Measure Killer Security Information and Documentation

Version 2.9.3

This document outlines the security details and data handling practices of Measure Killer.

Summary of requirements

This section provides a summary of all outbound connections made by Measure Killer. It is intended for IT and security professionals. Connections marked as **[Optional]** can be ignored if they are blocked or fail. For more information on each connection, please check the other sections in the document.

Domain: api.powerbi.com

- Goal: All Power BI related API calls (All listed below)
- Port: 443
- Protocol: HTTPS

Domain: api.fabric.microsoft.com

- Goal: All Fabric related API calls (All listed below)
- Port: 443
- Protocol: HTTPS

Domain: measurekillerlicense-hrbvcncabnddawb7.switzerlandnorth-01.azurewebsites.net

- Goal: License verification
- Ports: 443 (required)
- Protocol: HTTPS

[Optional] Domain: *.servicebus.windows.net

- Goal: License verification
- Ports: 443 (required), 5671 (optional)
- Protocol: TLS-encrypted AMQP over WebSockets or AMQP over TLS

[Optional] Domain: www.measurekiller.com (hosted on world4you.com)

- Goal: License verification – Blacklisted licenses
- Port: 443
- Protocol: HTTPS

[Optional] Domain: raw.githubusercontent.com

- Goal: Fallback for license verification – Blacklisted licenses
- Port: 443
- Protocol: HTTPS

[Optional] Domain: postman-echo.com

- Goal: License verification – Current date and time
- Port: 443
- Protocol: HTTPS

[Optional] Domain: time.windows.com

- Goal: 1st fallback for license verification – Current date and time
- Port: UDP 123
- Protocol: NTP

[Optional] Domain: pool.ntp.org

- Goal: 2nd fallback for license verification – Current date and time
- Port: UDP 123
- Protocol: NTP

[Optional] Domain: microsoft.com

- Goal: 3rd fallback for license verification – Current date and time
- Port: 443
- Protocol: HTTPS

[Optional] Domain: cloudfare.com

- Goal: 4th fallback for license verification – Current date and time
- Port: 443
- Protocol: HTTPS

[Optional] Domain: ifconfig.me

- Goal: License verification - Public IP address
- Port: 443
- Protocol: HTTPS

[Optional] Domain: www.measurekiller.com (https://measurekiller.com/downloads) (hosted on world4you.com)

- Goal**:** Checking if there are new versions available
- Port: 443
- Protocol: HTTPS

[Optional] Domain: brunner.bi (hosted on world4you.com)

- Goal**:** Start up message
- Port: 443
- Protocol: HTTPS

Data Handling and Privacy Assurance

Measure Killer is meticulously designed with user privacy and data security as paramount concerns. It is important for users to understand the scope and way that Measure Killer interacts with Power BI.

Metadata-Only Interaction

Measure Killer only interacts and utilizes metadata from Power BI Desktop or the Power BI Service. **At no point does Measure Killer read, store, or process actual data contained within Power BI reports or models**. This strict limitation to metadata ensures that sensitive data remains confidential and secure.

Beyond Power BI related data, Measure Killer needs to collect the following metadata for license verification (see also the '**License metadata'** section): Installed Measure Killer version, user and machine name, license key and public IP (optional).

What is considered Metadata?

Metadata is considered the following (examples):

- Names or artifacts (report, semantic models, workspaces, users/email addresses)
- DAX and M expressions including comments
- Names of measures, tables, calculated columns, calculated tables, calculation groups, field parameters, etc.
- Uncompressed size of columns
- All definitions (e.g. name of reports, pages, visuals, visual titles, visual size and other metadata)

Local Processing of Metadata

All processing and analysis of metadata conducted by Measure Killer is performed locally on the user's machine through the installed client software. This approach ensures that the operations on metadata do not involve or require any external processing or storage facilities. By doing so, Measure Killer upholds a high standard of data protection and minimizes potential security risks. **There is no database that is hosted, managed, or used to process any kind of information, everything is done on the user's machine and not transmitted anywhere besides when using XMLA or REST API calls with Microsoft directly**.

