CoCo flagged extension position 1->100:

1. Aahnibhpidkdaeaplfdogejgoajkjgob FP
   Hardcoded url const FirstSettings = { host: 'https://localhost:8080/gui/'

    a. bg_chrome_runtime_MessageExternal → fetch_resource_sink. Change params in the fetch with a param that can be torrent server set by attacker. In this case the test

```
function addTorrents(settings, link, where) {
  GetTokenNew(settings, function (token, cookie) {
    let params = "?action=add-url&download_dir=0&token=" + token +
"&s=" + encodeURI(link);

    if (where !== ROOT_FOLDER) params += "&path=" + where;

    const myHeaders = new Headers({
      Authorization: "Basic " + btoa(settings.user + ":" +
settings.password),
    });

    return fetch(settings.host + params, {
      method: "GET",
      headers: myHeaders,
    });
  });
}
```

```
chrome.runtime.sendMessage(
  "aahnibhpidkdaeaplfdogejgoajkjgob", // BitTorrent extension ID
  {
    type: "ADD_TORRENT",
    url: "magnet:?xt=urn:btih:MALICIOUS&dn=payload",
    folder: "../../../../etc", // Path traversal
  }
);
```

2. Aajfadplooajoiepmofheifpbbhdakoa TP
    a. storage_local_get_source to window_postMessage_sink. Only get no set

```
window.addEventListener("message", function(event) {
    if (typeof event.data.method != "undefined") {
        let method = event.data.method
        if (method == "get") {
            chrome.storage.local.get(key, dateResult => {
```

```
                let messge = {}
                messge.method = "result"
                messge.result = dateResult
                window.postMessage(messge)
            })
        } else if (method == "delete") {
            chrome.storage.local.remove(key, function() {
                console.log("remove")
            })
        }
    }
})
```

3. Aakfndbelpdpdigmhlknaahhhmlmhkoo FP CoCo
    a. Sink XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink . called at start - gets data from fixed URL and stores.

```
function Twitch() {
    chrome.storage.sync.get('twitch_online', function (obj) {
        var xmlHttp = new XMLHttpRequest();
xmlHttp.open("GET",
"http://nocache.armakiss.fr/webservices/twitch.php", true);
        xmlHttp.send();
        xmlHttp.onreadystatechange = function () {
            if (xmlHttp.readyState == XMLHttpRequest.DONE) {
 if (xmlHttp.status == 200) {
 var rep = JSON.parse(xmlHttp.responseText);
 chrome.storage.sync.set({ 'twitch_title': rep.twitch_title });
chrome.storage.sync.set({ 'twitch_game': rep.twitch_game });
chrome.storage.sync.set({ 'twitch_preview': rep.twitch_preview });
chrome.storage.sync.set({ 'twitch_online': rep.twitch_online });
```

4. Aallcmelejdfkmcdijkkllobloнbeiap TP
    a. Sink document_eventListener_bs_vote to chrome_storage_local_set_sink . called at start - gets data from fixed URL and stores.
    b. Sink 2 exportFunction("saveEncounter" 2*g

```
    var exportFunction = function(name, func){
        document.addEventListener("bs_"+name, function(event){
            var result = func(event.detail.params);
            document.dispatchEvent(new
CustomEvent(event.detail.resultListenerKey, { detail: result }));})}
exportFunction("vote", function (params) {
        var profileId = params.profileId;
```

```
        var vote = params.vote;
        storage.get(profileId, function (storageObject) {
            var savedEncounter = storageObject[profileId];
            savedEncounter.vote = vote;
            storage.set(buildStorageObject(savedEncounter.user_id,
savedEncounter))
        });
    });
```

5. Aamiahgongddccmfmocapbcahonkknig FP
    a. from fetch_source to fetch_resource_sink. Roblox API with asset ID set by from outside

```
function handlePluginSafetyCheck(details) {
    const url =
`https://api.roblox.com/marketplace/productinfo?assetId=${details.url.m
atch(regURL)[0]}`
        fetch(url)
            .then(response => response.json())
            .then(data => {
                console.log("Verbose output data: ");
                console.log(data);
                if(data["AssetTypeId"] !== 38) return;
                chrome.tabs.query({active: true, currentWindow: true},
(tabs) => {
                    if(tabs.length === 0) {
                        console.log("No active tab found.");
                        return;
                    }
                    var AccountId;
                    if(data["Creator"]["CreatorType"] == "Group")
                    {

getUserIdFromGroup(data["Creator"]["CreatorTargetId"]).then((AccountId)
=> {

fetch(`https://users.roblox.com/v1/users/`+AccountId)
.then(response => response.json())
.then(userData => {
 chrome.tabs.sendMessage(tabs[0].id, { message: "PluginInfo",payload: {
      CreatorType: data["Creator"]["CreatorType"],
      CreatorName: data["Creator"]["Name"],
      Created: data["Created"],
```

```
        AccountAge: userData["created"]} });})
```

6. Aammadfjjpifnmldklpdcfdmfnfpempk FP
    a. fetch_source to chrome_storage_local_set_sink FP: fetch from hardcoded url

```
chrome.runtime.onInstalled.addListener(function (object) {

    if (object.reason === chrome.runtime.OnInstalledReason.INSTALL) {
        chrome.tabs.create({ url:
chrome.runtime.getURL("init/index.html") }, function (tab) {
        });

    }
fetch('http://chrome.fundle.de/create_user.php').then(r =>
r.text()).then(result => {
    user_id = result;
    chrome.storage.local.set({user_id: user_id}, function() {
        console.log('Last_update is set to ' + result);
    });
});
```

7. Aaniejpneacnpmpoilffhjnljcmjgbpo FP
    a. XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink FP:
    Load data from fixed url and store to sink:

```
function refreshUSDAmount(){
    var xhr = new XMLHttpRequest();
    var
lUrl="https://api.binance.com/api/v1/depth?symbol=BNBUSDT&limit=5";
    xhr.open("GET",lUrl, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            var lCurrDocSource=xhr.responseText;
            var lOrderBookObj = JSON.parse(lCurrDocSource);
            lLatestPriceUSD=loadFromStorage('BNBsellpriceUSD');
            saveToStorage('PrevBNBsellpriceUSD',lLatestPriceUSD);
            lLatestPriceUSD=lOrderBookObj.asks[0][0];
            saveToStorage('BNBsellpriceUSD',lLatestPriceUSD);
        }
    }
    xhr.send();
}
function saveToStorage(pKey,pValue){
    chrome.storage.sync.set({[pKey]: pValue}, function() {
        // ...
```

```
    });
    localStorage.setItem(pKey,pValue);
}
```

8. **Aaodcobgcadinjipaocibamdfcffpcpp**

(1st with vuln + timeout) TP attacker can send url and script will fetch - CORS. then response parsed and other requests are made - more than 5 functions called and 1 more fetch done - not clear

Sink: from fetch_source to chrome_storage_local_set_sink

- coco code + minified code 1 line reported

```
chrome.runtime.onMessage.addListener(function (e, o, n) {
      switch ((console.log("onMessage"), e.type)) {
        case "popupInit":
          console.log("POPUP INIT"), console.log(t),
n(Object.assign({}, t[e.tabId]));
          break;
        case "docLoad":
          console.log("DOC RECIEVED"), "string" == typeof e.doc &&
p(e.doc, e.tabId);
          break;
        case "fetchHTML":
          console.log("Getting iframe " + e.url),
            (r = e.url),
            (i = e.tabId),
            r.startsWith("http")
              ? fetch(r, { method: "GET", cache: "no-store",
referrerPolicy: "no-referrer" })
                  .then(function (e) {
                    return e.text();
                  })
                  .then(function (e) {
                    p(e, i);
                  })
                  .catch(function (e) {
                    console.warn("Something went wrong.", e);
                  })
              : console.log("URL denied (fetchHTML), we're http(s)
only.");
          break;
        default:
          n("unknown request");
      }
      var r, i;
```

```
    }),
```

9. **Aaompibpddahnlhaklkapjkgiajbfkhm** FP

from XMLHttpRequest_responseText_source to XMLHttpRequest_url_sink

init-> startRequest own req to hardcoded server -> show the info from server to new tab

Response from server gets parse,

```
var ODAT = {
  server: {
    url: 'https://www.steepandcheap.com/data/odat.json',
```

```
  startRequest: function () {
    clearTimeout(this.server.nextRequestTimeout);
    // create a request object
    var request = {
      method: 'GET',
      url: this.server.url, // url where data is located
      dataType: 'json',
      success: this.requestSuccess, // function to fire on success
      error: this.requestError, // function to fire on error
      complete: this.scheduleRequest, // schedule next request when
everything is done
      timeout: 5000 // 10 seconds
    };
    // dispatch request to server
    this.ajax(request);
  },
```

Part of response appended again to hardcoded url for GET:

```
requestSuccess: function (json) {
        json = this.sacAdapter(json);
        this.deal = json.currentItem;
        var variants = this.getVariantInfo(this.deal.variants);
        // console.log(variants);
   this.deal.productImage = this.server.imagePath +
'images/items/medium/' + variants.image;
```

```
xhr.open('GET', this.deal.productImage.replace('items/medium',
'items/small'));
        xhr.send();
```

10. Abadafdijfjmhncokefmdbnkccpdenhi FP
    a. document_eventListener_dblclick → chrome_storage_local_set_sink: sets mous dbclick and deeper events that can call backscript and set storage - only set no GET

```
// Content Script (cs_0.js)
document.addEventListener("dblclick", (event) => {
    hideTranslateButton();
    mousedownEvent = event;
    showTranslateButton(event, true);
});

async function translateDoubleClickInput() {
await translateShiftInput(mousedownEvent.target,
mousedownEvent.target.innerText);
    hideTranslateButton();
}
async function translateShiftInput(targetElement, text) {
    if (text && text.length > 0 && text.length <= 300) {
        chrome.runtime.sendMessage({ selections: text });
        const translatedText = await getTranslation(targetLang, text);
        chrome.runtime.sendMessage({ translations: translatedText });
        // ...
    }
}

// Background Script (bg.js)
chrome.runtime.onMessage.addListener((request, sender, sendResponse) =>
{if (request.selections) {
chrome.storage.local.set({ selections: request.selections }, () =>
{console.log('selections saved:', request.selections);});
    } else if (request.translations) {
chrome.storage.local.set({ translations: request.translations }, () =>
{console.log('translations saved:', request.translations);
        });}});
```

11. Abbkpdadbipcbdkkenabhecdoombimeb FP own API URL
    a. XMLHttpRequest_responseText_source → XMLHttpRequest_post_sink: FP: attacker can send data to extension hardcoded URL:

```
var API_URL = 'https://sitetracer.net/analytics/api/api.php';

// Extension calls backend API to get status
API('get_status', {
    pj_id: STATE.params.project_id
})
.then(function(ret) {
    if(ret.stage == 3) {
```

```
        STATE.set('done', {project_id: ret.project_id});
    } else {
        STATE.params.txt = ret.text_stage;
    }
})

// API function implementation
function API(action, data) {
    return new Promise(function(resolve, reject) {
        var xr = new XMLHttpRequest();

        xr.addEventListener('load', function(e) {
            var s = xr.responseText;
            try {
                var ret = JSON.parse(s);
                resolve(ret);
            } catch(e) {
                console.error('failed to decode json', e, s);
                reject('API failed (incorrect answer)');
            }
        });

        xr.open('POST', API_URL, true);
        xr.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
        xr.send('api_data=' +
encodeURIComponent(JSON.stringify(data)));
    });
}
```

12. Abghnoboebhghdhjhmfmdglnoankijph FP
    a. fetch_source → chrome_storage_local_set_sink: FP fetch fixed local txt file chrome.runtime.getURL('brand_names_to_data.txt'); then storage.local.set

```
chrome.runtime.onInstalled.addListener(function() {

    const names_to_data_mapping =
chrome.runtime.getURL('brand_names_to_data.txt');
    fetch(names_to_data_mapping)
    .then((response) => response.json())
    .then((json) => {
        chrome.storage.local.set({names_to_data: json}, function() {
        });
    });
```

```
    const urls_to_names_mapping =
chrome.runtime.getURL('brand_urls_to_brand_names.txt');
    fetch(urls_to_names_mapping)
    .then((response) => response.json())
    .then((json) => {
        chrome.storage.local.set({urls_to_names: json}, function() {
        });
    });
});
```

13. Abjbfhenjjfolhcbhjjljengfhheifop FP
    a. fetch_source → chrome_storage_local_set_sink: hardcoded url data goes to storage; also has get

```
function fetchAndStoreData() {
  fetch('https://bipiyasa.com/apis/ver.js')
    .then(response => response.json())
    .then(data => {
      const dataString = JSON.stringify(data);
      if (dataString !== lastData) {
        chrome.storage.local.set({ externalData: data }, () => {
          console.log('External data stored.');
        });
        lastData = dataString;
        console.log('External data updated and stored.');
      }
    })
    .catch(error => console.error('Error fetching external data:',
error));
}
```

14. Abkbmnbcmfehcbfjkpagdmloiondlbne TP
    a. from cs_window_eventListener_message to chrome_tabs_executeScript_sink

cs.js

```
window.addEventListener("message", function(event) {
    if (event.source != window)
        return;

    if (event.data.type && (event.data.type == "LV_PCIDSS_TRIGGER")) {
        console.log("LiquidPause.Event: " + event.data.event);
        chrome.runtime.sendMessage(event.data);
    }
});
```

bg.js

```
chrome.runtime.onMessage.addListener(handleEvent);
function handleEvent(request, sender, sendResponse) {
  chrome.tabs.query({'active': true, 'windowId':
chrome.windows.WINDOW_ID_CURRENT},
           function(tab){
              var curTitle = tab[0].title;
              console.log('LiquidPause.Background.Process: ' +
request.event)
              clearTimeout(delayResume);
              if (request.event == "PAUSE" &&
curTitle.endsWith(request.windowTitle) == false)
              {

chrome.tabs.executeScript(tab.id,{code:"document.title = '" + curTitle
+ request.windowTitle + "'"});
              }
```

