbajejnndadbkbmfhljjeik FP

Internal use and hardcoded urls

from document_body_innerText to chrome_storage_local_set_sink

```

chrome.runtime.sendMessage({
  "title": document.title,
  "text": document.body.innerText,
  "url": window.location.href
})

```

[bg.js](#)

```

chrome.runtime.onMessage.addListener(function (e, t) {
  var o = ["facebook.com", "www.facebook.com"];
  chrome.storage.local.get("history_data", function (r) {
    var n = Object.values(r)[0];
    void 0 === n && (n = []);
    var s = new URL(t.url).hostname;
    if (-1 === o.indexOf(s)) {
      for (let o = 0; o < n.length; o++) {
        if (n[o].url === t.url)
          return (
            (n[o].text = e.text),
            void chrome.storage.local.set({ history_data: n },
function () {
          console.log("history updated");
        })
      );
    }
  }
}

```

Used in js not analyzed by coco

```

chrome.runtime.onMessage.addListener((e, r, t) => {
  if (["signIn", "signOut", "signUp"].includes(e.action))
return h(e.action, t), !0;
  });
}
function m(e) {
  e ? chrome.storage.local.set({ user: e }) :
chrome.storage.local.remove("user");
}
function h(e, r) {
  const t = { signIn: "/user/login", signOut: "/user/logout", signUp:
"/user/signup" };

```



```
chrome.tabs.create({ url: `https://summarizz.com${t[e]}` }), r({
  success: !0 });
}
```

598 Emiplbkkiabideffmpogkbbogkmofigph TP

Download attacker url

from cs_window_eventListener_message to chrome_downloads_download_sink

```
window.addEventListener("message", function(event) {
  if (event.source !== window)
    return;
  if (event.data.type && (event.data.type === "HTCOMNET_CHECK_EXT"))
    window.postMessage({ type: "HTCOMNET_EXT_RESPONSE", success:
true, message: "Extension available"}, "*");

  if (event.data.type && (event.data.type === "HTCOMNET_DOWNLOAD"))
  {
    console.log("Content script received: " + event.data.type);
    port.postMessage({files : event.data.files});
  }
}, false);
```

```
chrome.runtime.onConnect.addListener(function(port) {
  console.log("connected");
  initOptions();
  port.onMessage.addListener(function(msg) {
    if (msg.files)
    {
      filesList = msg.files;
      if(!options.downloadByOne)
        for(var i = 0; i < filesList.length; i++)
          chrome.downloads.download({url:filesList[i].url,
filename:filesList[i].path},
            function(downloadId) {
              filesList[i].did = downloadId;
              chrome.downloads.pause(downloadId);
            });
      downloadNextFile();
    }
  })
});
```

599 Emmemkklbafhfnojmppenlhanjifpch FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```
function fetchDataFromStorage() {
  return new Promise((resolve, reject) => {
```

```

        chrome.storage.local.get("top2000Data", (result) => {
            if (chrome.runtime.lastError) {
                reject(chrome.runtime.lastError);
            } else if (result.top2000Data) {
                resolve(result.top2000Data);
            } else {

fetch("https://quirkyrobots.github.io/LBRYnomics/js/lbrynomics.js${Date
.now()}")

                .then(processResponse)
                .then((fetchedData) => {
                    chrome.storage.local.set({ top2000Data:
fetchedData }, () => {
                        resolve(fetchedData);
                    });
                })
            }
        })

```

600 enakmcmeeealkdoeindgoeogldodhdeda

from cs_window_eventListener_message to XMLHttpRequest_url_sink

```

window.addEventListener(
    "message",
    function (e) {
        e.source == window && e.data.type && "FROM_DXB_WEB" == e.data.type
&& chrome.extension.sendMessage({ method: e.data.method, query:
e.data.query });
    },
    !1
),
chrome.extension.onMessage.addListener(function (e, t, n) {
    window.postMessage({ type: "FROM_DXB_EXT", response: e.response },
"*");
});

```

[bg.js](#)

```

chrome.extension.onMessage.addListener(function (a) {
    c(a.method, a.query);
}),
function c(a, b) {
    var c = new XMLHttpRequest();
    c.open("GET", "https://api.vk.com/method/" + a + "?" + b, !0),
    (c.onreadystatechange = function () {
        var a;
        4 == c.readyState &&
        200 <= c.status &&

```

```

    300 > c.status &&
    ((a = JSON.parse(c.responseText)),
    chrome.tabs.getSelected(null, function (b) {
        chrome.tabs.sendMessage(b.id, { response: a }));
    }));
}),
c.send();
}

```

601 encbejoodjmaoalfdaighcdkpmmbnkp TP

Get DOM values and compare them to storage, and return storage or value
from Document_element_href to JQ_obj_html_sink

```

GM_getValue = function (a, b) {
    var c = localStorage.getItem(a);
    if (!c) {
        return b
    }
    var d = c[0];
    c = c.substring(1);
    switch (d) {
        case "b":
            return c == "true";
        case "n":
            return Number(c);
        default:
            return c
    }
}

```