XMLA Endpoint Connection

To retrieve metadata from semantic models in the Power BI Service, XMLA endpoints are used. The authentication happens via the entered Microsoft account (see ADOMD client for more specifics on this).

Online / Offline Modes

Measure Killer is designed to be able to operate **both online and offline**:

In the offline modes, Measure Killer connects to the local instance of Analysis Services to retrieve the metadata it needs (The first 2 modes, "Single model and report" and "Shared model on local machine"). In these modes the only time there is an interaction with the internet is upon launch to verify valid licensing (see License Verification below for details)

Online modes (Modes 3, 4 and 5) are triggered when an active internet connection is detected and when the user has a valid license. In these modes there are external connections happening (REST API calls and XMLA connections to Microsoft as outlined above).

API Endpoints and Security Measures

- Secure requests (HTTPS) are made to Power BI API endpoints, ensuring data in transit is encrypted.
- Access tokens are securely passed in request headers, with responses parsed as JSON for internal processing.
- Measure Killer also makes use of NTP (Network Time Protocol) servers such as time.windows.com and pool.ntp.org to obtain the current UTC time for license validation. If NTP access (UDP port 123) is restricted, Measure Killer automatically falls back to secure HTTPS-based time sources (Listed below on the **Date and time** and on the S**ummary of requirements** sections) to ensure time synchronization is still available.
- Additional information, such as announcement messages and latest version releases are also collected from our domain www.measurekiller.com over HTTPS.

License Verification

In version 2.9.3 a new method of license verification was introduced. The verification now is conducted via a single API endpoint (**measurekillerlicense-hrbvcncabnddawb7.switzerlandnorth-01.azurewebsites.net**). A a minimal set of license-related metadata is transmitted via a POST request (More details in the **License metadata** section). The new endpoint is currently in it's experimental stage (version 2.9.3) and users can exclusively use the legacy method (datetime APIs and Event stream) by disabling the "Use new license verification API" in the general settings.

Date and time

Measure Killer needs to get the current date and time from a trusted source, for that it uses NTP via the **ntplib** library in python (version 0.4.0) or APIS via HTTPS:

- Primary: postman-echo.com via HTTPS (port 443)
- 1st fallback: time.windows.com via NTP (UDP port 123)
- 2nd fallback: pool.ntp.org via NTP (UDP port 123)
- 3rd fallback: microsoft.com via HTTPS (port 443)
- 4th fallback: cloudfare.com via HTTPS (port 443)

Public IP lookup (optional)

Measure Killer collects the public IP address with the following:

- ifconfig.me over HTTPS (port 443)

If this service is blocked, Measure Killer will proceed but report the IP as "unknown".

License metadata

For the purpose of validating active licenses and ensuring compliance, Measure Killer transmits a minimal set of license-related metadata to either the new API endpoint via POST request or to a secure EventStream endpoint. The transmitted payload includes the local machine username, license key, installed Measure Killer version, public IP address. No Power BI metadata, report content, or user data is ever included in this process. All data is sent exclusively over encrypted channels (HTTPS/TLS) and is handled in accordance with strict security and privacy practices.

For the new license verification API, only one endpoint is used:

- Domain: measurekillerlicense-hrbvcncabnddawb7.switzerlandnorth-01.azurewebsites.net
- Ports: 443.
- Protocol: HTTPS.

For the legacy license verification method, the Azure Event Hubs libraries for python will be used (azure-eventhub version 5.15.0). To make this work, we need outbound connectivity to the following:

- Domain: *.servicebus.windows.net
- Ports: 5671 (AMQP over TLS), 443 (AMQP over WebSockets) and as a fallback if 5671 cannot be opened.
- Protocol: TLS-encrypted AMQP traffic (not plain HTTP).