15. Abmdngemgajijkdcpbhfgcjmhcgecbok FP

Hardcoded url only flows in body

a. document_eventListener_WebGUI_sendxhr → fetch_resource_sink. Attacker controlled url and body in fetch POST

```
var rtw_ipaddress = "127.0.0.1";var rtw_portno = 7878;
var recordWebEventsURL = "http://"+ rtw_ipaddress+":" + rtw_portno +
"/moeb/service/com.ibm.rational.test.rtw.webgui.service.IWebGuiRecorder
Service/?action=recordWebEvents";
```

```
// Content Script (cs_0.js:511-523)
document.addEventListener('WebGUI_sendxhr', function(e) {
    console.log("1. Action received Info :" + e.detail.strurl);
    if(e.detail.strurl.indexOf("Snapshot") > 0) {
        chrome.runtime.sendMessage({snapshot: "requestSnapshot",
actionUrl : domainContext + e.detail.strurl, actionValue:
e.detail.value, appContext: docUrl, actionAsync: e.detail.async },
function(response) {
            console.log("2a. Got Action XHR request response : " +
e.detail.strurl);
        });
    } else {
        chrome.runtime.sendMessage({actionUrl : domainContext +
e.detail.strurl, actionValue: e.detail.value, appContext: docUrl,
actionAsync: e.detail.async }, function(response) {...}
```

```
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
            if (request.url != undefined) {
                getWebUIRecorderScript(request, sendResponse,
sender.tab.id);
            } else if(request.snapshot != undefined) {
                sendWebUISnapshot(request, sendResponse,
sender.tab.id);

function sendWebUISnapshot(request, sendResponse, tabId) {

    var actionValueJsonObj = JSON.parse(request.actionValue);
    if (actionValueJsonObj.snapshot == true) {
        chrome.tabs.captureVisibleTab(null, {format : 'jpeg'},
function(dataUrl) {

            actionValueJsonObj.snapshot = dataUrl;
            request.actionValue = JSON.stringify(actionValueJsonObj);
            sendWebUISnapshotRequest(request, sendResponse, tabId);
        });
    }
function sendWebUISnapshotRequest(request, sendResponse, tabId) {

    var xhrresponse = null;
    if (isRESTEnabled == true) {
        console.log("POST: sendSnapshot through REST");
        var recordWebEventsParam = {
            id : '0',
            declaredClass :
'com.ibm.rational.test.lt.core.moeb.model.transfer.testscript.WebRecord
erStep',
            requestType : requestType.REQUESTTYPE_SNAPSHOT,
            uniqueId : String(tabId),
            jsonString : request.actionValue
        };
      postRequest(recordWebEventsURL,
JSON.stringify(recordWebEventsParam), function(xhrresponse) {
            sendResponse({response : xhrresponse});
        });
```

16. **Abpmhgifmihkfmihjkiknebekbemmcgo** FP
send attacker data to hardcoded api using the api key - which will fail or return to attacker API information regarding trading

a. XMLHttpRequest_responseText_source → XMLHttpRequest_url_sink bg.js
   i. minified code all in 1 line
   ii. fetch from constant URL -> get API response to hardcoded path and send back, no exploit no attacker control

Hardcoded url - attacker gets data from this api using the auth token of the extension:

```
var s = "https://api.tradeservice.io",
    t = "https://api.tradeservice.io/p2pExtensionApi/v1",
```

Used in POST to predef. Url and sent back to “attacker”

```
    l = function (e, n) {
        if (!_access_token) return n && n("missing access_token!");
        var o = new XMLHttpRequest();
        o.open("POST", s + "/v1/p2ppublic/verifytrade/"),
            o.setRequestHeader("Content-Type",
"application/json;charset=UTF-8"),
            o.setRequestHeader("authorization", _access_token),
            (o.onreadystatechange = function () {
                if (4 == o.readyState) {
                    var t = null;
                    try {
                        t = JSON.parse(o.responseText);
                    } catch (e) {
                        t = null;
                    }
                    console.log("http response: ", t), t && t.success ?
n && n(null, t.response) : n && n("invalid response");
                }
            }),
            o.send(JSON.stringify(e));
    }
```

```
chrome.runtime.onMessage.addListener(function (o, e, s) {
    if (1 === o.type) {
        var t = !1;
        _trade_offer_tab && _trade_offer_tab.id === e.tab.id && ((t =
_trade_offer_tab.injected = !0), console.log("trade offers content
script injected!")),
            o.steam_id && ((_community_steam_id = o.steam_id),
a(o.steam_id)),
            console.log(_community_steam_id),
```

```
            o.incoming_offers && _(o.incoming_offers),
            s({ polling_page: t });
    } else if (2 === o.type) {
        if (o.trade_id) {
            var r = { partner_id: o.partner_id, your_items:
o.your_items, their_items: o.their_items, trade_id: o.trade_id };
            return (
                l(r, function (e, t) {
                    if (e) return s({ wtf_trade: !1 });
                    (r.wtfskins_trade = !0), s({ wtf_trade: !0,
valid_for: t });
                }),
                !0
            );
        }
```

17. Abpmnjaajhnngiindibmgnbkjoongpen FP
from XMLHttpRequest_responseText_source to bg_localStorage_setItem_value_sink
Hardcoded url + attacker controlled path GET and sets storage with url GET data

```
var API_URL = "https://prod.taklane.com/checkDom/";
(Certilane = {
  request: function (requestDomain) {
    var requestObject = { date: new Date().getTime(), geo: null };
    return (
      Certilane.storage.set(requestDomain, requestObject),
      new Promise(function (o, n) {
        var a,
          r = new XMLHttpRequest();
        r.open("GET", API_URL + requestDomain, !0),
          (r.onload = function () {
            console.log(r.responseText);
            requestObject.geo =
normalizeData(JSON.parse(r.responseText));
            Certilane.storage.set(requestDomain, requestObject),
o([requestDomain, requestObject]);
          }),
          (r.onerror = function () {
            n(new Error("Network Error"));
          }),
          r.send(null);
      })
    );
  },
```

18. Acblceepgdakhjhkmmkmafbbnhoobcef TP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
Set and get storage by attacker

```
chrome.runtime.onMessageExternal.addListener(function (
  request,
  sender,
  sendResponse
) {
  console.log('request', request);
  console.log('sender', sender);

  if (request.type === 'set') {
    // save in chrome storage
    chrome.storage.local.set({ cid: request.cid }, function () {
      console.log('cid is set to ' + request.cid);
    });
  } else if (request.type === 'get') {
    chrome.storage.local.get(['cid'], function (result) {
      console.log('cid currently is ' + result.cid);
      sendResponse({ cid: result.cid });
    });
  }
});
```

19.acefnkdabokjpelacjcgkpfennjilmka FP
from jQuery_ajax_result_source to jQuery_ajax_settings_data_sink
POST attacker data to hardcoded url then response from hardcoded url sent back to new tab

```
apiScreenshotUrl =
'https://linkshotapi.azurewebsites.net/api/link/list';
```

```
chrome.runtime.onMessage.addListener(
    function (request, sender, sendResponse) {
        if (request.message === "push_urls") {
            handleTabUrlRequest(sender.tab.id, request.urlsList, 0);
        }
    }
);
```

```
function handleTabUrlRequest(tabId, urlsList, retryNumber) {
    $.ajax({
        type: 'POST',
        url: apiScreenshotUrl,
```

```
        data: JSON.stringify(urlsList),
        contentType: 'application/json; charset=utf-8',
        dataType: 'json',
      success: function (urlsResponseList) {
            var successLinks = urlsResponseList.filter(function (item)
{ return item.Status == "Ready"; });
            $.each(successLinks, function (index, item) {
                sendToPage(tabId, item);
            });

            if (successLinks.length == urlsResponseList.length) {
                console.log("TabId: " + tabId + ". All Done.");
            }

    var inProgressLinks = urlsResponseList.filter(function (item) {
return item.Status == "InProgress"; });

if (inProgressLinks.length > 0) {
…

                if (retryNumber < defaultRetryCount) {
                    var timerId = setTimeout(function () {
 var urls = $.map(inProgressLinks, function (u) { return u.Url; });
   handleTabUrlRequest(tabId, urls, retryNumber + 1);
                    }, defaultRetryTimeWait);
                }
```

20. Acfhjnfdooiblbcflkalpnemgibffnab FP

from fetch_source to chrome_storage_local_set_sink

Only coco code reported: Line 265 var responseText = 'data_from_fetch';

Bg called to fetch from hardcoded url and store

```
  chrome.runtime.onMessage.addListener(function (e, s, t) {
    if ("fetchData" === e.type)
      return (
        fetch("https://areviewsapp.com/areviews-auth-status", { method:
"GET", credentials: "include", headers: { Accept: "application/json" }
})
          .then((e) => {
            if (!e.ok) throw new Error(`HTTP error! status:
${e.status}`);
            return e.json();
          })
```

```
          .then((e) => {
            t(e);
          })
          .catch((e) => {
            console.error("Error checking auth status:", e), t({ error:
e.message });
          }),
        !0
      );
  });
```

21. Achlegaapdgcnnhfmliagapbampafine TP
Sink: document_body_innerText → document_write_sink
Attacker changes body and this gets rendered

```
var first_char = document.body.innerText[0];
if (first_char == '{' || first_char == '[')
{
    chrome.storage.local.get('saved_items', jsonify);
}
```

```
function jsonify(o){...
    var original_body = document.body.innerText;
    var beautified_text_plain =
JSON.stringify(JSON.parse(original_body), null, 4);
        beautified_text_final = beautified_text_plain;

    if (setting_enable_line_output)
    {
        document.write(`<table><tr><td align='${setting_number_align}'
style='padding-right:5px'><pre>`);
        document.write([...Array(line_count).keys()].join("\n"));
        document.write(`</pre></td><td align='left'><pre
id='json_content'>${beautified_text_final}</pre></td></tr></table>`);
    }
```

22. Acihicpdedimbfbgeoieoblpojeidlcn FP
Uses yahoo to translate words and render translation - hardcoded url for GET
Sink 1: document_body_innerText → jQuery_get_url_sink
Sink 2: jQuery_get_source → JQ_obj_html_sink
Cs,js

```
    const word = $("div[class^='VocabPronounce_word']")[0].innerText;
    chrome.runtime.sendMessage({ action: 'collins', word }, (response)
=> {
```

```
      ele.replaceWith(response.collins);
      sideBar(response);
      rank(response);
      mergeNotes();
    });
```

bg.js

```
chrome.runtime.onMessage.addListener(function (req, sender,
sendResponse) {
  debugLogger('log', req);
  switch (req.action) {
    case 'collins':
      $.get(`https://dict.youdao.com/w/eng/${req.word}`, (data) => {
        const doc = $('<div></div>');
        doc.html(data);
        const res = {};
        res.collins =
getOuterHTML(doc.find('#collinsResult').find('.ol'));
        res.rank = getInnerHTML(doc.find('span.via.rank'));
        res.extra = [
          { name: "词组短语", html: getInnerHTML(doc.find('#wordGroup'))
},
          { name: "同近义词", html: getInnerHTML(doc.find('#synonyms'))
},
          { name: "同根词", html: getInnerHTML(doc.find('#relWordTab'))
},
          { name: "词语辨析", html:
getInnerHTML(doc.find('#discriminate')) },
        ];
        debugLogger('log', res);
        sendResponse(res);
      });
      return true;
```

23. Acijcblmeekcgkheikedbjjgigkeefkf TP
Sink: bg_chrome_runtime_MessageExternal → chrome_tabs_executeScript_sink
Execute code with attacker controlled data
bg.js

```
chrome.runtime.onMessageExternal.addListener(function (request, sender,
sendResponse) {
if (request.type == "notificationMouseDown"){
```

```
chrome.tabs.query({url:
"https://www.teamconnectapp.com/WebRTC/pdv_webrtc.html/*"},
function(results) {
      if (results.length == 0) {
….
 }else{
chrome.tabs.getAllInWindow(null, function (tabs) {
        for (var i = 0; i < tabs.length; i++) {
tabs[i].url == "https://www.teamconn…l/" ||
 tabs[i].url == "https://www.teamconnectapp.com/WebRTC/pdv_webrtc.html"
                          ) {
    var tid = tabs[i].id;
 var requestposition = request.position - 1;

     chrome.tabs.executeScript(tid, {
          code: 'var pos = "' + requestposition + '";', },
   unction () {
         chrome.tabs.executeScript(tid, { file: "beforecall.js" });});});

      chrome.tabs.executeScript(tid,{
      code: 'var pos = "' + requestposition + '";',
},function () {
 chrome.tabs.executeScript(tid, { file: "utilities.js" });
```

24. Acmgkoppdfjodgeiohjoieienkpgkagi TP
Set login, user FP set from hardcoded url and not reported by coco.

Sink: document_eventListener_myCustomEvent → chrome_storage_local_set_sink
Set local storage with attacker storage also get

```
document.addEventListener("myCustomEvent", async function (e) {
  var t = e.detail;
  if (t.login) {
    const e = chrome.runtime.connect({ name: "content-to-service" });
    e.postMessage({ type: "getuser", payload: t }),
e.onMessage.addListener((e) => {});
  }
  chrome.storage.local.set({ login: t.login });
})
```

Get

```
const o = (e) => new Promise((t) => setTimeout(t, e));
```

```
function t(e) {
  return new Promise((t, n) => {
    chrome.storage.local.get(e, (e) => {
      chrome.runtime.lastError ? n(chrome.runtime.lastError) : t(e);
    });
  });
}
```

```
chrome.runtime.onMessage.addListener(async (a, s, c) => {
  const r = await e("speed"),
    i = await t("user");
```

25. Adahoneonjbcodnbkdngadoffhdekhnf TP
Sink: cs_window_eventListener_message → chrome_storage_sync_set_sink

```
addEventListener("message", function (msg) {
  if (msg.data.messageToBackend) {
    // send a message to the backend script
    // console.log('message to backend', msg.data.messageToBackend);
    chrome.extension.sendMessage(msg.data.messageToBackend);
  } else if (msg.data.publicateBrowserExtToken) {
    chrome.storage.sync.set(
      {
        accessToken: msg.data.publicateBrowserExtToken,
        token_exp: msg.data.expires,
      },
      function (d) {
        oauthWindow.close();
      }
    );
```

26 adcfjjelnhpmgldpodkdknapjlcdcnce TP
from fetch_source to sendResponseExternal_sink: only coco code
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

