Albatros Bridge

User manual · Manuel d'utilisation · Manuale d'uso · Manual de utilizare · ユーザー マニュアル

Language · Langue · Lingua · Limba · 言語.....Page · Page · Pagina · Pagina · ページ

EN English.....2

FR Français.....23

IT Italiano.....46

RO Română.....67

JA 日本語.....89

DevService.net — Raffaele Romano

▣ User Manual – Albatros Bridge

Version 1.4.129.6121 – Build October 4, 2026

⚠ Important notices

Notice	Details
No affiliation with Meta	The author is in no way associated with, affiliated with, authorized, sponsored or endorsed by Meta Platforms, Inc., WhatsApp LLC or any other company of the Meta group. WhatsApp is a trademark of its respective owners and is mentioned only to describe the function of the program.
Unofficial procedure	The connection to WhatsApp is made through an unofficial procedure (a linked device, like WhatsApp Web, through open source libraries) and not through the official WhatsApp Business APIs. Meta, or any of the companies that operate the WhatsApp product, can change protocols, connection schemes, encryption and terms of use at any time and without notice: the program may therefore stop working, in whole or in part, until it is updated.
A situation common to all unofficial products	There are a great many companies around the world that develop software or offer unofficial WhatsApp services. The problem described in the previous point is the same for all of them: none of them can guarantee that the changes decided by Meta will not disrupt operation, nor predict when they will happen.
Risks for the phone number	Using unofficial clients may not comply with the WhatsApp terms of service and may result in restrictions on, or the banning of, the linked number. The assessment and the related risk are the sole responsibility of whoever uses the program.
Prohibited use: spam, advertising, bulk messaging	Using the program to send spam, commercial advertising or bulk messages is expressly prohibited by the WhatsApp rules. Improper use of the program can therefore lead to the blocking (ban) of the linked WhatsApp number. In that case the ban can in no way be attributed to the author: it is applied directly by Meta because of the improper use that was made of it.
Use “as is”	The program is provided “as is”, without warranties of any kind, express or implied, including warranties of operation, fitness for a particular purpose, continuity of service and absence of errors.
Disclaimer of liability	To the maximum extent permitted by applicable law, no liability can be attributed to the author for direct or indirect damages (loss of data or messages, loss of profits, business interruption, banning of the number or other) arising from the use of, or the inability to use, the program, regardless of how access to it was obtained: paid, as a gift, as free software or in any other way.
Personal data	Saved messages contain personal data. Whoever installs and uses the program is responsible for processing them, for the security of the database and for compliance with data protection legislation (GDPR).

```
**********************************
**        Albatros Bridge         **
** rel. 1.4.129.6121 - 04/10/2026 **
**    © Raffaele Romano 2026      **
**********************************
```

Contents

1. What the program is

2. Requirements

3. Command-line parameters

4. Main files and folders

5. Configuration file Instance_Config.ini

6. Database configuration

7. Encrypted database password and WhatsApp session

8. Starting the program

9. First connection to WhatsApp

10. General operation

11. HTTP interface (endpoints)

12. Connection logs and on-screen log

13. Troubleshooting

14. Best practices

15. Sending from the database (send table)

Appendix A. Configuration examples

1. ▪ What the program is

Albatros Bridge is a bridge between WhatsApp and a database. It connects to WhatsApp through a QR code, like WhatsApp Web, and handles receiving, classifying, persisting and exposing the data over HTTP.

• Receiving: text, images, videos, audio, documents, stickers, locations, statuses, reactions and technical messages.

• Edited messages: handling of edits, including the new LID-encrypted messages when the required messageSecret is available.

• Sending: text and media messages through HTTP endpoints, with validation of the data received.

• Sending from the database: another program writes signed rows to the hh_Albatros_SendQueue table and the bridge sends them (see 15).

• Database: SQL Server, MySQL, MariaDB, SQLite via ODBC, or embedded SQLite.

• Multiple instances: each instance uses its own folder and can have its own WhatsApp session.

• Persistent queues: data stays on disk if the database is unavailable; writing to disk is protected against sudden interruptions, and a repeated save does not create duplicates.

• Chronological order at startup: messages received before the linked number is available are queued on disk and recorded, from oldest to newest, as soon as the number is known.

• Reliable media: download automatically retried and size limit; if the media is not available, the message is saved anyway.

• HTTP security: configurable listening address (BindAddress) and token authentication (ApiToken).

• Protected WhatsApp session: the session files are always encrypted and bound to the PC on which the program is installed.

• Automatic reconnection to WhatsApp with progressive waits.

• Logs: diagnostic log in a table, connection log and copy of the text output.