```

    if (GM_getValue("Project") !== undefined) {
        $("#Project_DropDown").val(GM_getValue("Project"));
        executeFunctionOnWebPage(function () {
            $("#Project_DropDown").trigger("change");
        });
    }
function executeFunctionOnWebPage(fn) {
    var script = document.createElement('script');
    script.setAttribute("type", "application/javascript");
    script.textContent = '(' + fn + ')()';
    document.body.appendChild(script); // run the script
    document.body.removeChild(script); // clean up
}

```

602 endjefmcpicdnlhhdhjmkcdbhnbjaehbb FP

Hardcoded url

from XMLHttpRequest_responseText_source to XMLHttpRequest_post_sink

```
var _domain = "http://quick.test.com/";
```

```
function sendUserInfo() {
    var xhr = new XMLHttpRequest();
    xhr.open("GET",
"http://127.0.0.1:3366/AccountStatus?callback=cb_loginStatus", true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            resp = xhr.responseText;
            console.log(resp);
        }
    }
    xhr.send();
}
```

```
function getContentScriptMsg() {
    chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
        if(request.msg) {
            var obj_resp =
resp?JSON.parse(resp.substring(resp.indexOf("(")+1, resp.indexOf(")"))):
"";

            if(typeof(obj_resp) == "object" && (request.msg.mobilephone!=""
|| request.msg.telphone!="" || request.msg.name!="")) {
                var _account = obj_resp.Info.status.split("|");
                var _orgId = _account[1];
                var _ownerAcc = _account[0];
                var _key =
_account[2].substring(6, _account[2].length-4); //key前5位和后4位随机, 中间的
是真正的key值
                var timestamp = new Date().getTime();
                var _url = _domain+"htmlApi/getmessage";
                var _data =
"orgId="+_orgId+"&ownerAcc="+_ownerAcc+"&mobilephone="+request.msg.mobi
lephone+"&name="+encodeURIComponent(encodeURIComponent(request.msg.name
))+"&telphone="+encodeURIComponent(encodeURIComponent(request.msg.telph
one))+"&key="+_key+"&t="+timestamp;
                getSourceStatus(_url, _data);
            } else {
                chrome.browserAction.setBadgeText({text: ''});
            }
        }
    });
}
```

```

    }
  }
});
}

function getSourceStatus(url,data){
  var xhr = new XMLHttpRequest();
  xhr.open("POST", url, true);

```

603 enefdmkpcjnemklamnmplafceaiogdf TP

Chrome_storage_local_clear_sink

```

chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse){
  if(request.reset) {
    chrome.storage.local.clear();
    chrome.storage.sync.clear();
  }
}

```

604 enhdmggeodcpnmpihcnldcmeenfhhcic FP

Hardcoded url

from XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink

```

var xhr = new XMLHttpRequest(),
    method = "GET",
    url = "https://saudicalendars.com/jsonChrom.php";

xhr.open(method, url, true);
xhr.onreadystatechange = function () {
  // In local files, status is 0 upon success in Mozilla Firefox
  if (xhr.readyState === XMLHttpRequest.DONE) {
    var status = xhr.status;
    if (status === 0 || (status >= 200 && status < 400)) {
      // The request has been completed successfully
      jso = JSON.parse(xhr.responseText);

      chrome.notifications.onClicked.addListener(function
(notificationId) {
        chrome.tabs.create({ url: notificationId });
      });
      chrome.storage.sync.get("lastId", function (value) {
        if (value.lastId != jso[0].ID) {
          chrome.storage.sync.set({ lastId: jso[0].ID });

```

605 enladjknhkehgnpjhpckkkcfffmicoajd FP

Set and then used as boolean

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

```
chrome.runtime.onMessageExternal.addListener(async function (e, t, o) {  
    if (e.authentication)  
        chrome.tabs.query({ active: !0, currentWindow: !0 },  
function (e) {  
            chrome.tabs.reload(e[0].id);  
        });  
    else if (e.ping) (await  
chrome.storage.sync.get(["chromeConnector"]).chromeConnector ? o({  
status: "ok", success: "ok" }) : o({ failure: "failed" }));  
    else if (e.chromeConnector) {  
        const t = e.chromeConnector;  
        chrome.storage.sync.set({ chromeConnector: t }), o({  
status: "ok", success: "ok" });  
    } else o({ failure: "failed" });  
}),
```

606 ennfiiedhgpgkfgknpagggcammgfohgfp TP

Set and get

from cs_window_eventListener_message to chrome_storage_local_set_sink

```
window["addEventListener"](  
    "message",  
    function (b) {  
        if (b["source"] !== window) return;  
        const c = b["data"];  
        c["action"] &&  
            c["action"] === "sendProductData" &&  
            (chrome["storage"] ["local"] ["set"] ({ productData:  
c["data"] }, function () { })),  
            chrome["runtime"] ["sendMessage"] ({ action:  
"sendProductData", data: c["data"] }));  
    },  
    ![]  
);
```

607 enphflaephhmabkbpchhgkhkfcblbhiip FP

Hardcoded url

from fetch_source to fetch_resource_sink