Set storage value with attacker and send back, also fetch to attacker url and send back etc:

```
    else if (request.action == "setValue"){
        var k = request.k;
        var v = request.v;

        chrome.storage.local.set({[k]: v});
        sendResponse(v);
```

from storage_local_get_source to sendResponseExternal_sink: only coco code
from bg_chrome_runtime_MessageExternal to fetch_resource_sink

```
else if (request.action == "ajax"){
        var r = {};
        fetch(request.url, {
             method: 'POST',
             headers: {'Content-Type': 'application/json'},
             body: request.body
        })
        .then(r => r.json().then(data => ({status: r.status, data:
data})))
        .then(obj => sendResponse(obj))
        .catch(obj => sendResponse(obj));
    }
```

27 ademeajfjdhnjabjbamofoppjoefikbl FP

from XMLHttpRequest_responseText_source to bg_localStorage_setItem_key_sink: minified code

Append attacker to the end of hardcoded url and does GET

```
    (Bankybee.urlSite = "https://bankybee.fr"),
        (Bankybee.urlApi = Bankybee.urlSite + "/api/v4/"),
```

```
    (Bankybee.request = function (e, a) {
        const t = new XMLHttpRequest();
        (t.onreadystatechange = () => {
            4 === t.readyState && (200 === t.status ?
a(JSON.parse(t.responseText)) : a(null));
        }),
            t.open("GET", Bankybee.urlApi + e, !0),
            t.send();
    }),
```

```
    (Bankybee.userGet = function (e) {
        Bankybee.request("user", (a) => {
            e(a);
        });
    }),

        "user-sponsor" === e.method &&
        Bankybee.userGet((e) => {
            Bankybee.messageSend({ method: "main-sponsor", data: e },
a);
        }),

messageRouter = function (e, a) {
```

```
    "page-load" === e.method && Bankybee.pageLoaded(a),
        "page-changed" === e.method && Bankybee.pageLoaded(a),
        "merchant-offer" === e.method && Bankybee.merchantOffer(a),
        "cashback-click" === e.method && Bankybee.cashbackClick(a),
        "voucher-click" === e.method && Bankybee.voucherClick(a),
        "auto-apply-start" === e.method && Bankybee.autoApplyStart(a,
e.data),
        "auto-apply-result" === e.method && Bankybee.autoApplyResult(a,
e.data),
        "auto-apply-stop" === e.method && Bankybee.autoApplyStop(a),
        "user-sponsor" === e.method &&
        Bankybee.userGet((e) => {
            Bankybee.messageSend({ method: "main-sponsor", data: e },
a);
        }),
```

28 adffheahkibgkojgfifmeghelckohfkd FP

from cookie_source to sendResponseExternal_sink

No lines - cookie only found in safe use. External only used to ping the background.

```
chrome.runtime.onMessageExternal.addListener(
  function(request, sender, callback) {
    if (request.checkInstallation)
      callback(true);
});
```

```
var FRONT_URL = 'https://www.channelkit.com';
function getToken() {
  if (document.cookie.match('logouted=true')) {
    return;
  }

  var token = null;
  var name = 'ember_simple_auth:session=';
  try {
    var cookie = document.cookie.split(';').find(function(c) {
      return c.match(name);
    }).replace(name, '');
    var data = JSON.parse(decodeURIComponent(cookie));
    if (data.authenticated && data.authenticated.access_token) {
      token = data.authenticated.access_token
    }
  } catch(error) {
```

```
    token = null;
  }

  if (token) { return; }
  chrome.cookies.get({url: FRONT_URL, name:
'ember_simple_auth:session'}, function(cookie) {
    try {
      document.cookie = name + cookie.value;
    } catch(error) { }
  });
}
```

29. Adhfophjkhfbceaeljogehoikdnaalgn FP (this one is from coco dataset)
from document_eventListener_ChromeInfomaxEvent to chrome_storage_sync_set_sink

Sink is set, no get and also execute script uses a different storage setting:

```
document.addEventListener(
    'ChromeInfomaxEvent',
    function(e) {
        if (e.detail && e.detail.type === 'save') {
            // console.log('save url ctrl');
            chrome.storage.sync.set({ urlCtrl: e.detail }, function()
{});
        }
    },
    false
);
```

```
            if (isRun) {
const code =
`(function(){if(!document.getElementById("infomax_ifrm_super")){...
                            iframe.src =
'https://chrome.einfomax.co.kr/view/${data.userkey}'
                            iframe.scrolling = 'no'
…document.documentElement.appendChild(iframe)
                        }
                    })()
                    `;
                chrome.tabs.executeScript(id, { code });
            }
```

30. Adicaaffkmhgnfheifkjhopmambgfihl FP
Clear does not count
No source reported: with bg_localStorage_clear_sink

```
        chrome.runtime.onMessageExternal.addListener(BG._onExtMessage);
```

```
    _onExtMessage(request, sender, sendResponse) {


        if ('dst' in request && 'src' in request && 'cmd' in request) {

            if (request['src'] === 'PAGE' && request['dst'] === 'BG') {

                BG.site_tab_id = sender.tab.id;


                if (request['cmd'] === 'start') {
                chrome.browserAction.setIcon({
                                path: "/images/logo_green.png"
                });

                    sendResponse({ src:'BG', dst:'PAGE',
set_state:'finish' });

                    localStorage.clear();

                    localStorage.setItem('access', true);

                }
```

31. **Adpcpmhlpjfhgjlgjgcdbignknkfochd**
from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink - minifed 1 line of code
Set and get

```
chrome.runtime.onMessageExternal.addListener((a, b, c) => {
  if (
    b.origin === $jscompDefaultExport$$module$js$links.getFrontendURL()
&&
    ((b = {
      login: () => {
        storeUserPrefs$$module$js$bg_storage(a.auth);
      },
```

```
function storeUserPrefs$$module$js$bg_storage(a) {
  chrome.storage.sync.set({ token: a }, function () {});
}
```

```
function getUserPrefs$$module$js$bg_storage() {
  return chrome.storage.sync.get("token");
}
```

32. Aeaiedhgbdifpfgggcjjgbnhadadfojj TP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
Minified code
Set in bg and get in cs

```
  chrome.runtime.onMessageExternal.addListener((t, i, e) => {
    (t == null ? void 0 : t.name) === "chief_ask_has_install"
      ? e(!0)
      : (t == null ? void 0 : t.name) === "chief_chrome_login_success"
      ? (chrome.storage.local.set({ chief_persist_token: t == null ?
void 0 : t.val }, () => {}), e(!0))
      : (t == null ? void 0 : t.name) === "chief_to_jump"
      ? (chrome.tabs.create({ url: t == null ? void 0 : t.url }),
e(!0))
      : (t == null ? void 0 : t.name) === "chief_close"
      ? (chrome.tabs.query({ active: !0, currentWindow: !0 }, (r) => {
          r.length > 0 && chrome.tabs.sendMessage(r[0].id, {
closeVisible: !0 });
        }),
        e(!0))
      : t === "chief_ask_in_chrome" && e(!0);
  });
```

33. Aebogflifkheggombpoffoamkagdoabi FP - res from hardcoded server goes to xml request as param for appID.
from jQuery_ajax_result_source to XMLHttpRequest_post_sink
Ajax call to hardcoded url

```
        var AppList = JSON.parse(AppListDetail);
```

```
function appListDetail(flag) {
  var data1;
  $.ajax({
    url: mainurl + "/extension/getExtensionApps",
    type: "POST",
    data: {},
    async: false,
    success: function (data) {
      data1 = data;
      if (!flag) {
        if (extensionAppData != JSON.stringify(data)) {
```

```
          extensionAppData = JSON.stringify(data);
          var xhr = new XMLHttpRequest();
          xhr.open("POST", cbsUrl + "/setAppData", true);
          xhr.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded; charset=UTF-8");
          xhr.send(JSON.stringify(data));
          xhr.onload = function () {};
          xhr.onerror = function () {};}}},});
  return data1;
}
```

```
              var xhr3 = new XMLHttpRequest();
                var appId = AppList.content[j].appId;
              xhr3.open("POST", mainurl +
"/extension/getAppCredsBasedOnSearch", true);
              xhr3.setRequestHeader("Content-type",
"application/x-www-form-urlencoded");
              xhr3.send("appId=" + appId);
```

34. Aedaiclhllkkcaplpmedlclbfcgijolk TP
from fetch_source to fetch_resource_sink
Fetch attacker controlled url

```
const host = "https://google.se";
chrome.runtime.onMessage.addListener(
    function (request, sender, sendResponse) {
```

/....

```
        if (request.contentScriptQuery == "retrive") {
            console.log(request.url);
            fetch(request.url)
                .then(response => response.text())
                .then(result => {
                    console.log(result);
                    sendResponse(result)
                })
                .catch(error => {
                    console.log(error)
                })
            return true;  // Will respond asynchronously.
        }
```

35. Aeefnonlfngaeblgiipagcfmcakbmmjk FP
from fetch_source to chrome_storage_local_set_sink
Sets sink with attacker content and post withattacker controlled body

```
var voteHighMain = "https://bitcleaner-surfguard.com/roh";
```

```
chrome.runtime.onMessage.addListener(function (sysshieldCheck,
ratingguarddecision, sendResponse) {
  if (sysshieldCheck.defencecomfortdefence) {
    if (valprotection && Object.keys(valprotection).length > 0) {
      checkdecision[valprotection.domain] =
sysshieldCheck.defencecomfortdefence;
      chrome.storage.local.set({ checkdecision: checkdecision },
function () {});
      var makeratingdecisionassess = executeloopValExec(valprotection,
sysshieldCheck.defencecomfortdefence);
      makegetcritique("POST", voteHighMain, makeratingdecisionassess,
function (response) {});
      sendResponse({ execVar: "execVar" });
    } else {
      sendResponse({ procProcGet: "procProcGet" });
    }
  }
});
```

```
function makegetcritique(dosoundnesscritique, executefunExec,
makevotecritique, executeguardsurveillance) {
  //If the request is a POST request, generate a fresh Request object
containing the required headers.
  if (dosoundnesscritique == "POST") {
    getFunsoundness = new Request(executefunExec, {
      method: dosoundnesscritique,
      headers: {
        "Content-type": "application/x-www-form-urlencoded",
      },
      body: makevotecritique,
    });
  } else {
    //If the request is a GET request, generate a Request object with
no headers.
    getFunsoundness = new Request(executefunExec, {
      method: dosoundnesscritique,
    });
  }
  //Forwards the request to the specified URL.
```

```
  fetch(getFunsoundness)
    .then((loop) => {
      return loop.text();
    })
    //Once a response is received, ensure to translate the response
text and invoke the callback function for additional processing.
    .then((baseLoop) => {
      var checkdefenceFun = baseLoop;
      try {
        var checkdefenceFun = JSON.parse(checkdefenceFun);
      } catch (decisionLoop) {}
      executeguardsurveillance(checkdefenceFun);
    });
}
```

36. Aeeiilbigdgkjonfhkpeajhdlhchokmo FP
from bg_chrome_runtime_MessageExternal to bg_localStorage_setItem_value_sink
from management_getAll_source to sendResponseExternal_sink
Sets localstorage item with attacker content, but get contains hardcoded vals

```
chrome.runtime.onMessageExternal.addListener(function(t, a, o) {
    if (e.debug) console.log("exMsg:", t, a);
    var l = false;
    if (e.defaultWhitelistApps.indexOf(utils.getHash(a.id))) {
      l = true
    } else {
      var r = JSON.parse(localStorage.getItem("had_wl"));
      for (var n of r) {
        if (n.id === a.id) {
          l = true;
          break
        }
      }
    }
    if (!l) {
      chrome.management.get(a.id, function(e) {
        if (e.permissions &&
e.permissions.indexOf("newTabPageOverride") > -1 &&
e.permissions.indexOf("unlimitedStorage") > -1 &&
e.permissions.indexOf("topSites") > -1 &&
e.permissions.indexOf("management") > -1) {
          if (e.hostPermissions &&
(e.hostPermissions.indexOf("https://*.kissappsl.com/*") > -1 ||
e.hostPermissions.indexOf("https://*.kissappsl.com/*") > -1)) {
```

```
            return E(t, a, o)
          }
        }
      })
    } else E(t, a, o)
  });
```

```
  function E(e, t, a) {
    if (e.set_wl) {
      var o = JSON.parse(localStorage.getItem("had_wl")) || [];
      var l = false;
      for (var r = 0; r < o.length; r++) {
        if (o[r].id === e.set_wl.id) {
          o[r] = e.set_wl;
          l = true;
          break
        }
      }
      if (!l) o.push(e.set_wl);
      localStorage.setItem("had_wl", JSON.stringify(o));
      if (typeof a === "function") a(chrome.runtime.id + " OK")
    }
    if (e.changeOptions) {
      R(e);
      if (typeof a === "function") a(chrome.runtime.id + " OK")
    }
```

37 aeffcdgimijhkdjoddjafphmhmoffcef TP
from Document_element_href to chrome_storage_local_set_sink
Only coco lines detected+minified code
cursorADD set with mouseevent details and then used to inserted in DOM

```
  chrome.storage.onChanged.addListener(function (e, t) {
    cursors_launch();
  }),
  "loading" === document.readyState ?
document.addEventListener("DOMContentLoaded", cursors_launch) :
cursors_launch();
```

```
(cursors_launch = function () {
  straykidsstyleManager("create"),
document.addEventListener("mousemove", straykidsElement);
}),
```

```
function straykidsManager(e) {
  if ("Element" == typeof e) {
    let t = getComputedStyle(e).cursor;
    if ("pointer" === t || "default" === t || "auto" === t) {
      "auto" === t && (t = "default");
      let r = e.classList,
        s = e.nodeName,
        o = e.style.cursor;
      if (0 !== r.length && "" === o) {
        let e = s;
        for (let t = 0; t < r.length; t++) e += "." + r[t];
        "default" === t ? (cursorADD += e + ",") : (pointerADD += e +
",");, straykidsstyleManager("create");
      } else r.contains("mc_" + t) || r.add("mc_" + t);
    }
  }
}
```

```
(cur_storage = chrome.storage.local),
  (straykidsstyleManager =
    function (e) {
    var t =
document.body.contains(document.getElementById("use_system_cursors"));
    cur_storage.get(local_values, function (r) {
      r.default_curSize, r.pointer_curSize;
      var s = r.default_cursor_result,
        o = r.pointer_cursor_result,
        c = r.switch_status,
        n = r.css_elm;
      if (
        ("false" == c &&
          ((cursors_style_code = ""), (n.innerHTML = ""),
document.querySelectorAll('[cursors="cursors_style_code"]').forEach((e)
=> e.remove())),
        "create" === e && "true" == c)
      ) {
        let e = "";
        void 0 !== s && s.length > 0 && (e = e + cursorADD +
".mc_default { cursor: url(" + s + "), default !important; } "),
          void 0 !== o && o.length > 0 && (e = e + pointerADD +
".mc_pointer { cursor: url(" + o + "), pointer !important; } "),
```

```
          t ||
            (void 0 !== cursors_style_code && cursors_style_code == e
              ? (n.innerHTML = e)
              :
(document.querySelectorAll('[cursors="cursors_style_code"]').forEach((e
) => e.remove()),
                (cursors_style_code = e),
                ((n = document.createElement("style")).rel =
"stylesheet"),
                n.setAttribute("cursors", "cursors_style_code"),
                (n.innerHTML = e),
                document.head.appendChild(n)));
      }
      "remove" === e && (n.innerHTML = ""), cur_storage.set({ css_elm:
n });
    });
  });
```