Additionally, the tool also requires verifying cryptographically signed blacklist data:

- Primary source: **www.measurekiller.com** over HTTPS (port 443)
- Fallback source: **raw.githubusercontent.com** over HTTPS (port 443)

Version Check and User Notification

To inform users about available updates, Measure Killer queries **https://measurekiller.com/downloads** and **https://en.brunner.bi/post/measure-killer-feedback-1** to compare the installed version against the most recent version available the website and to provide additional information for our users:

- https://measurekiller.com/downloads over HTTPS (port 443)
- https://en.brunner.bi/post/measure-killer-feedback-1 over HTTPS (port 443)

List of REST API Calls

REST API calls are made to Microsoft's Power BI Service for the acquisition of metadata. These API calls are made in compliance with all relevant security protocols. Authentication happens via browser interaction (OAUTH2). Once metadata has been acquired, all subsequent analyses and operations are carried out locally. See the list at the bottom of this document for a list of REST API calls done.

- Datasets - Get Datasets In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/datasets

- Datasets - Get Refresh History In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/datasets/{dataset.id}/refreshes
- Datasets - Get Refresh History: GET https://api.powerbi.com/v1.0/myorg/datasets/{dataset.id}/refreshes
- Datasets - Execute Queries In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/datasets/{dataset.id}/executeQueries
- Reports - Get Reports In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/reports
- Reports - Get Datasources In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/reports/{report.id}/datasources
- Reports - Get Datasources: GET https://api.powerbi.com/v1.0/myorg/reports/{report.id}/datasources
- Reports - Export Report In Group: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/reports/{report.id}/Export
- Reports - Export Report: GET "https://api.powerbi.com/v1.0/myorg/reports/%7breport.id%7d/export">https://api.powerbi.com/v1.0/myorg/reports/{report.id}/Export
- Groups - Get Groups: GET https://api.powerbi.com/v1.0/myorg/groups/{workspace.id}/users
- Groups - Get Groups: GET https://api.powerbi.com/v1.0/myorg/groups
- Admin - Get Capacities As Admin: GET https://api.powerbi.com/v1.0/myorg/admin/capacities
- Admin - WorkspaceInfo GetScanStatus: GET https://api.powerbi.com/v1.0/myorg/admin/workspaces/scanstatus/{scan.id}
- Admin - WorkspaceInfo GetScanResult: GET https://api.powerbi.com/v1.0/myorg/admin/workspaces/scanresult/{scan.id}
- Admin - WorkspaceInfo PostWorkspaceInfo: POST https://api.powerbi.com/v1.0/myorg/admin/workspaces/getinfo?lineage=true
- Admin - Get Activity Events: POST https://api.powerbi.com/v1.0/myorg/admin/activityevents
- Admin - Groups GetGroupsAsAdmin: GET https://api.powerbi.com/v1.0/myorg/admin/groups
- Admin - Groups AddUserAsAdmin: POST https://api.powerbi.com/v1.0/myorg/admin/groups/{workspace.id}/users
- Admin - Groups DeleteUserAsAdmin: DELETE https://api.powerbi.com/v1.0/myorg/admin/groups/{workspace.id}/users/{user.email}
- Admin - Users GetUserArtifactAccessAsAdmin: GET https://api.powerbi.com/v1.0/myorg/admin/users/{user.email}/artifactAccess
- Admin - Reports GetReportSubscriptionsAsAdmin: GET https://api.powerbi.com/v1.0/myorg/admin/reports/{reportId}/subscriptions

- Items - Get Item Definition: POST https://api.fabric.microsoft.com/v1/workspaces/{workspace.id}/items/{item.id}/getDefinition
- Long Running Operations - Get Operation State: GET https://api.fabric.microsoft.com/v1/operations/{operation.id}
- Long Running Operations - Get Operation Result: GET https://api.fabric.microsoft.com/v1/operations/{operation.id}/result
- Domains - List Domain Workspaces GET https://api.fabric.microsoft.com/v1/admin/domains/{domainId}/workspaces
- Domains - List Domains GET https://api.fabric.microsoft.com/v1/admin/domains

Additional Security and Testing Approach

We use a combination of automated scanning and testing along with manual reviews to ensure Measure Killer's security before and after each release. Below are our main practices:

- **Software composition analysis**: When we begin development on a new version, we check all open-source dependencies for known vulnerabilities using pip-audit and OSS Index. Because many issues can be resolved by library updates, we track each new release. We then review any security fixes and confirm compatibility with Measure Killer. If it is stable and relevant, we incorporate it into our codebase.
- **Static code analysis**: We run automated static analysis with tools such as Bandit (a Python security analyzer) and PyLint (for code quality and style). These tools flag potential security weaknesses. Our developers then review each flagged item to determine if it represents a real vulnerability or a false positive.
- **Automated Testing and Code Review**: We use pytest for extensive unit and integration testing. Security-wise, our tests focus on:
- Validation and Parsing Logic: Ensuring .pbix and .rdl files are parsed securely.
- User Authentication: Verifying usage of the azure.identity library and tabular connections.
- Error-Handling and Fail-Safe Behavior: Simulating network or parsing failures using mocks, and testing responses to Power BI API errors.
- Regression Testing: Checking that known issues from previous Measure Killer versions remain fixed.

Rapid response to new vulnerabilities

If a vulnerability is discovered, our team follows a quick release process to track the issue, apply a prompt fix, and release an updated version of Measure Killer. We typically release bug-fix or patch versions on a monthly basis.

These measures, combined with our architecture that avoids storing or processing actual data, are designed to protect sensitive information and maintain a high level of security for all Measure Killer users.

Known CVEs

V8

Regarding the sandbox escape vulnerabilities associated with V8: In Measure Killer, V8 is bundled with the Qt WebEngine library, but our application does not load any external web content or scripts. It only renders internal HTML charts (for example, in the “Plot results” tab). Because no untrusted content is ever processed, **these vulnerabilities cannot be exploited in Measure Killer.**

LLVM

This library is pulled in by PyInstaller when converting our Python code into an executable. Measure Killer does not compile or handle external code, and PyInstaller is only used during development to build the .exe (never at runtime). Thus, there is no way for an LLVM-Based attack in Measure Killer.

Python (3.11.8)

CVEs in Python:

- CVE-2024-8088 (CWE-835)
- CVE-2024-12254 (CWE-400)
- CVE-2017-20052 (CWE-427)
- CVE-2024-7592 and CVE-2024-6232 (CWE-1333 or CWE-400)
- CVE-2015-5652 (Broad CWE category)
- CVE-2025-0938 (CWE-20)
- CVE-2024-6923 (CWE-94)
- CVE-2024-3219 (CWE-306)

These vulnerabilities arise when an application can run or process arbitrary code or malicious inputs. **Measure Killer does not allow external code execution**, nor does it ingest untrusted data. We only pull metadata from Microsoft’s Power BI Service (over HTTPS) and parse local files that the user explicitly opens.

libxml2 and Expat

CVEs associated with libxml2 and Expat (such as CVE-2016-4616, CVE-2016-4615, CVE-2016-4614, CVE-2021-3517, CVE-2024-25062, and more) that typically become relevant if an application processes malformed or hostile XML from unknown sources. In Measure Killer, we only parse .rdl files created by Power BI Report Builder, Microsoft Report Builder, or generated via Power BI’s “Export Report In Group” API. **These .rdl files have a defined structure and come from trusted sources, making the known XML-related exploits not applicable.**

libvpx and libaom

These are part of Qt’s multimedia modules. Measure Killer does not process or render video files, so potential vulnerabilities in these libraries (usually triggered by malformed video content) do not pose a risk in our application.

Qt (version 6.7.2)

Qt provides the main user interface framework. While Qt can load external HTML or JavaScript, Measure Killer does not permit loading unknown URLs or scripts. Similar to V8, we only use Qt WebEngine for charts generated internally by the tool.

Skia (version 118)

Skia is an image rendering engine used indirectly by Qt WebEngine. We do not load or display arbitrary images in Measure Killer; only predefined icons and resources stored locally. **With no untrusted image handling, Skia-related exploits are not a concern.**

SQLite3 (version 3.42.0)

Measure Killer uses SQLite to store activity logs of the Power BI Tenant, such as report views, and Excel activities. **We do not open or process any external .sqlite files, thereby avoiding typical exploitation paths for SQLite vulnerabilities.**