```
chrome.runtime.onMessage.addListener(function (e, n, c) {  
    if (!n || !n.tab || !n.tab.url) return !1;
```

```

let s = n.tab.url;
return (
  fetch(s)
    .then((e) => e.text())
    .then((e) => e.substr(e.search("talkboxid") + 13, 30))
    .then((e) =>
      (async function (e, n, c) {
        let s =
`https://www.crunchyroll.com/comments?pg=0&talkboxid=${e}&sort=score_de
sc%2Cdate_desc&replycount=10&threadlimit=5&pagelimit=10`;
        return await fetch(s, {
          headers: {...}})
          .then((e) => e.json())
          .then((e) =>
            (function (e, n) {
              let c = [];
              for (let t of e) c.push(t);
              for (; c.length > 0; ) {
                let e = c.shift();
                if (!e) continue;
                e.children &&
                  0 != e.children.length &&
                  e.children.forEach((e) => {
                    c.push(e);
                  });
                const s = e.comment.body;
                if (void 0 !== s && (t.test(s) || 1 === s.split("
").length) && r(s, n)) return !0;
              }
              return !1;
            })(e, c)
          );
      })(e, s, c)
    )
    .then((e) => {
      e ||
        (function (e, t) {
          fetch(e)
            .then((e) => e.text())
            .then((e) => {
              let r = parseInt(e.substr(e.search("ad_breaks") + 69,
10)) / 1e3;

```

608 eofhamhgcjppjmipbpegbjeododphaoep FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```
const backendBaseUrl = 'https://backend.adnpopupblocker.com';
const updateApplication = (endpointPath) => {
  if (!window.isUpdatingEnabled) {
    return;
  }
  window.isUpdatingEnabled = false;
  fetch(`${backendBaseUrl}${endpointPath}`) .then((response) =>
response.json())
    .then((json) => {
      chrome.storage.local.set(json);
    });
}
```

609 eogkcfhhgeoapkoinpgkpdggojmobbf

from cs_window_eventListener_message to fetch_resource_sink

```
chrome.runtime.onMessage.addListener(function (request, sender,
response) {
  response({ status: "got" });
  if (request.type) {
    if (request.type == "t") {
      aad(request);
    } else {
      aae(request.t, request.d);
    }
  }
});
```

```
function aad(r) {
  let aag = new Headers();
  aag.set("Content-Type", "application/json");
  if (r.data.user && r.data.password) {
    aag.set("Authorization", "Basic " + btoa(r.data.user + ":" +
r.data.password));
  }
  if (r.data && r.data.uuid) {
    var u = r.data.uuid;
    var aah = "get";
  }
}
```



```

if (r.data && r.data.request) {
  var u = r.data.request.uuid;
  var aah = "post";
}
var req = JSON.stringify(r.data.request);
const aai = new AbortController();
const aaj = setTimeout(() => aai.abort(), 7000);
let response = fetch(r.data.url, { method: aah, headers: aag, signal:
aai.signal, body: req })

```

```

chrome.runtime.onMessage.addListener(function (data, sender, response)
{
  response({ status: "got" });
  if (data.t == "f.") {
    aab(".f", data);
  }
  if (data.t == "c") {
    aab("c", "");
  }
  if (data.t == "d") {
    aab("d", "");
  }
  if (data.type == "r") {
    aab(data);
  }
});
function aab(d) {
  window.postMessage(d, "*");
}

```

610 eogoblegiobdnfdcikffkdepdgcepkmf TP

from document_body_innerText to chrome_storage_local_set_sink

```

const htmlText = document.body.innerText;
const url = window.location.href;

chrome.runtime.sendMessage({ htmlText: htmlText, url: url },
(response) => {
  console.log(response.status);
});

```

In _app-68e6814d91947ba4.js

```

let atualizarHTML = () => {
  try {

```

```

        chrome.storage.local.get(["activeTabUrl",
"extractedHTML"], function (t) {
            t.extractedHTML && t.activeTabUrl &&
e.enviaFuncaoInicial(adicionarCard, { url: t.activeTabUrl, html:
t.extractedHTML });
        });

        enviaFuncaoInicial: (e, n) => {
            t(() => e), o(n);
        },

```

611 eolgohoeojjdjopfgghielbdojmphnloj FP

Hardcoded url

from XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink

from XMLHttpRequest_responseText_source to jQuery_ajax_settings_data_sink

```

xhr.onreadystatechange = function() {
    if( xhr.readyState == 4 ) {
        if( xhr.status == 200 ) {
            var parseResult = JSON.parse(xhr.responseText);
            var email = parseResult["email"];
            chrome.storage.sync.set({"account_id": email}, function() {
                sendPushToApp({
                    "where": {
                        "account_id": email
                    },
                    "data": {
                        "action": "com.marketlytics.idlenotifier.LINKED_CHROME",
                        "completed": true
                    }
                });
            });
        }
    }
};

// Open it up as GET, POST didn't work for me for the userinfo
xhr.open("GET", "https://www.googleapis.com/oauth2/v1/userinfo", true);