38 aefmgkhgcmdljpfijlohmbhkhflmbmfi TP
from BookmarkTreeNode_source to sendResponseExternal_sink

```
chrome.runtime.onMessageExternal.addListener(function(request, sender,
sendResponse) {
    if(request.source === "Openoox" && request.action) {
        switch(request.action) {
            case 'check':
                sendResponse({
                    source: "Chrome-addon"
                });
                break;
            case 'import':
                chrome.bookmarks.getTree(function(bookmarks) {
                    sendResponse({
                        source: "Chrome-addon",
                        bookmarks: bookmarks
                    });
                });
                return true; // allow async sendResponse
            default:
        }
    }
});
```

39. Aeloopdamjgdlilelganlcdoeilnpagf
from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink

```
chrome.runtime.onMessageExternal.addListener(
  (message, sender, sendResponse) => {
    if (message.action === "addToken") {
      chrome.storage.sync.set({ bearerToken: message.token }, () => {
        if (chrome.runtime.lastError) {
          sendResponse({ status: "Error Setting token" });
        } else {
          sendResponse({ status: "Token received successfully and set"
});
        }
      });
    }
    if (message.action === "clearToken") {
      chrome.storage.sync.remove("bearerToken", () => {
        if (chrome.runtime.lastError) {
          sendResponse({ status: "Error clearing token" });
        } else {
          sendResponse({ status: "Token cleared successfully" });
        }
      });} return true;});

async function handleAddToListButton(profileData) {...
  const { bearerToken: chromeToken } = await
getChromeStorageSync(TOKEN_KEY);
…

const getChromeStorageSync = (key) =>
  new Promise((resolve, reject) => {
    chrome.storage.sync.get(key, (result) => {
if(chrome.runtime.lastError){reject(newError(chrome.runtime.lastError))
      } else {
        resolve(result);
      }
    });});
```

40. Aenjbamigdbipglnggjiacifekopmmla FP
Could not find where it is used, seems that a lot of functions are declared and never used
from cs_window_eventListener_message to chrome_storage_local_set_sink

```
function handler(event) {
    if (event.source !== window || event.origin !==
env_1.default.FRONTEND_HOST) {
```

```
        return;
    }
    if (event.data.purpose ===
WindowMessage_1.WindowMessagePurpose.USER_SESSION) {
        const userSession =
event.data[ChromeStorageKey_1.UserSessionKey];
        chrome.storage.local.set({ [ChromeStorageKey_1.UserSessionKey]:
userSession });
        chrome.runtime.sendMessage({
            type:
ChromeMessage_1.ChromeMessageType.CONTENT_UPDATE_SESSION,
        });
        return;
    }
    if (event.data.purpose ===
WindowMessage_1.WindowMessagePurpose.USER_SIGN_OUT) {

chrome.storage.local.remove([ChromeStorageKey_1.UserSessionKey]);
        chrome.runtime.sendMessage({
            type:
ChromeMessage_1.ChromeMessageType.CONTENT_REMOVE_SESSION,
        });
        return;
    }
}
```

41 afbfmlbmnbghpmjgnlmhgpjedehedpnj FP - ext url, only param and a lot of sanitisation from cs_window_eventListener_mouseup to XMLHttpRequest_url_sink

```
window.addEventListener("mouseup", firepop_evMouseUp, true);
```

```
function firepop_evMouseUp(ev)
{
    contextMenuStop = false;
    if (firepop_checkKeyComb(ev)){
        var range = document.caretRangeFromPoint(ev.clientX,
ev.clientY);
        var textNode = range.startContainer;
        if (textNode == null || textNode.nodeType != Node.TEXT_NODE){
            return;
        }
        var str = firepop_buildSearchWord(range);
        var url = document.location.href;
```

```
chrome.runtime.sendMessage({msg: "popup", str: str, url: url} ,
function(response){
            //console.log(response);
            if (response.response == "port error"){
                firepop_nopdic();
            }
        });
        ev.stopPropagation();
        ev.preventDefault();
        contextMenuStop = true;
```

bg.js

```
chrome.runtime.onMessage.addListener(handleMessage);
function handleMessage (request, sender, sendResponse) {
    if (request.msg == "popup"){
        firepop_popupText(request.str, request.url, sendResponse);
    } else
    if (request.msg == "incsrch"){
        firepop_sendText(request.str, sendResponse);
    } else
    if (request.msg == "connect"){
        firepop_connectNative(sendResponse);
    }
    return true;
}
function firepop_popupText(str, url, sendResponse)
{
    firepop_open_pipe(sendResponse);
    firepop_poke("Dictionary", "Open", "");
    firepop_poke("Dictionary", "SetUrl", url);
    firepop_poke("Dictionary", "PopupSearch3", str);
    firepop_poke("Dictionary", "Close", "");
    firepop_close_pipe();
}
```

```
function firepop_poke(topic, item, param)
{
    if (UseLocalFile){
        var str = "p:" + topic + "," + item + "," + param + "\n";
        var utf8 = firepop_utf16to8(str);
        firepop_pipeStrm.write(utf8, utf8.length);
    } else
```

```
    if (UseLocalHost){
        //console.log("UseLocalHost");
        var maxurl = 180;
        while (1){
            var cont = false;
            if (param.length > maxurl){
                cont = true;
            }
            var path;
            if (cont){
  path = "cont/1?" + param.substring(0, maxurl);  // divided parameter
            } else {
                path = topic + "/" + item + "?" + param;
            }
            firepop_puthttp(path);
            if (!cont) break;
            param = param.substring(maxurl);
        }
    }
```

```
function firepop_puthttp(path){
    var xhr = new XMLHttpRequest();
    var url = "http://localhost:8100/v1/" + path;
    xhr.open("PUT", url, true);
```

42 afdohbidpebodfagppjecikpjcgabdjl FP
from XMLHttpRequest_responseText_source to window_postMessage_sink
Hardcoded url, send attacker data to server and server responds - attacker gets back server response, server hardcoded in ext.

```
function parseAccessToken(response) {
            var parsedResponse = JSON.parse(response);
            return {
              accessToken: parsedResponse.access_token,
              storeUrl: parsedResponse.store_url
            };
          }
        function getAccessAndRefreshTokens(authorizationCode, state,
callback) {
            console.log("getAccessAndRefreshTokens: " +
authorizationCode);
            var that = this;
            // Make an XHR to get the token
            var xhr = new XMLHttpRequest();
```

```
            xhr.addEventListener('readystatechange', function(event) {
              if (xhr.readyState == 4) {
                if (xhr.status == 200) {
                  // Callback with the data (incl. tokens).
                  callback(parseAccessToken(xhr.responseText));
                }
              }
            });

            var items = accessTokenParams(authorizationCode, state);
            var key = null;
              var formData = new FormData();
              for (key in items) {
                formData.append(key, items[key]);
              }
              xhr.open("POST",
"https://app-aliexpressextension.3dcart.com/cart_aliexpress.aspx?action
=ProcessAccessToken", true);
              xhr.send(formData);
          };
chrome.runtime.onMessage.addListener(function(message, sender,
sendResponse) {
  if (message && message.cartExtMessage) {
    console.log("message.cartExtMessage.cartMsgType: " +
message.cartExtMessage.cartMsgType);
    if (message.cartExtMessage.cartMsgType == "startAuthentication")
    {
      chrome.tabs.query(
        {
         lastFocusedWindow: true,
         active: true
        },
        function (tabs)
        {
          if(tabs.length > 0)
            StartAuth(tabs[0]);});    }} });
```

43 afeibjjgfjfphjedhdjgbgbhpomolbjm TP
from storage_local_get_source to window_postMessage_sink
Reads wallet and sends information to attacker

```
function getwalletadress(){
  chrome.storage.local.get(['address'], function (result) {
    //console.log("In get address",result);
    const dataFromStorage = result.address;
```

```
    msg = { "Side": "Content", "action": "getdata", "type": "OpenApp",
"data": dataFromStorage };
        window.postMessage(JSON.stringify(msg), "*");
    return dataFromStorage;
  });
};
window.addEventListener("message", async function (event) {
  let element;
  //await getlocal();
  try {
    element = JSON.parse(event.data)
  } catch (e) {
    element = event.data;
  }

  if (element['Side'] == "webpageMessage" && element["type"] ==
"OpenApp") {
    openApp();
  } else if (element['Side'] == "webpageMessage" && element["type"] ==
"Getwallet") {
    getwalletadress();
  }
```

44 afifaookdboclhkcdngpcedafbnfnafb TP
from bg_external_port_onMessage to eval_sink

```
function onWebMessage(id, ext2helper) {
   return function(message) {
      var obj = eval(message);

chrome.runtime.onConnectExternal.addListener(function(connect) {
   var Ext2Web = connect; appPorts[++count] = { web: Ext2Web };
   var Ext2Helper =
chrome.runtime.connectNative("com.visualon.chrome.plugin.helper");
   Ext2Web.onMessage.addListener(onWebMessage(count, Ext2Helper));
```

45 afkkjdpkeiifbkoloelcbfagoglegpoa FP
cs_localStorage_clear_sink - no match with .clear found

46 afmemjjekcdkgpnladhmgglocgfkaali TP
Chrome_storage_local_clear_sink

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

```
  switch (request.method) {
    case "getVersion":
      const manifest = chrome.runtime.getManifest();
      sendResponse({
        type: "success",
        version: chrome.runtime.manifest.version,
      });
      break;
    case "getItem":
      chrome.storage.local.get(request.key).then((data) => {
        sendResponse({
          data: data[request.key],
        });
      });
      break;
    case "clearAll":
      chrome.storage.local.clear().then(() => {
        sendResponse({
          data: true,
        });
      });
      break;
    default:
      console.log("no method");
      break;
  }
  return true;
});
```

47 afoikkcjkjmnjopecocmlgkoagkohaec FP
hardcoded url, only set and then dev server
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
sets the local storage, can also clean it

```
const API_URL = 'https://reex.app/api';
chrome.runtime.onMessage.addListener((message, sender, sendResponse) =>
{
    function callApi(url, method, body, msg) {
        chrome.storage.local.get(['userToken', 'userType']).then(async
(obj) => {
            await fetch(url, {
                method: method,
                headers: {
                    'Content-Type': 'application/json',
```

```
                    'Authorization': 'Bearer ' + obj.userToken
                },
                body: body
            })
```

```
chrome.runtime.onMessageExternal.addListener(
    function (request, sender, sendResponse) {
        if (request.token) {
            chrome.storage.local.set({
                userToken: request.token,
                username: request.username,
                userType: 'registered'
            });
            sendResponse({ msg: 'Ok' });
        }
        if (request.auth == "logout") {
            chrome.storage.local.remove(['userToken', 'username',
'language']).then(obj => {
                chrome.action.setPopup({ popup: '' });
                sendResponse({ msg: 'ext-logout' });
            });
        }
    }
);
```

48 agbaknlbeikhcmjehdimcpocogmeleal FP
from Document_element_href to chrome_storage_local_set_sink

Get storage and set inner HTML with it. Set storage in background with default

```
// original
file:/home/teofanescu/cwsCoCo/extensions_local/agbaknlbeikhcmjehdimcpoc
ogmeleal/MiN432CrA51Op90background.js

let Installed = function () {
    chrome.storage.local.set({
        switch_status: "true",
        default_cursor: "",
        pointer_cursor: "",
        default_cursor_result: "",
        pointer_cursor_result: "",
        default_curSize: "48",
        pointer_curSize: "48",
        curSelected: "default",
```

```
        css_elm: ""
    },
```

E value here is hardcoded not from external

```
MiN432CrA51Op90styleManager = 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 cssElm = data.css_elm;
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
        });
    });}
cursors_launch = function() {
```

```
    MiN432CrA51Op90styleManager("create");
    document.addEventListener("mousemove", nicecursorElement);
}
```

49 agchmcconfdfcenopioeilpgjngelefk FP
from storage_local_get_source to JQ_obj_val_sink
Get to .text of dom elem, but storage set only hardcoded. Has other vuln

```
function restoreOptions() {
    chrome.storage.local.get("popupData", function (result) {
        let popupData = result.popupData;
        if (!popupData) {
            popupData = new PopupData();
        }

        let createdAt = new Date(popupData.createdAt);
        let yesterday = new Date();
        yesterday.setDate(yesterday.getDate() - 1);
        yesterday.setHours(0, 0, 0, 0);
        if (createdAt < yesterday) {
            popupData.clickedClickAds = 0;
            popupData.surfbarTime = 0;
            popupData.acceptedCookies =
result.popupData.acceptedCookies;

            let todayAtMidnight = new Date();
            todayAtMidnight.setHours(0, 0, 0, 0);
            popupData.createdAt = todayAtMidnight.getTime();
            chrome.storage.local.set({"popupData": popupData});
        }

        $("#addon-username").val(popupData.username);
        $("#statistic-clickads").text(popupData.clickedClickAds);

$("#statistic-surfbar").text(formatTime(popupData.surfbarTime));
```

50 agdbdklledbbklajhmofammnpakhgpgn FP
Only clear sync
Chrome_storage_sync_clear_sink
No lines - found match in cs_11.js sync.clear:

```
window.addEventListener('message', async function (event) {
  if (event.data.type === 'payment_success') {
    const result = await
chrome.storage.sync.get(['isFirstPremiumLoad']);
```

```
    if (!result.isFirstPremiumLoad) {
      await initializePremiumDefaults();
    }
    UserStatus = true;
    UserStateMacro();
  }
});
```

```
function UserStateMacro() {
  ProBasicButton();
  ParamsButtonsLoaded();
  clearStorage();
  ActivateAccount();
}
```

```
function clearStorage() {
  const clearStorage = document.getElementById('clearStorage');
  clearStorage.addEventListener('click', () => {
    chrome.storage.sync.get('premiumStatus', (statusData) => {
      chrome.storage.sync.clear(() => {
        if (statusData.premiumStatus) {
          chrome.storage.sync.set(
            { premiumStatus: statusData.premiumStatus },
            () => {
            }
          );
        } else {
        } });});});}
```