• Post-insert executable: optional, after a successful save.

2. ✓ Requirements

Requirement	Details
Operating system	Windows 64-bit.
Database via ODBC	64-bit ODBC driver installed for SQL Server, MySQL, MariaDB or external SQLite.
Embedded SQLite	InternalSqlite: does not require an ODBC driver for the database.
Phone	WhatsApp active and available for scanning the QR code at the first connection.
Network	Internet access to the WhatsApp servers. If the HTTP client is on another PC, the Port port must be reachable (Windows Firewall) and setting ApiToken is recommended.
HTTP client	Any program able to send HTTP requests with JSON data (for example a VB.NET client).
Windows PowerShell	Windows PowerShell 5.1 (included in Windows) present and usable by the account that runs the program, with the WMI service running and no Group Policy that blocks the execution of PowerShell scripts. It is essential for encrypting the database password and the WhatsApp session: without PowerShell the key bound to the PC cannot be computed correctly and consistently.

3. Command-line parameters

Parameter	Effect	Starts bridge?
--h / --help	Shows help for the parameters and exits.	No
--version	Shows the banner and version number and exits.	No
--create_config <file>	Creates a sample configuration file, with all parameters commented. Does not overwrite an existing file.	No
--config <file|folder>	Uses the specified file, or Instance_Config.ini in the specified folder.	Yes
--pwd	Asks for the password twice with hidden input and generates PwdConnStr=ENC:...; the same encrypted value is also offered as DbSendSecret=ENC:... (see 15).	No
--pwd=<password>	Directly generates the encrypted PwdConnStr from the specified value.	No
--man_IT [=<file|folder>]	Saves the Italian user manual (.docx) included in the program. Without a path it saves it in the current folder; with a folder it saves it inside that folder; otherwise with the specified name. Default name: Manuale_Albatros_Bridge_<version>.docx. Does not overwrite an existing file.	No
--man_EN [=<file|folder>]	Same as --man_IT, for the English user manual. Default name: Manual_Albatros_Bridge_<version>_EN.docx.	No
--Skip-Manuals	Does not automatically save the user manuals in the program folder at startup (see 8). Equivalent to Skip-Manuals=Y in the configuration file.	Yes
no parameter	Standard startup with automatic search for the configuration.	Yes

Parameters can be given in the form with a space or with '=' where supported. Names are not case-sensitive. Paths enclosed in quotation marks are normalized.

```
Albatros-Bridge-odbc.exe --help
Albatros-Bridge-odbc.exe --version
Albatros-Bridge-odbc.exe --config "C:\Albatros\Instance01"
Albatros-Bridge-odbc.exe --config="C:\Albatros\Instance01\Instance_Config.ini"
Albatros-Bridge-odbc.exe --create_config="C:\tmp\MyConfig.ini"
Albatros-Bridge-odbc.exe --pwd
Albatros-Bridge-odbc.exe --pwd=MiaPassword123
Albatros-Bridge-odbc.exe --man_IT
Albatros-Bridge-odbc.exe --man_EN="C:\Doc\"
Albatros-Bridge-odbc.exe --Skip-Manuals
```

Environment variable: PHONES_BRIDGE_CONFIG can specify the configuration file or folder when --config is not present.

4. Main files and folders

The structure is organized under PathWork. Each instance is kept separate through the Instance_ prefix and the CodeInstance value.

Item	Description
Instance_Config.ini	Instance configuration.
Manuale_Albatros_Bridge_<version>.docx, Manual_Albatros_Bridge_<version>_EN.docx	Italian and English user manuals, saved automatically in the program folder at startup if they do not exist (see 8).
PathWork	Root working folder. Default: c:\Albatros\Phone\.
Instance_<CodeInstance>\	Instance folder.
...\QrCode\latest.png	Current QR code, only when the QR code is actually shown.
...\auth_info\	WhatsApp session credentials, always encrypted and bound to the PC (see 7.1). It must not be used by two instances at the same time.

Item	Description
...\auth_info.undecryptable_<date>\	Previous session that cannot be decrypted on this PC (see 7.1): moved aside, not deleted.
...\pending_db\	Persistent queue of data not yet saved (*_M.json files for messages, *_S.json for messageSecrets).
...\pending_db*.json.tmp	Temporary file used while writing the queue; if it remains after a sudden interruption, it is deleted at startup.
...\pending_db\failed\	Items that exceeded the retry/time limits or that cannot be saved (unreadable file or invalid data).
...\early_queue\	Startup queue: messages and events received before the linked number is available, waiting to be recorded (see 10.6).
...\early_queue\failed\	Startup queue items that are unreadable or cannot be processed (see 10.6).
...\Albatros_Embedded.db	Embedded SQLite database, when used without ConnStr.
PathWork\Instances_Log\<number>\Connection.log	Connection log and copy of the text output.
%TEMP%\AlbatrosBridgeOdbc\	Temporary area for the ODBC embedded in the executable.