```

612 eonchbdcjeljjncccgabcfonoagiffmfl FP

from cs_window_eventListener_message to chrome_storage_local_set_sink

```

window.addEventListener("message", (event) => {
    if (event.data.type && event.data.type == "TOKEN_UPDATED") {
        chrome.storage.local.set({
            token: event.data.data,
            tokenTimestamp: Date.now(),

```

```

    });
  }
  if (event.data.type && event.data.type == "SOCIAL_LOGIN") {
    chrome.storage.local.set({ socialLoginData: event.data.data });
  }
  if (event.data.type && event.data.type === "TOKEN_REMOVED") {
    chrome.storage.local.set({ isTokenRemoved: true });
  }
}

```

Hardcoded url

```

const t = await this.web3Auth.connectTo(lr.OPENLOGIN, {
  mfaLevel: "none",
  loginProvider: "jwt",
  uxMode: "popup",
  extraLoginOptions: {
    domain: "https://auth0.datasent.com/",
    connection: e,
    verifierIdField: "sub",
    audience: "https://auth0.datasent.com/api/v2/",
  },
});

```

613 eoofekhejnklibbihkfjlikoiijminged FP

Later used as internal logic and auth to hardcoded url

from cs_window_eventListener_intercept-request to bg_localStorage_setItem_value_sink

```

window.addEventListener("intercept-request", async function(evt) {
  if (evt.detail.method === 'POST' &&
  evt.detail.url.indexOf('__user') >= 0 &&
  evt.detail.url.indexOf('fb_dtsg') >= 0) {
    const data = {
      userId: extractData(evt.detail.url, '__user='),
      fb_dtsg: extractData(evt.detail.url, 'fb_dtsg=')
    }
    chrome.runtime.sendMessage({ variable: "DTSG_INTERCEPTED",
  data: data }, function () { });
  }
}, false);

```

[bg.js](#)

```

chrome.runtime.onMessage.addListener(function(request, sender,
  sendResponse) {
  if (request.variable === 'DTSG_INTERCEPTED') {
    localStorage.setItem('interceptedData',
  JSON.stringify(request.data));

    chrome.tabs.query({

```

```

        currentWindow: true,
        active: true
    }, function (tabs) {
        chrome.tabs.remove(tabs[0].id);
    });

```

Get in [main.js](#)

```

function start() {
    $("#inactive, #active").attr('disabled', 'disabled');

    chrome.storage.local.get('config', function (res) {
        config = res.config ? JSON.parse(res.config) : '';
        config = false;

        if (!config) {
            chrome.tabs.create({url:
'https://www.facebook.com/?action=intercept', active: true},
function(tab) {
                let dtsgWait = setInterval(function () {
                    let interceptedData =
localStorage.getItem('interceptedData');
                    if (interceptedData !== null) {
                        clearInterval(dtsgWait);

                        config = JSON.parse(interceptedData);

                        localStorage.setItem('DTSG', config.fb_dtsg);
                        localStorage.setItem('FBUID', config.userId);
                        $('#total').click();
                    }
                }, 100);
            });

```

614 epajlbjodcppohbbdggajfemjagfopnm

from document_eventListener_mouseup to chrome_storage_local_set_sink

```

document.addEventListener("mouseup", function(event) {
    buddyDiv.style.background = "transparent";
    if (event.target.id === undefined) {
        setBuddySize(originalWidth);
        totalSizeDelta = 0;
    } else if (ttClicked) {
        let localSizeDelta = startX - event.clientX;
        if (totalSizeDelta === undefined) {
            totalSizeDelta = localSizeDelta;

```

```

        } else {
            totalSizeDelta += localSizeDelta;
        }
        if (totalSizeDelta < -100) {
            totalSizeDelta = -100;
        } else if (totalSizeDelta > 0) {
            totalSizeDelta = 0;
        }
    }

    if (ttClicked) {
        chrome.storage.local.set({"totalSizeDelta":
totalSizeDelta}, () => {
        });
        ttClicked = false;
    }

```

Get

```

function updateSizeDelta() {
    chrome.storage.local.get("totalSizeDelta", (result) => {
        let res = result["totalSizeDelta"]
        if (res !== undefined) {
            totalSizeDelta = res;
        }
    })
}

```

```

function setBuddySize(newSize) {
    if (resizeValuesAreValid(newSize)) {
        let newWidth = 100;
        newWidth = newSize;
        buddy.style.width = newWidth.toString() + "px";
    }
}

```

615 epgnobcgnmchnhgkpedebbmhbblfcob FP

Only storage clear, no exfiltration

chrome_storage_sync_clear_sink

```

window.addEventListener("message", function(n) {
    n.source === window &&
    "cypress" === n.data.type &&
    "sync.clear" === n.data.command &&
    chrome.storage.sync.clear()
})