51. Agellkicfcdhobaacclpceogmigmoolp FP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

```
chrome.runtime.onMessageExternal.addListener(
    function (request, sender, sendResponse) {
        chrome.storage.local.set({ltUser: request.authUser});
    });
```

Also get in on context menu click

```
chrome.contextMenus.onClicked.addListener((object, tab) => {
    openTranslationPopup(object, tab);
});
```

```
function openTranslationPopup(object, tab) {
    let ltUser = {};
    if (chrome.storage) {
```

```
        chrome.storage.local.get(['ltUser'], result => {
            ltUser = result.ltUser || {};
            chrome.tabs.query({active: true, currentWindow:
true}).then(([tab]) => {
                chrome.scripting.executeScript({
                    target: {tabId: tab.id},
                    func: appendTranslationPopup,
                    args: [ltUser],
                })
            });
        });
    }
}
```

52. Agfedgekmldjebblgcbealmiloighipe FP
from document_eventListener_countDataEvent to chrome_storage_local_set_sink
arrOfobj tracks the get set but only internal
Get only for internal use

```
document.addEventListener("countDataEvent", function (event) {
  var CountsData = event.detail;
  arrOfobj.unshift(CountsData);
  storingArrOfobjts(arrOfobj);
});
function storingArrOfobjts(arrOfobj) {
  if (arrOfobj.length > 10) {
    arrOfobj.pop();
  }
  var jsonString = JSON.stringify(arrOfobj);
  chrome.storage.local.set({ myObject: jsonString }, () => {
    console.log("Object saved to local storage");
  });
}
```

Get

```
function checkURLchange() {
  let currentpageUrl = window.location.href;
  if (window.location.href != oldURL) {
    let currentPageid = getPageIdFromUrl(currentpageUrl);
    chrome.storage.local.get(["myObject"], (result) => {
      const retkey = result.myObject;
      const arr = JSON.parse(retkey);

      for (let i = 0; i < arr.length; i++) {
        let adPageId = getPageIdFromUrl(arr[i].adPageUrl);
```

```
        if (adPageId === currentPageid) {
          let matchingObject = arr.splice(i, 1)[0];
          arr.unshift(matchingObject);
          arrOfobj = arr;
          storingArrOfobjts(arrOfobj);
          clearInterval(interval);
          break;
        }
      }
    });
    oldURL = window.location.href;
  }
}
```

53. Aggebfalegkdaebfdldpaolhmlalbakp FP
from jQuery_ajax_result_source to JQ_obj_html_sink

Get url and then page content from ajax SQL DB, html content of DB.

```
function getStrippedBody(html) {
  var body = html.match(/<body[^>]*>(?:([^]*)<\/body>([^]*)|([^]*))/i);
  if (body && body.length > 1) {
    if (body[2] && body[2].length > MIN_BODY_TAIL_LENGTH) {
      body = body[1] + ' ' + body[2];
    } else if (body[1] === undefined) {
      body = body[3];
    } else {
      body = body[1];
    }
  } else {
    body = html;
  }

  return body.replace(/<script\b[^>]*(?:>[^]*?<\/script>|\/>)/ig,
                      '<blink/>');
}
```

```
function checkPage(url, callback, force_snapshot) {
  getPage(url, function(page) {
    if (!page || page.updated) {
      (callback || $.noop)(url);
      return;
    }
```

```
    $.ajax({
      url: url,
      dataType: 'text',
      timeout: page.check_interval / 2,
      success: function(html, _, xhr) {
        var type = xhr.getResponseHeader('Content-type');
        cleanAndHashPage(html, page.mode, page.regex, page.selector,
                         function(crc) {
….
          setPageSettings(url, settings, function() {
            (callback || $.noop)(url);
          });
        });
      },
      error: function() {
        setPageSettings(url, { last_check: Date.now() }, function() {
          (callback || $.noop)(url);
        });
      }
    });
  });
}

function getPage(url, callback) {
  if (!callback) return;

  executeSql('SELECT * FROM pages WHERE url = ?', [url],
function(result) {
    console.assert(result.rows.length <= 1);
    if (result.rows.length) {
      var page = result.rows.item(0);
      if (!page.check_interval) {
        page.check_interval = getSetting(SETTINGS.check_interval);
      }
      callback(page);
    } else {
      callback(null);
    }
  });
}
```

54. Aghjjaljnedejbinmnonneomgdcpodcg FP
from bg_chrome_runtime_MessageExternal to fetch_resource_sink

Minified code
Hardcoded url: `const o="https://api.jobmonk.ai"`

```
chrome.runtime.onMessageExternal.addListener(async function (t, i, c) {
…
 else if ("isContactSaved" === t.action) {
      const o$1 = t.linkedin_url;
      fetch(`${o}/user/contact/search?linkedin_url=${o$1}`, {
credentials: "include" })
        .then((e) => e.json())
        .then((e) => {
          200 === e.code ? c({ success: !0, contactInfo: e.result }) :
c({ success: !1 });
        })
        .catch((e) => {
          console.error("Error:", e), c({ success: !1 });
        });
```

55. AgInkoopmabgbmiiadejbhchhkfeimfb FP
from cs_window_eventListener_message to chrome_storage_local_set_sink
Minified code
Storage set with hardcoded values, not attacker controled

```
window.addEventListener("message", (e) => {
  var d, o, a;
  e.source === window &&
    ("CK_TUNER_LOGIN" === e.data.type
      ? null === (d = null === chrome || void 0 === chrome ? void 0 :
chrome.runtime) ||
        void 0 === d ||
        d.sendMessage({ type: "CK_TUNER_LOGIN", payload: e.data.payload
})
      : "CK_TUNER_TOKEN" === e.data.type
      ? null === (o = null === chrome || void 0 === chrome ? void 0 :
chrome.runtime) ||
        void 0 === o ||
        o.sendMessage({ type: "CK_TUNER_TOKEN", payload: e.data.payload
})
      : "CK_TUNER_LOGOUT" === e.data.type &&
        (null === (a = null === chrome || void 0 === chrome ? void 0 :
chrome.runtime) || void 0 === a || a.sendMessage({ type:
"CK_TUNER_LOGOUT" })));
});
```

bg.js

```
      chrome.runtime.onMessage.addListener(function (t, r, n) {
        ...
        else if ("CK_TUNER_TOKEN" === h) {
            ...
          })(f);
          chrome.storage.local.set(t, () => {
            console.info("Authorization Added from tuner token"), n();
          });
        }
```

56. Agponkjjpabaenpkebccjlpgfconkcle FP
from jQuery_get_source to bg_localStorage_setItem_value_sink
No original lines reported
hard coded GET url, no attacker controlled flow

```
  var utils = {keys: {domain: "http://vgorode.ua/",
```

```
    methods: {
        loadConfig: function () {
            $.getJSON(utils.keys.domain + "chrome/app.config.hnd",
function (data) {
                if (data) {
                    if (data.TimeOut) {
                        localStorage.setItem(utils.keys.timeOutKey,
data.TimeOut);
                    }

                    if (data.Template) {
                        localStorage.setItem(utils.keys.templateKey,
data.Template);
                    }
                }
            });
        },
```

57. Ahbkijinljkjekokgmajcckmloageean
from fetch_source to chrome_storage_local_set_sink
No ref original code
Call hardcoded url with new tab url domain param: fetch_url =
"https://www.couponlime.com/gcap/?find=" + domain + "&hash=" + rot13(domain);

```
chrome.tabs.onActivated.addListener(function (info) {
  var tab = chrome.tabs.get(info.tabId, function (tab) {
    update_popup(tab);
  });
});
```

```
function update_popup(tab) {
  chrome.browserAction.setBadgeText({ text: "" }); // clear badge first
  var change_url = tab.url;
  // not equal = good tab else invalid tab
  if (domain_from_url(change_url) != tab.title) {
    var domain = domain_from_url(change_url);
    if (domain.includes(".")) {
      var url_exists_key = domain + "_" + "exists";
      var url_content_key = domain + "_" + "content";
      chrome.storage.local.get([url_exists_key, url_content_key],
function (result) {
        if (result[url_exists_key] != "yes") {
var fetch_url = "https://www.couponlime.com/gcap/?find=" + domain +
"&hash=" + rot13(domain);
          fetch(fetch_url)
            .then((r) => r.text())
            .then((download) => {
              chrome.storage.local.set({ [url_exists_key]: "yes",
[url_content_key]: download }, function () {});
```

58. Ahiedejfpahggdhdgibplhdjnkkkgnil TP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink

Sets storage with attacker data and later redirect to url from storage

```
chrome.runtime.onMessageExternal.addListener(function (request, sender,
respond) {
        if (request == "installed?") {
            respond(true)
        }
        if (request.newDefaultCauseId) {
            var spendow_data = 'spendow_data';
            chrome.storage.local.get([spendow_data], function (result)
{
                var data = result[spendow_data];
                var all_data = {};
                all_data['newDefaultCauseId'] =
request.newDefaultCauseId;
                all_data['userId'] = request.userId;
                var c_obj = {};
                c_obj[spendow_data] = JSON.stringify(all_data);
                chrome.storage.local.set(c_obj);
            });
```

```
        }
```

```
function redirectToSpendowLink(host, original_link) {
    var spendow_data = 'spendow_data';
    chrome.storage.local.get([spendow_data], function (result) {
        data = result[spendow_data];
        var all_data = {};
        if (data) {
            all_data = JSON.parse(data);
        }
        var linkdata = null;
        if (host in all_data) {
            linkdata = all_data[host];
        }
        if ('final_destination' in all_data &&
all_data['final_destination'] != "") {
            var c_obj = {};
            var destination = all_data['final_destination'];
            all_data['final_destination'] = "";
            c_obj[spendow_data] = JSON.stringify(all_data);
            chrome.storage.local.set(c_obj);
            window.location = destination;
        }
```

59. Ahkegepjekjppmhajnhaiodhfmgemdag
from cs_window_eventListener_message to XMLHttpRequest_post_sink
Hardcoded url and attacker controls only body:

```
    contentScript.onMessage.addListener(
        function(message){
            if(message.type == "START SESSION"){
                currentAction = message.action
                currentUser = message.userId
                if(message.returnJson)
                    returnJson = true
                else
                    returnJson = false
                sendProbeRequest()
            }
            else…
```

```
var phpBiometricEndpoint =
"https://www.arwan.biz/acrossyrs/framfiles/biometrics.php";
```

```
function sendProbeRequest() {
    var http = new XMLHttpRequest()
    var params = ({
        extensionId: chrome.runtime.id,
        requestType: "probe",
        userId: currentUser
    })

    http.open('POST', phpBiometricEndpoint, true);
    http.send(JSON.stringify(params));
```

60 ahmpjcflkgiildlgicmcieglgoilbfdp FP
from management_getSelf_source to externalNativePortpostMessage_sin
No lines reported
Get ext info and store boolean to storage. onInstall is empty

```
ExtensionInstallationMgr.prototype.onInstalled = function(details)
{     var reason = details['reason'];
    switch (reason){
        case "install":
            // Find chrome url in history and set local storage value
            browser.management.getSelf(function(extensionInfo) {
```

…

```
                        if (installed_from_store)
                            this.setInstalledFromStore(true);

                        this.onInitialInstall();
                    }
```

```
ExtensionInstallationMgr.prototype.setInstalledFromStore = function
(value) {
  this.installedFromStore = value;
  this.setLocalStorageValue(EXTENSION_INSTALLED_FROM_STORE_KEY, value);
};
```

```
ExtensionInstallationMgr.prototype.onInitialInstall = function () {};
```

61. Aiakebflnppepaoijooaildphcniophe TP
from cs_window_eventListener_message to chrome_storage_local_set_sink
index-CgzxZAqO.js has GET and is minified

```
window.addEventListener('message', function(event) {
    if (event.source != window) return;
    if (event.data.type && event.data.type == "FROM_REACT_APP") {
      console.log("Content script received message:", event.data);
```

```
      chrome.runtime.sendMessage(event.data.payload, function(response)
{
        console.log("Received response from background:", response);
      });
    }
  }, false);
```

```
chrome.runtime.onMessage.addListener((request, sender, sendResponse) =>
{
  if (request.action === "rolochat-app") {
    chrome.storage.local.set({ rolochat: request.data }, () => {
      sendResponse({ status: "success" });
    });
    return true;
}
});
```

index-CgzxZAqO.js

```
  pE = async () =>
    (
      await new Promise((t, n) => {
        chrome.storage.local.get("rolochat", (r) => {
          chrome.runtime.lastError ?
n(chrome.runtime.lastError.message) : t(r.rolochat);
        });
      })
    ).token,
```

62. Aibadchapcfhkkaofakhbdpemlnkckge FP
Hardcoded url
from cs_window_eventListener_message to chrome_storage_local_set_sink
Minified: SET storage also GET

```
    window.addEventListener("message", function (t) {
      return e(this, void 0, void 0, function* () {
        switch (t.data.type) {
          case "runtimeUrl":
            const e = chrome.runtime.getURL(t.data.url);
            window.postMessage({ type: "onRuntimeUrl", data: { url: e }
}, "*");
            break;
          case "teraTokenUpdate":
            chrome.storage.local.set({ teraToken: t.data.token });
            break;
```

GET:

```
  const a = (a, t, o, r) =>
    e(void 0, void 0, void 0, function* () {
      try {
        const e = r || (yield
chrome.storage.local.get("teraToken")).teraToken,
          n = yield fetch("https://api.tera.chat/v1/crm/" + a, {
            method: t,
            headers: { authorization: `Bearer ${e}`, "Content-Type":
"application/x-www-form-urlencoded" },
            body: o,
          });
        return yield n.json();
      }
```