5. Configuration file Instance_Config.ini

The file uses one Key=Value or Key:Value pair per line. Lines beginning with # (hash) or ; (semicolon) are comments. Keys are case-insensitive.

Parameter	Default / example	Description
Key	Albatros	Instance label (maximum 50 characters).
TypeDB	InternalSqlite	Database type: MySql, SqlServer, MariaDB, SqLite, InternalSqlite.
ConnStr	empty	Connection string; it can be omitted for InternalSqlite.
PwdConnStr	empty	DB password, in plain text or ENC:.... Ignored with embedded SQLite.
PathWork	c:\Albatros\Phone\	Root working folder.
CodeInstance	Default	Instance code (CodeIstance is also accepted).
PathAuth	automatic	Alternative folder for the WhatsApp credentials.
PathBin	c:\Albatros\Phone\bin\	Folder of the post-insert executable (bin is also accepted).
ExeName	empty	Executable launched after a successful save, with the parameters --rec, --istance and --Cell (see 10.1). Exe is also accepted.
Save_Status	N	Saves delivery/read updates.
Save_StatusBroadcast	N	Saves status@broadcast statuses/stories. With N, their messageSecret, edits and deletions are not saved either.
Save_Sent	S	Saves messages sent by the linked number. With N, their messageSecret and the edits and deletions made by the linked number are not saved either.
Save_protocolMessage	N	Saves technical protocol messages.
Save_Secrets	S	Saves messageSecrets, required for encrypted edits. Those of messages excluded by Save_Sent, Save_StatusBroadcast and Save_protocolMessage are not saved.
ConsoleLog	N	Enables informational/debug messages on screen; banner, summary, QR code, warnings and errors remain visible.
DebugLog	empty	Enables the diagnostic log in hh_Albatros_Webhooks_log for INFO/WARNING/ERROR/DEBUG.
DebugLogFilter	empty	Filters only the media content present in DebugData.
DebugLogMaxRecords	10000	Maximum number of records to keep in the hh_Albatros_Webhooks_log table. Empty value or 0 = no limit.
DbRetrySec	30	Base of the linear backoff toward the DB.
DbRetryMaxSec	300	Maximum cap on the wait between retries (maximum 86400).
DbMaxRetries	0	Maximum attempts per item; 0 = unlimited.
PendingMaxSec	172800	Maximum time in pending; 0 = no time limit.
Port	3000	HTTP port.

Parameter	Default / example	Description
BindAddress	empty	Address on which the HTTP server listens. Empty = all network interfaces (as in previous versions); 127.0.0.1 = this PC only. See 5.3.
ApiToken	empty	Token required by all HTTP endpoints. Empty = no authentication. See 5.3.
MaxMediaMB	100	Maximum size in MB of a media item to download; 0 = no limit. See 5.4.
MaxMemQueue	500	Maximum number of items in the in-memory HTTP queues; 0 disables the queues.
SendMinIntervalSec	3	Minimum seconds between two sends (decimals allowed); the interval is randomly lengthened by up to 30%. 0 = no wait. See 5.5.
SendMaxPerHour	60	Maximum sends in the last hour (sliding window); 0 = no cap. See 5.5.
SendMaxWaitSec	60	Maximum wait of a request in the send queue, then a 429 response (maximum 3600); 0 = no wait. See 5.5.
SendCheckNumber	Y	Y = before sending, checks that the number uses WhatsApp; if it does not, 422 response. See 5.5.
SendIdempotencyHours	24	Hours of validity of clientMsgId (maximum 720); 0 = clientMsgId ignored. See 5.5.
Skip-Manuals	N	Y = at startup, does not automatically save the user manuals in the program folder. Equivalent to the --Skip-Manuals parameter. See 8.
DbSend	N	Y = reads the hh_Albatros_SendQueue table and sends the rows of the linked number, with the same limits as HTTP sends. See 15.
DbSendSecret	empty	Secret for signing the rows (minimum 32 characters), in plain text or ENC:... generated with --pwd. Without a valid secret, DbSend stays disabled. See 15.
DbSendPollSec	5	Seconds between two reads of the table (from 1 to 300). POST /send-queue/poll brings the read forward. See 15.
DbSendMaxAgeMin	60	Minutes of validity of a row from CreatedAtUtc: values below 30 become 30, values above 1440 (24 hours) become 1440. See 15.