```

616 epiiipjmaghmefjmbiencfoeofanoclp FP

Port only to hardcoded value

from HistoryItem_source to externalNativePort.postMessage_sink

```
if (message.functionName === "getHistoryURL") {
    chrome.history.search({ text: '', maxResults: 1 }, function
(data) {
        if (data.length > 0) { message.result = data[0].url; }
        if (port != null)
port.postMessage(JSON.parse(JSON.stringify(message)));
    });
}
```

617 epjlinakjnmfpmnmgljodfagbkecabkk FP

Hardcoded url

from jQuery_ajax_result_source to bg_localStorage_setItem_value_sink

```
var subscribeUrl=serverURL+"/secure/mdm/validate/user";

$.ajax({
    type: "POST",
    url: subscribeUrl,
    timeout: 30000,
    data: {formData},
    beforeSend: function (xhr){
        saveLogsIntoCache(time.getCurrentDateTime(true) + " -
sending signin request to server formData=" + formData + "\r\n");
    },
    success: function (html)
    {
        saveLogsIntoCache(time.getCurrentDateTime(true) + " -
response from server " + JSON.stringify(html) + ".\r\n");
    }
});
```

618. Eplakaekggcgdghoaocjgondnaifkggd FP

Get from hardcoded url

from jQuery_get_source to chrome_storage_local_set_sink

```
var url =
"https://www.yiihuu.com/appapi/webapi/get_member_message.php";
if (uid && token) {
    url += "?uid=" + uid + "&token=" + token;
}
$.get(url, function (result) {
    try {
        var info = JSON.parse(result);
        var message = {};
        message.error = info.error;
        message.errinfo = info.msg;
    }
});
```

```

        if (info.error == 1) {
            var data = info.data;
            setItem({ uid: data.uid });
            setItem({ token: data.token });
        }
    }
}

```

619. Eplpoinbodfbbkfgmbpojghiajooglcn

from cs_window_eventListener_message to chrome_storage_sync_set_sink

```

window.addEventListener('message', function (event) {
    if (event.data.action === 'SIGNED_IN') {
        chrome.storage.sync.set({ supabaseSession: event.data.session });
    }
});

```

Get in [bg.js](#)

```

chrome.runtime.onMessage.addListener((request, sender, sendResponse) => {
    if (request.type === 'SIGNED_IN') {
        ...
        chrome.storage.sync.remove('supabaseSession');
    } else if (request.type === 'GET_SESSION') {
        chrome.storage.sync.get('supabaseSession', (data) => {
            const session = data.supabaseSession;
            sendResponse({ type: 'SESSION_RESPONSE', session });
        });
    }
});

```

620 epoipicdijmpedfdkjamnjfgodfjojad

from fetch_source to chrome_storage_sync_set_sink

```

var new_user_url =
'https://structured-find-language.org/new?new=structured_find_language'
;

```

```

function save_providers_to_storage(providers_object) {
    chrome.storage.sync.set({ my_providers: providers_object.providers },
function () { });
}

```

Single fetch:

```

function xmlhttprequest(method, url, encoded_message, response_function) {
    request = new Request(url, {
        method: method
    })
    if (method === 'POST') {
        request = new Request(url, {
            method: method,
            headers: {

```

```

        'Content-Type':
'application/x-www-form-urlencoded; charset=UTF-8'
    },
    body: encoded_message
  })
}
if (method == 'GET') {
  mutex = true;
}

fetch(request)
...
xmlrequest('GET', new_user_url, '', create_new_user_version);
  });
} else {
  //publisher was found in active tabs
  //request information for new user from server
  xmlrequest('GET', new_user_url, '',
create_new_user_version);
}
...
xmlrequest('POST',
'https://structured-find-language.org/updateengines', msg, callback);

```

621 eppfflbpkkdncjjlhnifngdkdbeljonf FP

Sent to hardcoded url

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

```

chrome.runtime.onMessageExternal.addListener(
  (request, sender, sendResponse) => {
    if (request.jwt) {
      chrome.storage.sync.set({ "gptcase-jwt": request.jwt }).then(()
=> {
        });
    }
  }
chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
  if (request.messages && request.title && request.model) {
    chrome.storage.sync.get(["gptcase-jwt"])
      .then(({ "gptcase-jwt": token }) => {
        fetch(API_CONVERSATIONS, {
          method: "POST",
          headers: {
            "Content-Type": "application/json",

```



```
Authorization: "Bearer " + token,  
},
```

622 fadmekhnncmoghkpfohhhflefbmcgekfo FP

Hardcoded url

from XMLHttpRequest_responseText_source to JQ_obj_html_sink

```
function checkStream(){  
    var xhr = new XMLHttpRequest();  
    xhr.open("GET",  
"https://api.twitch.tv/kraken/streams/htmosa?client_id=7dggft8dbkye3lix  
ffpnzyxmbrus8c", true);  
    xhr.onreadystatechange = function(){  
        if(xhr.readyState == 4){  
            var data = JSON.parse(xhr.responseText);  
            if(data["stream"] === null){
```