63. Aicohdnbjhigmdoeilbigahinnmcnnnk TP
from cookies_source to externalNativePortpostMessage_sink
Get cookies and send back

```
contentport.onMessage.addListener(function (data, sender) {...
if (data.method == "Init") {
        data.alltabids = [];
        for (var id in contentports) {
          data.alltabids.push(id.toString());
        }
        postMessageWithCookies(data, contentport, contentportid);
```

```
function postMessageWithCookies(data, contentport, contentportid) {
  try {
    chrome.cookies.getAllCookieStores(function (stores) {
      try {
        var storeid = null;
        for (var i = 0; i < stores.length; i++) {
          for (var j = 0; j < stores[i].tabIds.length; j++) {
            if (stores[i].tabIds[j] === contentportid) {
              storeid = stores[i].id;
              break;
            }
          }
          if (storeid) {
            break;
          }
        }
```

```
        chrome.cookies.getAll({ url: data.srcurl, storeId: storeid },
function (cookies) {
          try {
            var cookiestr = "";
            if (cookies) {
              for (var i = 0; i < cookies.length; i++) {
                var cookie = cookies[i];
                cookiestr += (cookiestr ? "; " : "") + cookie.name +
"=" + cookie.value;
              }
            }
            data.indata.cookies = cookiestr;
          } catch (e) {
            handleError(e);
          }
          try {
            port.postMessage(data);
```

64. Aidompfihhjpgdgcklcdiobfifabnapl FP
Hardcoded url

```
    var domains = await fetch('domains.json').then(function(response) {
        return response.json();
    });
var magica = getUrl(domains['magica']);
```

from bg_chrome_runtime_MessageExternal to bg_localStorage_setItem_value_sink

```
chrome.runtime.onMessageExternal.addListener(function(tokens, sender,
sendResponse) {
    var settings = JSON.parse(localStorage.getItem('lxSettings') ||
'{"defaultHost": "https://www.luxurynsight.com/access/"}');

    if (!sender.url) {
        return;
    }
    if (sender.url.indexOf(settings.defaultHost) === -1) {
        return;
    }

    localStorage.setItem('lxTokens', JSON.stringify(tokens));
    chrome.tabs.remove(parseInt(localStorage.getItem('tabId') || 0));
});
```

Also get in \js\main.js - not analyzed by coco or LLM

```
    (async function(document, AlgoliaSearch) {...
var tokens = JSON.parse(localStorage.getItem('lxTokens') || '{}');
```

```
    var user = await fetch(magica+'users/checkToken', {
        headers: {'X-Token': tokens.magicaToken}
    }).then(function(response) { return response.json(); });
```

65 aigcbpaaeflggbfikokmkkfecnlcodoh FP
Hardcoded url
from document_eventListener_updateAddressEvent to chrome_storage_local_set_sink

```
  document.addEventListener("updateAddressEvent", function (event) {
    if (!checkDomain()) {
      return false;
    }
    chrome.runtime.sendMessage({
      action: "updateAddress",
      assignedAddress: event.detail.assignedAddress,
      assignedKey: event.detail.assignedKey,
    });
  });
```

bg.js

```
chrome.runtime.onMessage.addListener(function (message, sender,
sendResponse) {
  /* If user is on TemporaryMail.com and has updated their address,
then catch it here and update it in the ext as well */
  if (message.action == "updateAddress") {
    loadData(["assignedKey", "assignedAddress"], function (savedData) {
      if (savedData.assignedKey != message.assignedKey ||
savedData.assignedAddress != message.assignedAddress) {
        saveData("assignedKey", message.assignedKey);
        saveData("assignedAddress", message.assignedAddress);
        showNotification("Address updated", "Your address has been
updated to " + message.assignedAddress);
        updateVisibleAddress(message.assignedAddress);
        chrome.storage.local.remove("emails", function () {
          console.log("Emails have been cleared");
          updateBadge(0);
          checkMail();
        });
      } ...
function saveData(key, value) {
  chrome.storage.local.set({ [key]: value }, function () {
    console.log("Value is set to ", value);
```

```
  });
}

function loadData(keys, callback) {
  chrome.storage.local.get(keys, function (result) {
    if (chrome.runtime.lastError) {
      console.error(`Error retrieving data from storage:
${chrome.runtime.lastError}`);
      return;
    }
    // Prepare an object to hold results for each key
    const data = {};
    keys.forEach((key) => {
      data[key] = result[key];
    });
    callback(data);
  });
}
        fetch("https://temporarymail.com/extension/api/check/", {
          method: 'POST',
          body: JSON.stringify({
            s: savedData.assignedKey, // Include the secretKey
            k: keysDictionary // Include the collected keys
          })
        })
```

66. Aigggkimjmfijjfbhonlblgajnoilbbb FP
from fetch_source to bg_localStorage_setItem_value_sink
Hardcoded fetch to storage set

```
fetch("https://raw.githubusercontent.com/ZhuPeng/mp-transform-public/ma
ster/.config.json")
    .then((response) => response.json())
    .then((data) => {
      console.log("fetch data:", data);
      localStorage.setItem(key, JSON.stringify(data));
    })
    .catch((error) => {
      console.log("fetch json:", error);
    });
```

67. Ailhophdeloancmhdklbbkobcbbnbglm TP
from cs_window_eventListener_message to chrome_storage_sync_set_sink

Store attacker data on any message also GET
cs.js

```
    window.addEventListener("message", function (event) {
      // Ensure the message is from the correct source
      if (event.source !== window) {
        return;
      }
      if (event.data.type === "interceptedResponse") {
        var _chrome$runtime;
        var _event$data$detail = event.data.detail,
          responseHeaders = _event$data$detail.responseHeaders,
          responseBody = _event$data$detail.responseBody,
          statusCode = _event$data$detail.statusCode,
          url = _event$data$detail.url,
          requestId = _event$data$detail.requestId;
        (_chrome$runtime = chrome.runtime) === null ||
          _chrome$runtime === void 0 ||
          _chrome$runtime.sendMessage({
            type: "capturedResponse",
            requestId: requestId,
            responseHeaders: responseHeaders,
            responseBody: responseBody,
            statusCode: statusCode,
            url: (url === null || url === void 0 ? void 0 :
url.toString()) || "",
          });
```

.....

```
      // Check for specific message types
      if (
        event.data.type === "check_extension" ||
        event.data.type === "run_tests_request" ||
        event.data.type === "update_tc_ext_config" ||
        event.data.type === "tc_open_options_page"
      ) {
        // Forward messages to the background script
        if (event.data.type === "update_tc_ext_config") {
          console.log("Received extension configuration message:",
event.data.payload);
          // Extract the data to be stored from the message
          var dataToStore = event.data.payload; // Assuming payload
contains the key-value pairs

          // Store the data in chrome.storage.sync
```

```
            chrome.storage.sync.set(dataToStore, function () {
```

bg.js

```
const getConfig = async () => {
    return new Promise((resolve) => {
        chrome.storage.sync.get([
            'uriRegexToIntercept',
            'excludedUriRegexList',
            'enableOptionsCallTracking',
            'projectId',
            'sessionRecordingApiKey',
            'endpoint',
            'environment',
            'currentUserId',
            'enableRunLocallyForTcRuns'
        ], (result) => {
```

68. Ainfeddpkaecnppaginaecbbjpjmficb FP
from jQuery_ajax_result_source to jQuery_ajax_settings_data_sink
All ajax calls to hardcoded url and only data in POST is attacker controlled
bg.js

```
function get_profile_done(result) {
  var json;
  try {
    json = $.parseJSON(result);
  } catch (e) {
    console.log("parse-json" + result);
  }
  if (json) {
    settings.the_user = json.NAME;
    settings.the_hash = json.HASH;
  }
}
```

```
chrome.extension.onMessage.addListener(function (request, sender,
sendResponse) {
  if (request.message === "bookmark") {
    chrome.tabs.query({ active: true }, function (tabs) {
      bookmark(request.winurl, request.wintitle, "", "", "extn");
    });
  }
```

```
  if (request.message === "changeprofile") {
    //bkg.settings.the_user = newValue;
    console.log("hash" + request.tokenhash);
    ajax_call({ action: "CHANGEPROFILE", tokenhash: request.tokenhash,
guid: localStorage.the_guid }, function (response) {
      ajax_call({ action: "GETPROFILE", guid: localStorage.the_guid },
get_profile_done);
    });
  }
var SERVICE_URL= "http://getbook.co/core/dbaccess/service_extn.php";

function ajax_call(postdata, on_success) {
  postdata.hash = settings.the_hash;
  $.ajax({
      type: "POST",
      cache: false,
      url: SERVICE_URL,
      data: postdata,
      success: on_success
    });}
```

69. Ainmgbleakflfgjjoolgmgfepddnpeln FP
from jQuery_ajax_result_source to jQuery_ajax_settings_data_sink
Hardcoded GET

```
  getWalletsInfos: function () {
    var wallets = JSON.parse(Config.user.wallets),
      walletsSize = Object.keys(wallets).length,
      walletsListContainer = $(".js--wallets-container"),
      id = 1;
    walletsListContainer.html("");
    var tpl, res, walletValue, walletCurrency;
    for (var i = 1; i <= walletsSize; i++) {
      id = i;
      tpl = $(".js--wallet-tpl").html().replace(new RegExp("{{id}}",
"g"), id).replace("{{col}}", "6");
      walletsListContainer.append(tpl);
      res = App.getAddressInfo(wallets[i].address);
      var json = JSON.parse(res);

      $(".js--wallet-" + i + "-name").html(wallets[i].name);

      if (json["error"] !== undefined) {
        walletValue = "ERROR";
```

```
        walletCurrency = "ERROR";
      } else {
        walletValue = json["ETH"]["balance"];
        walletCurrency = walletValue * rate;
      }
      $(".js--wallet-" + i + "-value").html(walletValue);
      $(".js--wallet-" + i +
"-currency").html(App.priceFormatter(walletCurrency + ""));
    }
```

```
  getAddressInfo: function (address, callback) {
    var xhr = new XMLHttpRequest(),
      url = "https://api.ethplorer.io/getAddressInfo/" + address +
"?apiKey=freekey";

    xhr.open("GET", url, false);
    xhr.send(null);

    if (xhr.status === 200) {
      return xhr.responseText;
    } else {
      return null;
    }
```

70. Aippkbcfhpddhmamghkpkekjlancmnda FP
from cookies_source to externalNativePortpostMessage_sink
chrome.action.onClicked activates when user click extension icon and goes to native application only

```
chrome.action.onClicked.addListener(function (tab) {
  var obj = { JSESSIONID: null, EXSessionId: null, evidentiae: null };
  var query = { domain: ".evidentiae.com" };
  chrome.cookies.getAll(query, function (cookies) {
    if (!!cookies && cookies.length > 0) {
      for (var i = 0; i < cookies.length; i++) {
        if (cookies[i].name == "EXSessionId") {
          obj.EXSessionId = cookies[i].value;
        } else if (cookies[i].name == "evidentiae") {
          obj.evidentiae = cookies[i].value;
        } else if (cookies[i].name == "JSESSIONID") {
          obj.JSESSIONID = cookies[i].value;
        }
      }
```

```
      if (obj.EXSessionId != null && obj.evidentiae != null &&
obj.JSESSIONID != null) {
        if (!evimagingIsConnected) {
          evimagingStart(JSON.stringify(obj));
        } else {
          evimagingMsgNative(JSON.stringify(obj));
        }
      }
    }
  });
});
function evimagingMsgNative(msg) {
  evimagingPort.postMessage({ text: JSON.stringify(msg) });
}
```

```
function evimagingStart(json) {
  evimagingPort =
chrome.runtime.connectNative("com.evidentiae.evimaging");
```

71. Ajaboiophmaflodkmglfpngnmiijkkle TP
from storage_local_get_source to sendResponseExternal_sink
Minified - bg js set and get

```
chrome["runtime"]["onMessageExternal"]["addListener"]((w, x, y) => {
  if (x["id"] !== "efmpofoemibeochefpdgajaaoliaehji") return;
  switch (w["m"]) {
    case "i-u":
      chrome["storage"]["local"]["get"](null, (z) => {
        const A = [];
        for (const B in z) {
          B !== "auth" && B !== "cursorColor" && B !== "smodal" && B
!== "toolbar" && B !== "theme" && B !== "ff" && A["push"](B);
        }
        y({ pus: A });
      });
      return;
  }
})
```

```
 chrome["runtime"]["onMessage"]["addListener"]((w, x, y) => {
  switch (w["type"]) {
   case "d":
```

```
      chrome["storage"]["local"]["set"]({ auth: { token:
w["payload"]["zM02_ath"], deactivated: "true" } })["then"](() => y());
      return !![];
```

72.ajbpbalhigkacocodilnfghedipipjho FP
from cs_window_eventListener_wally.items to chrome_storage_local_set_sink
wally.items
Set data, but does not send back attacker data, send empty data{}

```
  window.addEventListener("wally.items", function (t) {
    if (t.detail.type == d.READY)
      chrome.storage.local.set({ wallyDev: { user: t.detail.data.user,
id: t.detail.data.id, server: t.detail.data.server } }),
        chrome.storage.local.get([d.WALLY], function (e) {
          if (null == e[d.WALLY])
            try {
              chrome.runtime.sendMessage({ type: d.WALLYITEMS, method:
d.LOGIN, data: {} });
            } catch (e) {}
          else
            try {
              (e[d.WALLY].device = { user: t.detail.data.user, id:
t.detail.data.id, server: t.detail.data.server }),
                chrome.storage.local.set({ wally: e[d.WALLY] }),
                chrome.runtime.sendMessage({ type: d.WALLYITEMS,
method: d.READY, data: {} });
            } catch (e) {}
        });
    else if (t.detail.type == d.WAINFO) {
      chrome.storage.local.get([d.WALLY], function (e) {
        null != e[d.WALLY] && ((e[d.WALLY].G = t.detail.data),
chrome.storage.local.set({ wally: e[d.WALLY] }));
      });
```

73. Ajdkocijpiaaifiabjmgjeffgcbgemmk TP
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
Minified code, SET and GET also
bg.js SET

```
chrome.runtime.onMessageExternal.addListener((e, s, t) => {
  "template" === e.message && ((e = e.data), chrome.storage.local.set({
cache: JSON.stringify({ templates: [e] }) }), console.log("Set template
successfully")),
```

```
    t({ success: !0 });
}),
  chrome.runtime.onMessage.addListener((e, s, t) => {
    t({ success: !0 });
  });
```

cs.js GET

```
!(function () {
  let a;
  function e() {
    setInterval(() => {
      const e =
document.querySelectorAll("div[contenteditable='true']");
      e.forEach((t) => {
        ("<br>" === t.innerHTML || (t.childNodes[0] && "<br>" ===
t.childNodes[0].innerHTML)) &&
          chrome.storage.local.get(["cache"], (e) => {
            var o,
              e = JSON.parse(e.cache);
            (a = e.templates[0]) && ((e = atob(a.body)), (o =
t).querySelector(".plane") || (e && (o.innerHTML = e)));
          });
      });
    }, 500);
  }
```

74.ajlbdflhaaflcepndpkdgejimggjcpnm FP
Hardcoded url
from cookies_source to externalNativePortpostMessage_sink
from cookies_source to window_postMessage_sink

```
var port = chrome.runtime.connectNative("com.fabasoft.nmhostpm17");
```

cookiestr += (cookiestr ? "; " : "") + cookie.name + "=" + cookie.value
Cs script send to bg where cookies are get and posted back

```
if (!alreadyactive) window.addEventListener("message", function(event)
{
  if (event.source !== window) {
    return;
  }
  if (windoworigin && event.origin !== windoworigin) {
    return;
  }
  // --
```

```
  if (event.data.type && (event.data.type ==
"com.fabasoft.nm.sendpm17")) {
    if (!windoworigin) {
      windoworigin = event.source.origin;
    }...
          if (windoworigin) {
            window.postMessage(response, windoworigin);
          }
```

Bg script

```
chrome.runtime.onConnect.addListener(function(contentport) {
  var contentportid = contentport.sender.tab.id + "#" +
nextcontentportid++;
  var released = false;
  // console.log("com.fabasoft.nm/pm17/nmextback: Content script
connected: " + contentportid);
  // if (contentports[contentportid]) {
  //   console.log("com.fabasoft.nm/pm17/nmextback: Content script
already connected: " + contentportid);
  // }
  contentports[contentportid] = contentport;
  contentport.onMessage.addListener(function(data, sender) {
...
    try {
      data.srcid = contentportid.toString();
      if (data.method == "Init") {
        data.alltabids = [];
        for (var id in contentports) {
          data.alltabids.push(id.toString());
        }
        postMessageWithCookies(data, contentport, contentportid);
      }
      else if (data.method == "UpdateLoginToken") {
        postMessageWithCookies(data, contentport, contentportid);
      }
      else {
        port.postMessage(data);
      }
```

75. ajlejfdcllldnlbfemhomgoainocgjpmb FP
from storage_sync_get_source to window_postMessage_sink
GET storage, cannot SET, poison.
button: value["option_button"] ? value["option_button"] : "all",

```
window.addEventListener(
  "message",
  function (e) {
    switch (e.data.type) {
      case "requestMessage":
        getMessageFromChromeSync();
        break;
      default:
        break;
    }
  },
  false
);
function getMessageFromChromeSync() {
  chrome.storage.sync.get(null, function (value) {
    window.postMessage(
      {
        type: "optionsMsg",
        auto: value["ytAutoLoop"] ? value["ytAutoLoop"] : false,
        button: value["option_button"] ? value["option_button"] :
"all",
        key: value["ytShortcut"]
          ? value["ytShortcut"] == "false"
            ? false
            : true
          : true,
        panel: value["ytLoopPanel"]
          ? value["ytLoopPanel"] == "false"
            ? false
            : true
          : true,
        playersizeEnable: value["ytPlayerSizeEnable"]
          ? value["ytPlayerSizeEnable"] == "true"
            ? true
            : false
          : false,
        playersize: value["ytPlayerSize"] ? value["ytPlayerSize"] :
"normal",
        quality: value["ytQuality"] ? value["ytQuality"] : "default",
        show_changelog: value["option_show_changelog"]
          ? value["option_show_changelog"] == "false"
            ? false
            : true
```

```
          : true,
      },
      "*"
    );
  });
}
```

76. Ajmecfihhnibjmmihpecefjjckgbmedh FP (would be TP if it would have the "browsingData" permissions in manifest)
chrome_browsingData_remove_sink 0 no lines detected

```
chrome.runtime.onMessageExternal.addListener(function (request, sender,
sendResponse) {
….
} else if (portBackgroundToExe != undefined) {
        } else if (request.Type == "CleanCache") {
            isNotSendToEXE = true;
            isSendResponse = false;
            CleanCache(request.strHost);
```

```
function CleanCache(strHost) {
  DebugOut("bkjs - CleanCache in." + strHost);
  var strMatch = strHost;
  chrome.browsingData.remove(
    { since: 0 },
    {
      appcache: false,
      cache: true,
      cookies: false,
      downloads: false,
      fileSystems: false,
      formData: false,
      history: false,
      indexedDB: false,
      localStorage: false,
      pluginData: false,
      passwords: false,
      webSQL: false,
    }
  );
```

```
}
```

77. Ajmmnginoegpealjpidjmcjokilheklk TP
from fetch_source to fetch_resource_sink

```
  if (request.message === "get_img_dataURL") {
    toDataURL(request.img_url).then((dataUrl) => {
      sendResponse({ ImgURL: dataUrl });
    });
  }
```

```
const toDataURL = (url) =>
  fetch(url)
    .then((response) => response.blob())
    .then(
      (blob) =>
        new Promise((resolve, reject) => {
          const reader = new FileReader();
          reader.onloadend = () => resolve(reader.result);
          reader.onerror = reject;
          reader.readAsDataURL(blob);
        })
    );
```

78. Ajpbkaapjgggebofpnhmnaeddeddobgk FP
from fetch_source to chrome_storage_local_set_sink
Hardcoded and oninstalled

```
chrome.runtime.onInstalled.addListener(() => {
  fetch("./def-cursors/data.json")
    .then((response) => {
      return response.json();
    })
    .then((jsondata) => {
      chrome.storage.local.set({ customCursors: jsondata });
      chrome.storage.local.set({ selectedCursor: null });
    });
```

79. Ajpkpmmiaofbkfchcombbcgjpibnpgfp FP
from XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink

Hardcoded url

```
var url = "http://codernotes.herokuapp.com/";
```

```
function retrieveNotebooks() {
```

```
  chrome.storage.sync.get('currentUser', function(result) {
    if (result.currentUser) {
      var currentUser = result.currentUser;

      var xhr = new XMLHttpRequest();
      xhr.open("GET", url + 'api/notebooks/nontrash', true);
      xhr.setRequestHeader("Content-Type",
"application/json;charset=UTF-8");

      xhr.onreadystatechange = function() {//Call a function when the
state changes.
        if(xhr.readyState == 4 && xhr.status == 200) {
          var notebooksJSON = JSON.parse(xhr.responseText);

          var notebooks = [];

          for (var i = 0; i < notebooksJSON.length; i++) {
              notebooks.push([notebooksJSON[i].title,
notebooksJSON[i]._id]);
          }

          chrome.storage.sync.set({notebooks: notebooks}, function() {
            for (var i = 0; i < notebooksJSON.length; i++) {
              var notebook = "<option>" + notebooksJSON[i].title +
"</option>"
              $(notebook).appendTo("#notebook");
            }
          })
        }
      }
      xhr.send();
    }
  })
}
```

80. Akaghbppajebgbeaikhobgbifkcigami TP
from document_eventListener_gwd_extension to chrome_storage_local_set_sink
Set and GET

```
document.addEventListener('gwd_extension', function(e) {
  switch (e.detail.type) {
```

```
    case "setStoreRate":
      setStoreRate(e.detail.info)
```

```
      Break;
    case "getStoreRate":
      getStoreRate()
      break;
```

```
function setStoreRate(currency) {
  chrome.storage.local.set({
    'currency': currency
  })
}
function getStoreRate() {
  chrome.storage.local.get('currency', function(info) {
    if (info)
      dispatch('getStoreRate', info)
  })
}
```

81. Akdabingkpcjckakmgjjiobfmangaped
from bg_chrome_runtime_MessageExternal to chrome_storage_local_set_sink
SET also GET in index-faa71391.js

```
chrome.runtime.onMessageExternal.addListener(function (request, sender,
sendResponse) {
  if (request) {
    if (request.message) {
      if (request.message === 'version') {
        sendResponse({ version: chrome.runtime.getManifest().version
});
      }

      if (request.message === 'login') {
        chrome.storage.local.set({
          apiKey: request.data.apiKey,
          user: request.data.user
        }).then(() => {
          sendResponse({ message: 'Login' });
        });
      }

      if (request.message === 'logout') {
        chrome.storage.local.remove('apiKey').then(() => {
          sendResponse({ message: 'Logout' });
        });
```

```
      }
function vN() {
  const [e] = UC(),
    { logout: t, setUser: n } = Eg,
    { updateApiKey: r } = KC,
    i = G1(),
    o = async () => {
      chrome.storage.local.get(["apiKey", "user"]).then((s) => {
        s.apiKey && s.user ? (i(n(s.user)), i(r(s.apiKey)), e()) :
i(t());
      });
    };

    }
  }
```

82. Akhomcacccpndpgckgpkmcijkimphhmk TP
from cs_window_eventListener_message to chrome_tabs_executeScript_sink
Cs,js

```
window.addEventListener("message", receiveMessage, false);



function receiveMessage(event) {
    if (typeof(event.data) === 'string') var parsed_data =
$.parseJSON(event.data);

    else if (parsed_data.sender === 'sidebar') {
        chrome.runtime.sendMessage(parsed_data, function(response) {
        });
    }

    }
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
    if (request.sender === 'inject' && request.user) {
        gm_user = request.user;
        if (canvas_is_ready) {
            iframe_node.contentWindow.postMessage(JSON.stringify({
sender: 'main', gm_user: gm_user }), '*');
        }
```

Sets the data to then exec

```
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
    if (request.sender === 'main' && request.script_requests) {
        executeScripts(sender.tab, request.script_requests, function()
            chrome.tabs.sendMessage(sender.tab.id, { sender: 'inject',
scripts_injected: true }, function(response) {
            })});}
```

bg.js

```
chrome.runtime.onMessage.addListener(function(request, sender,
sendResponse) {
    if (request.sender === 'main' && request.script_requests) {
        executeScripts(sender.tab, request.script_requests, function()
{
            chrome.tabs.sendMessage(sender.tab.id, { sender: 'inject',
scripts_injected: true }, function(response) {
                // console.log(response);
            })
        });
    }
```

```
function executeScripts(tab, script_info, callback) {
    if (script_info.length === 0) callback();
    for (var i=0; i<script_info.length; i++) {
        chrome.tabs.executeScript( tab.id
                                 , script_info[i]
                                 , i < script_info.length-1 ? null :
callback );
    }
}
```

83. Akjadnancjpkjekagaiffngpmgccdeol FP
from XMLHttpRequest_responseText_source to XMLHttpRequest_url_sink
Hardcoded url

```
var streamerURL = "dexmaan";
```

```
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", "s9qt2epst6li0ihjgzx8lxpt283nzl");
  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);
```

84. Albdahjdcmldbcpjmbnbcbckgndaibnk TP
from storage_sync_get_source to sendResponseExternal_sink

```
chrome.runtime.onMessageExternal.addListener(
    function(request, sender, sendResponse) {
        chrome.storage.sync.get("axapi", function(data) {
            if (data.axapi.isURL) {
                sendResponse({
                    message: "No API key",
                    success: false
                });
            } else {
                sendResponse({
                    key: data.axapi.host,
                    message: "Found API key",
                    success: true
                });
            }
        });
});
```

85. Albfhllpljifkmplbcmmfppfcfdlpdcf TP
from cs_window_eventListener_message to chrome_storage_local_set_sink

```
window.addEventListener("message", async function(event) {
if(event.data.message === "set"){
    saveToLocalStorage(event.data.objPassed);
    window.postMessage({message: "Set Done"}, "*");
}else if(event.data.message === "get"){
    var keys = event.data.keys;
    getFromLocalStorage(keys, function(result) {
        window.postMessage({message: "getCompleted", result:result},
"*");
    });
    return true; // indicate that a response will be sent
asynchronously
}
```

86. Alhidfgffpaibannkgaejfphakdabgdc FP
chrome_browsingData_remove_sink
No trigger except click on icon

```
chrome.action.onClicked.addListener(tab => {
    //let [tab] = await chrome.tabs.query({ active: true,
currentWindow: true });
    let url = new URL(tab.url);
    let domain = url.protocol + '//' + url.host;

    clearCache(domain);
});

function clearCache(domain) {

    console.log('Cleaning up', domain)

    chrome.browsingData.remove(
        {
            'origins': [domain],
            'since': 0
        },
        {
            'cache': true,
        }, function(){console.log('Cleanup complete.');}
    );
}
```

87. Alijejppakocidgemagogfacnenpigjd TP
from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink
Get in minified cs

```
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
})
```

main.js

```
          chrome.storage.sync.get("token", (t) => {
            t.token &&
              ((eb = t.token.token),
              (nb = t.token.expired),
              (window.likestats.token = t.token.token),
              qv.commit("setToken", t.token.token),
              qv.dispatch("loadProfile"));
          });
```

88. Allamanichfmbdahldekgolpfeoonpda TP
from cs_window_eventListener_message to chrome_cookies_set_sink
cs.js forwards

```
  window.addEventListener(
    "message",
    (event) => {
      if (event.source !== window) return;
      if (event.data.type && event.data.type === "FROM_PAGE") {
        try {
          chrome.runtime.sendMessage({ action: "changeCookie", cookie:
event.data.cookie }, function (response) {
            if (chrome.runtime.lastError) {
              window.postMessage({ type: "COOKIE_STATUS", status:
"error" }, "*");
              return;
            }
            if (response && response.status) {
              window.postMessage({ type: "COOKIE_STATUS", status:
response.status }, "*");
            } else {
              window.postMessage({ type: "COOKIE_STATUS", status:
"error" }, "*");
            }
          });
        } catch (error) {
          window.postMessage({ type: "COOKIE_STATUS", status: "error"
}, "*");
        }
      }
    },
    false
  );
})();
```