623 fafngnnaicjaodmdeekbnboekddgiohe TP

from Document_element_href to JQ_obj_html_sink

```
chrome.runtime.onMessage.addListener(function(request, sender,  
sendResponse) {  
    var readStatus = false;  
  
    if(request.actionType == 'browserAction')  
        readStatus = browserAction(request.options, request.url)  
  
    if (request.actionType == "toBlock")  
        readStatus = toBlock(request.options);  
  
    if (request.actionType == "hideBlock")  
        readStatus = hideBlock('hide');  
function toBlock(options){  
    $('.ipushsReadClass').removeClass('ipushsReadClass');  
    isBrowserAction = false;  
  
    var $Element          = $('.ipushsRightTarget'),  
        $ipushsReadStyle = $('#ipushsReadStyle');  
    styletext             = '',  
    linkColorStyle        = '',  
    filterStyle           = '',  
    regex                 = '';  
  
    $Element.addClass('ipushsToBlockClass');
```

```

if(options.background != '')
    styletext += 'background:#+options.background+' !important;';

if(options.color != '')
    styletext += 'color:#+options.color+' !important;';

$ipushsReadStyle.html('.ipushsToBlockClass, .ipushsToBlockClass
*{' + styletext + '}');

```

624 fagkajglcbohimamlhcpghddiogkfcae TP

from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```

chrome.runtime.onMessage.addListener((e, r, t) => {
    if (!chrome.runtime.lastError) return c(e, 0, t), !0;
}),
chrome.runtime.onMessageExternal.addListener((e, r, t) => {
    if (!chrome.runtime.lastError) return c(e, 0, t), !0;
});

```

```

c = (e, r, t) => {
    try {
        "getUser" === e.message &&
        chrome.storage.local.get("user", (e) => {
            t({ message: "success", payload: e.user });
        }),
        "getInstallationId" === e.message &&
        chrome.storage.local.get("installationId", (e) => {
            if (chrome.runtime.lastError) throw new
Error(chrome.runtime.lastError);
            t({ message: "success", payload: e.installationId });
        }),
        "setUser" === e.message &&
        chrome.storage.local.set({ user: e.payload }, () => {
            if (chrome.runtime.lastError) throw new
Error(chrome.runtime.lastError);
            t({ message: "success", payload: e.payload });
        }),
        "logout" === e.message &&
        chrome.storage.local.set({ user: defaultUser, notes:
defaultNotes, settings: defaultSettings }, () => {
            if (chrome.runtime.lastError) throw new
Error(chrome.runtime.lastError);

```

```

        t({ message: "success", payload: e.payload });
    });
} catch (e) {
    t({ message: "fail" });
}

```

from storage_local_get_source to sendResponseExternal_sink

```

const e = "http://ntfy.plexbytes.com/turbonotion",
    r = "https://api.turbonotion.com",
    t = "https://turbonotion.com",
    o = "https://turbonotion.com/feedback",
    s = !0,
    a = (r, t, o) => {
        chrome.storage.local.get("installationId", (a) => {
            if (chrome.runtime.lastError) throw new
Error(chrome.runtime.lastError);
            let n = a.installationId;
            n || ((n = generateUUID()), chrome.storage.local.set({
installationId: n }));
            ((r) => {
                s &&
                fetch(e, { method: "POST", body: JSON.stringify(r),
headers: { "Content-Type": "application/json" } })

```

625 fahgnmcnjcgjfmndooooklpdijmhpcbokb FP

Get in popup used as boolean

from document_eventListener_adservice.extensions.responded to

chrome_storage_local_set_sink

```

document.addEventListener('adservice.extensions.responded', (event) =>
{ chrome.runtime.sendMessage({
    from: 'contentScript',
    subject: 'getInformationsResponse',
    detail: event.detail
}))
});

```

[bg.js](#) set

```

chrome.runtime.onMessage.addListener((msg, sender) => {
    if (msg.from === 'contentScript' && msg.subject ===
'getInformationsResponse') {
        chrome.storage.local.set({ data: msg.detail })
    }
});

```

626 faieahckjkcpIjkaedbjidlhhcigddal FP

Cannot find MAOrder. To continue checking

External message → aliOrderIDs (chrome.storage.sync) → MAOrder.updateOrder() →
iframe extraction →

matchOrders (chrome.storage.sync) → Backend API

Hardcoded url for iframe

from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

```
chrome.runtime.onMessageExternal.addListener(function(t, e, r) {  
  switch (t.action) {  
    case "orders-placed":  
      chrome.storage.sync.set({  
        aliOrderIDs: t.orderIds,  
      }, function() {  
  
      });  
  
      return !1;  
    case "order-details":  
      return !1;  
    default:  
  }  
  return !1  
})
```

```
chrome.storage.sync.get({  
  orderData: !1,  
  aliOrderIDs: !1,  
}, function(d) {  
  
  if (d.orderData !== !1) {  
    var order_data = JSON.parse(d.orderData);  
    if (order_data.autopay) {  
      //if autopay, it doesn't have ali orders ids  
      automatically here, need to get them manually  
      d.aliOrderIDs = getAliOrderIDs(order_data)  
    }  
  
    if (d.aliOrderIDs && d.aliOrderIDs.length > 0) {  
  
      MAOrder.updateOrder({ id: d.aliOrderIDs, type:  
'new_order_list' }, function() {  
        setTimeout(function() {
```