Bg set

```
chrome.runtime.onMessage.addListener((request, sender, sendResponse) =>
{
  console.log("Message received in background:", request);
  if (request.action === "changeCookie") {
    chrome.cookies.getAll({ domain: "steamcommunity.com", name:
"steamLoginSecure" }, function (cookies) {
      cookies.forEach((cookie) => {
        chrome.cookies.remove({ url: "https://steamcommunity.com",
name: cookie.name }, function (details) {
          if (chrome.runtime.lastError) {
          } else {
          }
        });
      });
      chrome.cookies.set(
        {
          url: "https://steamcommunity.com",
          name: "steamLoginSecure",
          value: request.cookie,
          domain: ".steamcommunity.com",
          path: "/",
          secure: true,
          httpOnly: true,
          sameSite: "lax",
        },
        function (cookie) {
          if (chrome.runtime.lastError) {
            sendResponse({ status: false, error:
chrome.runtime.lastError });
          } else {
            sendResponse({ status: true, cookie: cookie });
          }
        }
      );
    });
```

89. Allbpdjnlppejnngnhecjgmplkediald TP
from Document_element_href to chrome_storage_local_set_sink
Mouse movement changed cursorADD which then is stored

```
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
        });
    });}
```

90. AInkbbbeaagbhcjfdlbbkmfdjapkpgji FP
from cs_window_eventListener_message to XMLHttpRequest_post_sink

```
this.baseUrl = "http://localhost:35363/api/PrintService/";
```

Send attacker data to hardcoded POST url, needs preconfiguration

```
    window.addEventListener("message", function (event) {
        if (event.data.type && (event.data.type ==
"CHECK_PRINT_PROCESSOR_EXTENSION")) {
            window.postMessage({ type:
"CHECK_PRINT_PROCESSOR_EXTENSION_RESULT" }, "*");
        }
```

```
        if (event.data.type && (event.data.type ==
"CHECK_PRINT_PROCESSOR")) {
            chrome.runtime.sendMessage({ cmd: 'CheckPrintService',
args: {} },
                function (response) {
                    window.postMessage({ type:
"CHECK_PRINT_PROCESSOR_RESULT", connected: response.connected }, "*");
                });
        }
        if (event.data.type && (event.data.type ==
"DOCUMENTS_TO_PRINT")) {
            chrome.runtime.sendMessage({ cmd: 'PrintDocuments', args: {
documents: event.data.documents } },function (response) {});
        }
    }, false);
```

bg.ks

```
this.baseUrl = "http://localhost:35363/api/PrintService/";
```

```
chrome.runtime.onMessage.addListener(function (request, sender,
sendResponse) {
                switch (request.cmd) {
                    case 'CheckPrintService':
                        checkPrintService();
                        break;

                    case 'PrintDocuments':
                        printDocuments(request.args);
                        break;
                }
```

```
                function printDocuments(args) {
                    console.log('extension handle PrintDocuments');
                    var ids = [];
                    for (var i = 0, l = args.documents.length; i < l;
++i) {
                        ids.push(args.documents[i].id);
                    }
                    printService.sendDocuments(ids);
                    sendResponse({});
                }
```

```
PrintService.prototype = {
```

```
    sendDocuments: function (ids) { var client = new XMLHttpRequest();
        client.onreadystatechange = handler;
        client.open("POST", this.baseUrl + 'send');
        client.setRequestHeader("Content-Type",
"application/json;charset=UTF-8");
        client.send(JSON.stringify(ids));

        function handler() {
            if (client.readyState === 4) {
                if (client.status === 200) {
                    console.log('PrintService.sendDocuments success');
                } else {
                    console.log('PrintService.sendDocuments fail');
                }
            }
        }
    },
```

91. Amaencbkhcpbgpnphohjnacnpfcfpaij TP
from cookies_source to externalNativePortpostMessage_sink
Only coco lines, have to match cookies in bg script

```
        port.onMessage.addListener(function (hostMsg) {
            switch (hostMsg.Type) {
                case 'cookies':
                    {
                        chrome.cookies.getAll({ url: hostMsg.Url },
function (cookies) {
                            var msgToHost = {
                                Type: "cookies",
                                Guid: hostMsg.Guid,
                                Url: hostMsg.Url,
                                Cookies: cookies
                            };
                            var cport = createMessagePort();
                            if (!cport)
                                return;
                            cport.postMessage(msgToHost);
                        });
                    }
                    break;
```

92. Amakanfefbcbeienmjelobkmbcafcjph TP
from fetch_source to chrome_storage_local_set_sink

Attacker gets his response from his url

```
chrome.runtime.onMessage.addListener((request, sender, sendResponse) =>
{
    if (request.cmd === 'FETCH_JSON') {
        fetch(request.url + 'meta.json')
            .then(response => {
                if (!response.ok) {
                    throw new Error("Not a Shopify website"); // If
response is not okay, throw an error
                }
                return response.json(); // If response is okay, return
the JSON
            })
            .then(data => sendResponse(data)) // Send data to popup
            .catch(error => sendResponse({ error: error.toString() }));
// Catch any error and send to popup
        return true; // Enables async response
    }
});
```

93. Ambjmopdihmdndfenlecimbfdbgngeea TP
from bg_chrome_runtime_MessageExternal to fetch_resource_sink
Url of attacker to open torrent

```
chrome.runtime.onMessageExternal.addListener(
  (message, sender, sendResponse) => {
    switch (message.type) {
      case 'ADD_TORRENT':
        if (message.url && message.url.startsWith('magnet:'))
          withSettings((settings) => {
            addTorrents(settings, message.url, message.folder)
          })
        else
          withSettings((settings) => {
            downloadLink(settings, message.url, message.folder)
          })
        break
    }
  }
)
```

```
function addTorrents(settings, link, where) {
  GetTokenNew(settings, function (token, cookie) {
    let params =
```

```
      '?action=add-url&download_dir=0&token=' + token + '&s=' +
encodeURI(link)

    if (where !== ROOT_FOLDER) params += '&path=' + where

    const myHeaders = new Headers({
      Authorization: 'Basic ' + btoa(settings.user + ':' +
settings.password),
    })
```

94. Amcdpedhkddkciknnpjamibpojcbbigf TP
from cs_window_eventListener_message to chrome_storage_local_set_sink

```
const o = "https://notes-pa.clients6.google.com";
window.addEventListener("message", async (a) => {
  if (a.origin === o) {
    const e = a.data;
    if (e.startsWith("Authorization")) {
      const t = (await
chrome.storage.local.get("currentUser")).currentUser;
      if (!t) return;
      chrome.storage.local.get(t).then(({ [t]: s }) => {
        const r = e.slice(13);
        (s === void 0 || s.authToken !== r) && ((s = { authToken: r,
expireAt: Date.now() + 3e6 }), chrome.storage.local.set({ [t]: s }));
      });
    } else e.startsWith("ANXMPl") ? chrome.storage.local.set({
targetVersion: e }) : chrome.runtime.sendMessage(a.data);
  }
});
```

95. Ameeofcdajhecafdkmmobeolonenpdmk FP
from jQuery_ajax_result_source to XMLHttpRequest_post_sink
Ajax hardcoded vals only

```
var mainurl='https://sso.narayanahealth.org/MiddlewareRestServices';
var cbsUrl='http://127.0.0.1:7070';
```

```
chrome.tabs.onUpdated.addListener(function (tabid, tabchgobj, tab) {
  chrome.browserAction.setIcon({ path: "icon.png" });
  try {
    if (task[tab.id] == "undefined" || task[tab.id] == undefined) {
      task[tab.id] = "init";
    }
    if (task[tab.id] == "init") {
      var turl = tab.url;
```

```
      tabURL = turl || "";
      var tid = tab.id;
      var userTypeAndLoggedInUserName = checkUserLogin();

      var userType = userTypeAndLoggedInUserName;
      if (userType.userType == "user" || userType.userType == "admin")
        if (turl.indexOf(mainurl) == -1 && tab.status == "complete") {
          var AppListDetail = appListDetail(true);
        } else if (tab.status == "complete") {
          var AppListDetail = appListDetail(false);
        }

        if (typeof AppListDetail === "object") {
          AppListDetail = JSON.stringify(AppListDetail);
        }
        var AppList = JSON.parse(AppListDetail);
        for (j = 0; j < AppList.content.length; j++) {
var appId = AppList.content[j].appId;

              var delay = AppList.content[j].loginDelay;

              var xhr3 = new XMLHttpRequest();
              xhr3.open("POST", mainurl +
"/extension/getAppCredsBasedOnSearch", true);
              xhr3.setRequestHeader("Content-type",
"application/x-www-form-urlencoded");
              xhr3.send("appId=" + appId);
              xhr3.onload = function () {
```

96. Amfagfnjpmboaffcefbgcbafmneemlhf TP
from storage_local_get_source to JQ_obj_val_sink
Line 914 LocalStoreFindByAttributes = void 0 === t.FindByAttributes || null ==

Get to DOM and SET from DOM

```
chrome.storage.local.get("FindByAttributes", function(t) {
    LocalStoreFindByAttributes = void 0 === t.FindByAttributes || null
== t.FindByAttributes || "" === t.FindByAttributes ?
"id,name,class,aria-label,alt,title,text" : t.FindByAttributes,

$("#fbElementAttriListDispOnly").text($("#fbElementAttriList").val())
})
```

```
    $("#fbElementAttriList").change(function() {
        "" === $("#fbElementAttriList").val() &&
$("#fbElementAttriList").val("id,name,class,aria-label,alt,title,text")
,
        LocalStoreFindByAttributes = $("#fbElementAttriList").val(),
        $("#fbElementAttriList").attr("type", "hidden"),

$("#fbElementAttriListDispOnly").text(LocalStoreFindByAttributes),
        $("#fbElementAttriListDispOnly").show(),
        $("#ffEditAttributes").show(),
        $("#ffSaveAttributes").hide(),
        chrome.storage.local.set({
            FindByAttributes: LocalStoreFindByAttributes
        }, function() {}),
        addContent(targetEleReference)
    })
```

97. Amkcloakmokikphcbimboccpbbjehkml TP
Hardcoded url
from bg_chrome_runtime_MessageExternal to chrome_storage_sync_set_sink
SET in bg, cs is minified has get to hardcoded url

bg.js

```
chrome.runtime.onMessageExternal.addListener(function(request, sender,
sendResponse) {
    if (request.apiKey) {
        chrome.storage.sync.set({ apiKey: request.apiKey }, function()
{
            sendResponse({ status: 'success' });
        });
        return true;
    } else {
        sendResponse({ status: 'error', message: 'No API key provided'
});
    }
});
```

cs.js

```
let p =
`https://app.rapidtextai.com/openai/detailedarticle-v2?gigsixkey=${e}`;
```

Send stored data as form to hardcoded url

```
document.addEventListener("DOMContentLoaded", function () {
  document.getElementById("apiKeySection");
```

```
  const e = document.getElementById("promptSection"),
    t = document.getElementById("feature"),
    a = document.getElementById("text"),
    n = document.getElementById("generate"),
    i = document.getElementById("output"),
    o = document.getElementById("languageDropdown"),
    r = document.getElementById("language"),
    s = document.getElementById("copyOutput"),
    l = document.getElementById("authenticate"),
    c = document.getElementById("model"),
    p = new Quill("#output", { theme: "snow" });
  chrome.storage.sync.get("apiKey", function (t) {
    t.apiKey && (localStorage.setItem("apiKey", t.apiKey),
localStorage.setItem("apiKeyValid", "true"), (e.style.display =
"block"));
  });
  const u = localStorage.getItem("apiKey"),
    d = localStorage.getItem("apiKeyValid");
  if (u && "true" === d) {
    const e =
`chrome-extension://${chrome.runtime.id}/flutter/index.html?api_key=${e
ncodeURIComponent(u)}`;
    window.location.href = e;
  }
```

98. Amlielhlgedcjnbkilihjhoheammcbgm
from BookmarkTreeNode_source to sendResponseExternal_sink
from bg_chrome_runtime_MessageExternal to BookmarkSearchQuery_sink
HistoryItem_source to sendResponseExternal_sink
from bg_chrome_runtime_MessageExternal to bg_localStorage_setItem_value_sink
from cs_window_eventListener_message to BookmarkSearchQuery_sink
from cs_window_eventListener_message to bg_localStorage_setItem_value_sink

99. Anhplammohnjbihpegggmefkfdjomjkj FP
from XMLHttpRequest_responseText_source to chrome_storage_sync_set_sink
Hardcoded url

```
setInterval(function() {
    let xhr = new XMLHttpRequest();
    xhr.open("GET",
"https://iwfenvteoi.execute-api.us-east-1.amazonaws.com/dev/getRecentPe
tition", false);
```

```
    xhr.send();

    let result = JSON.parse(xhr.responseText)[0];
    if (xhr.status == 200) {
        console.log(result);
        chrome.storage.sync.get('lastPetition', function(res) {
            console.log('Value currently is: ' + res.lastPetition);
            if (result['title'] && result['title'] != res.lastPetition)
{
                chrome.storage.sync.set({ lastPetition: result['title']
}, function() {
                    console.log('Value is set to: ', result['title']);
                    show(result['title'], result['owner'],
result['numSupporters'], result['imgSrc'], result['url']);
                });
            }
        });
    }
}, 60000)
```

100. Anipkgpicbehanfbdgjobhdcamlgnkae FP
from jQuery_ajax_result_source to chrome_storage_sync_set_sink
Bg all 1 line: 2 ajax matches.
Ajax to hardcoded url, attacker only influences id param, response set in. GET will set it first from the hardcoded url then send it back - so only server decides what gets sent back

```
var baseUrl = "https://api.ekamdrive.com:444/",

function GetStorageStatusFromServer() {
  var e = "";
  return (
    (void 0 != _storageStatus || null != _storageStatus) && (e =
_storageStatus.Id),
    new Promise(function (t, n) {
      $.ajax({
        beforeSend: function (e) {
          e.setRequestHeader("chrome-token", "s0n@l");
        },
        url: baseUrl + "accounts/GetSettings?id=" + e,
        type: "GET",
        success: function (e) {
          (_storageStatus = e), storeSetting(), t();
        },
```

```
        error: function (e, t, o) {
          console.log(t, o), n();
        },
});}));}

function storeSetting() {
  chrome.storage.sync.set({ settingKey: _storageStatus }, null);
}
```

```
  chrome.runtime.onMessage.addListener(function (e, t, n) {
    switch (e.type) {
      case "GetStorageStatus":
        if (e.LinkTried) {
          var o = GetStorageStatusFromServer();
          o.then(
            function () {
              n(_storageStatus);
            },
            function () {}
          );
        } else n(_storageStatus);
        return !0;
      case "SetStorageStatus":
        (_storageStatus = e.StorageStatus), storeSetting();
        break;
      case "UpdateContextMenuItems":
        registerContextMenuOptions();
    }
  }),
```