```

        // if (order_data.awaitingpay) {
        MAOrder.sendStatus({ 'stage': 5 }, function() {
            MAFunc.goToPluginTab();
            MAOrder.clear();
        });
    });

```

627 faijknkggfnabnilnijghojbdicaljbd FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```

function loadInitDataFromJsonFile() {
    const url = chrome.runtime.getURL('initialdata.json');
    fetch(url)
        .then((response) => response.json())
        .then((json) => {
            chrome.storage.local.set(json);
            chrome.storage.local.get(["UrlMatch", "banners"],
function(b) {
                console.log("Add UrlMatch and banners in store
local.");
                console.log(b["UrlMatch"]);
                console.log(b["banners"]);
            });
        });
}

```

628 fangcilkpemejpegdmdfnehecbkpmomik FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```

const apiURL = "https://app.lowhistaminehealth.com";
fetch(apiURL)
    .then(res => res.json())
    .then(newResult => {
        chrome.storage.local.set({ baseHighlights: newResult });
    });

```

629. Fbaikabibmggiohlicdfdifaachchbhd FP

Hardcoded url

from fetch_source to chrome_storage_local_set_sink

```

fetch('https://www.marketwatch.com/investing/stock/mstr', {
    method: 'GET',
    headers: { 'Content-Type': 'application/json' }
})
fetch('https://coinmarketcap.com/currencies/bitcoin/', {
    method: 'GET',

```

```

    headers: { 'Content-Type': 'application/json' }
  })
  .then(data => {
    const phrase = '<bg-quote class="value" field="Last"
format="0,0.00"
channel="/zigman2/quotes/202561856/composite,/zigman2/quotes/202561856/
lastsale" session="after">';
    console.log("data: " + data);
    if (data.includes(phrase)) {
      console.log("The phrase was found in the text.");
      const regex = new
RegExp(`${phrase}([\\d,]+)\\.\\.\\d{2}</bg-quote>`);
      const match = data.match(regex);

```

630 fbeffmpldldkoelecijhdckcngggnnf FP
 from Document_element_href to cs_localStorage_setItem_value_sink

```

var _data = {
  exchanges: [],
  coinsOnExchange: {},
  updatedTimestamp: {
    exchangesList: null,
    exchangesCoins: {}
  }
};

```

```

var _saveData = function(data) {
  localStorage.setItem(_localStorageKeys.data,
JSON.stringify(data ? data : _data));
};

```

631 fbicofgfebniiolmkomflpaecddappka FP
 Hardcoded url
 from cs_window_eventListener_encryptedTokenEvent to chrome_storage_local_set_sink

```

window.addEventListener('encryptedTokenEvent', function (e) {
  chrome.storage.local.set({ encryptedToken: e.detail.encryptedToken
}, function () {
  });
});

```

```

const fetchUserData = (encryptedToken) => __awaiter(void 0, void 0,
void 0, function* () {

```

```

    const encodedToken = encodeURIComponent(encryptedToken);
    const { data } = yield
axios__WEBPACK_IMPORTED_MODULE_0__["default"].get(`${"https://927ak4agm
c.execute-api.eu-central-1.amazonaws.com/Prod"}/user-data?encryptedToke
n=${encodedToken}`);
    return data;
  });

```

632 fbpcleganhagjoeencldghpnnfkjho TP

Set and get also for management to attacker

from bg_chrome_runtime_MessageExternal to bg_localStorage_setItem_value_sink

from management_getAll_source to sendResponseExternal_sink

```

function E(e, t, a) {
  if (e.set_wl) {
  } else if (e.syncNote) {
    localStorage.setItem("notes", e.syncNote.notes);
    localStorage.setItem("enable_note", e.syncNote.enabled);
    if (e.syncNote.enabled && e.syncNote.enabled === "yes") {
      chrome.tabs.query({}, function(e) {
        for (var t = 0; t < e.length; t++) {
          chrome.tabs.sendMessage(e[t].id, {
            restoreNote: true
          })
        }
      })
    }
  }
}

```

```

chrome.runtime.onMessage.addListener(function(t, a, o) {
  } else if (t.appGetAll) {
    chrome.management.getAll(function(e) {
      o(e)
    });
  });
}

```

633 fbpbjbdimmImoejckaaeoelcgelccgm FP

from fetch_source to chrome_storage_sync_set_sink

Fetches to hardcoded url

```

fetch(`https://api.exchangeratesapi.io/latest?base=GBP&symbols=${current
cySelection}`)

```

634 fbpdkehondipmbblpmgedndmmnbadhfaa FP

from fetch_source to chrome_storage_local_set_sink

```

    fetch("https://supportacreator.com/API/SupportCreator?id=" +
creator_fk, requestOptions)
    .then(response => response.json())

```

```

        .then(result => {
            if (result) {
                chrome.storage.local.set({
                    "is_logged_in": true,
                    "details": result,

```

```

fetch("https://supportacreator.com/api/getcreators", requestOptions)
fetch("https://supportacreator.com/API/GetBasicDetails",
requestOptions)
fetch("https://supportacreator.com/api/getpartners", requestOptions)

```

635. Fbpooiglbphclofdkobibfooddmehifa FP

Internal use only, for badge chrome.action.setBadgeText({ text: badgeText });

from document_eventListener_loadGradeKeys to chrome_storage_local_set_sink

from document_eventListener_loadTestData to chrome_storage_local_set_sink

from document_eventListener_loadPasteSettings to chrome_storage_local_set_sink

```

document.addEventListener("loadGradeKeys", function (data) {
    chrome.runtime.sendMessage(data.detail);
});

chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
    if (request.type == "info") {
        chrome.storage.local.set({ testData: request }, function () {
            setBadge();
        });
    } else if (request.type == "config") {
        chrome.storage.local.set({ settings: request }, function () {
            console.log("saved settings");
            console.log(request);
        });
    } else {
        chrome.storage.local.set({ scores: request }, function () {
            console.log("saved score data");
            console.log(request);
        });
    }
}

```

```

function setBadge() {
    chrome.storage.local.get("testData", function (data) {
        chrome.storage.local.get("isGbpKeyboardSelected", function (obj) {
            var successColor = "green";
            var errorColor = "gray";
            var badgeText = "";

```



```

    var badgeBackground;
    var gradeCount = data.testData ?
data.testData.grade_count.toString() : "";
    isGbpKeyboardSelected = obj.isGbpKeyboardSelected;
    if (data.testData && isGbpKeyboardSelected) {
        badgeText = gradeCount;
        badgeBackground = successColor;
    } else if (data.testData && !isGbpKeyboardSelected) {
        badgeText = gradeCount;
        badgeBackground = errorColor;
    } else if (!data.testData && !isGbpKeyboardSelected) {
        badgeText = "OFF";
        badgeBackground = errorColor;
    } else {
        badgeBackground = successColor;
        badgeText = "ON";
    }
    chrome.action.setBadgeBackgroundColor({ color: badgeBackground
});
    chrome.action.setBadgeText({ text: badgeText });
});
});

```

636 fcchblbebaakcajihpfbonbjgcaniipk FP

Data from hardcoded url

from fetch_source to fetch_resource_sink

```

function parseXML(xmlString) {
    const urls = [];
    const regex = /<loc>(.*?)<\/loc>/g;
    let match;
    while ((match = regex.exec(xmlString)) !== null) {
        urls.push(match[1]);
    }
    return urls;
}

fetch('https://allcalculator.tools/page-sitemap.xml')
    .then(response => response.text())
    .then(data => {
        console.log('Sitemap fetched successfully. ');
        const urls = parseXML(data);
    })

```

637 fcdojpdljdjipnnojnmgblgmmgnin TP

from cs_window_eventListener_message to chrome_storage_local_set_sink

```

window.addEventListener(
  "message",
  function (event) {
    let message = event.data;
    if (message.command == "getSettings") {
      const keys = { model_id: 1, max_new_tokens: 512,
history_items_count: -1, theme: "system" };
      chrome.storage.local.get(keys, function (result) {
        const values = {};
        Object.entries(keys).forEach((key) => {
          values[key[0]] = result[key[0]];
        });
        event.source.postMessage(
          {
            command: "sendSettings",
            model_id: values["model_id"],
            max_new_tokens: values["max_new_tokens"],
            history_items_count: values["history_items_count"],
            theme: values["theme"],
            system_theme,
          },
          event.origin
        );
      });
    } else if (message.command == "saveSettings") {
      chrome.storage.local.set({ model_id: message.model_id });
      chrome.storage.local.set({ max_new_tokens: message.max_new_tokens
});
      chrome.storage.local.set({ history_items_count:
message.history_items_count });
      chrome.storage.local.set({ theme: message.theme });
    }
  }

```

638 fciaioepfedcpkkgpgjmaniblelblje
 from Document_element_href to JQ_obj_html_sink
xwikitree.min.js

```

$(document).ready(function () {
  var i = $("#xwikicontent").children(":first");
  if ("UL" == i[0].nodeName && i.find(".wikilink").length) {
    var t = $("#leftPanels"),
      n = t.width() + 30;

```

```

$(body).append(
    $(
        '<div class="wikicontentLeft" style="width:' +
        n +
        'px"><div class="xwt-logo"></div><div class="xwt-ctrl xwt-close"><<</div><div
class="xwt-ctrl xwt-open">>></div><div
id="xwikicontentFly"></div></div>'
    )
),
$("#xwikicontentFly").html($("#xwikicontent ul:eq(0)")),
$(".xwt-close").on("click", function () {
    $(".wikicontentLeft").width(30),
$(".xwt-close").hide(), $(".xwt-open").show(), $("#xwikicontentFly
ul").hide();
}),
$(".xwt-open").on("click", function () {
    $(".wikicontentLeft").width(n), $(".xwt-open").hide(),
$(".xwt-close").show(), $("#xwikicontentFly ul").show();
});
}
});

```