Accepted Boolean values: S, Y, SI, SÌ, YES, TRUE, 1 (lowercase too). Any other value, including N or empty, is equivalent to No.

5.1 DebugLog and DebugLogFilter

DebugLog concerns writing to the hh_Albatros_Webhooks_log table. DebugLogFilter does not decide which rows are written: it acts exclusively on the media content present in the DebugData field.

Setting	Result
DebugLog=All	INFO, WARNING, ERROR, DEBUG. All takes precedence over the other values.
DebugLog=Debug, Error	DEBUG and ERROR only.
DebugLog empty / N / NO / FALSE / 0	Diagnostic log in table disabled.
DebugLogFilter=No-Video	Removes videos from DebugData.
No-Audio	Removes audio from DebugData.
No-PDF	Removes PDFs from DebugData.
No-Other	Removes other media from DebugData.
No-Media	Excludes all media from DebugData.

```
DebugLog=All
DebugLogFilter=No-Video, No-Other
ConsoleLog=Y
```

The DEBUG log of a message is written only once, even if saving to the database is retried several times.

5.2 DebugLogMaxRecords

Limits the number of rows kept in hh_Albatros_Webhooks_log. After each successful log write, the oldest rows are deleted in a separate transaction: if the cleanup fails, the row just written remains saved anyway and the cleanup is retried at the next log entry. Default 10000; empty, 0 or invalid value = no limit.

5.3 HTTP interface security: BindAddress and ApiToken

The HTTP endpoints allow sending messages from the linked number, reading received messages and downloading the linking QR code. For this reason it is important to restrict who can reach them.

Scenario	Recommended configuration
Client on the same PC as the bridge	BindAddress=127.0.0.1. In the client use http://127.0.0.1:<Port> (not localhost, which on Windows may be resolved first as the IPv6 address ::1). ApiToken optional.
Client on another PC or on a virtual machine	BindAddress empty (or 0.0.0.0 or the IP address of the network adapter), ApiToken mandatory, and a Windows Firewall rule that opens the port only to the client addresses.

If ApiToken is set, **all** endpoints (including /qr) require the token, sent in one of the two following ways; otherwise the response is HTTP 401. The token value is never shown by /config.

```
Authorization: Bearer <token>
X-Api-Key: <token>
```

Use a long, random token. To generate one with PowerShell:

```
$b = New-Object byte[] 32
[System.Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($b)
[Convert]::ToBase64String($b)
```

Example of a firewall rule that opens port 3000 only to the client PC (sample address):

```
New-NetFirewallRule -DisplayName "Albatros Bridge 3000" -Direction Inbound `
    -Protocol TCP -LocalPort 3000 -RemoteAddress 192.168.1.20 -Action Allow
```

If the server is reachable from the network (BindAddress empty or not local) and ApiToken is not set, the SECURITY warning is shown at startup as the last line of the summary.

5.4 MaxMediaMB

Limits the size of downloaded media. The check is performed before the download, on the size declared by WhatsApp, and again on the actual size. A media item over the limit is not downloaded: the message is saved without the attachment and an error is recorded in the log. Default 100 MB; 0 = no limit. A value that is too high increases memory usage and space used in the disk queue, because the media is kept in pending until it is saved to the database.

5.5 Send limits and checks

The limits reduce the risk of the number being banned for sends that are too fast or too numerous, but they do not make it legitimate to use the program for spam, advertising or bulk messaging (see the initial notices).

Send queue. Requests to /send and /send-media enter a queue and are sent in order of arrival, spaced at least SendMinIntervalSec seconds apart; each interval is randomly lengthened by up to 30% to avoid a fixed rhythm. No more than SendMaxPerHour messages are sent in one hour. The HTTP request waits for its turn and receives the response once the message has been sent. If its turn would come after more than SendMaxWaitSec seconds, the bridge immediately responds 429 with the retryAfterSec field and the Retry-After header, which indicate after how many seconds to retry. SendMaxWaitSec must stay below the timeout of the HTTP client (in .NET the HttpClient default is 100 seconds). If SendMinIntervalSec and SendMaxPerHour are both 0, a SECURITY warning is shown at startup.

Number check. With SendCheckNumber=Y, before queuing the message the bridge checks that the recipient uses WhatsApp; if not, it responds 422 (code NOT_ON_WHATSAPP) and the message does not count toward the hourly quota. The result is remembered for 24 hours if the number exists and for 10 minutes if it does not. Groups (@g.us) and @lid identifiers are not checked. If the check fails for technical reasons (for example a timeout), the message is sent anyway and a warning appears in the log.

Sending without duplicates (clientMsgId). The client can specify in the JSON a unique message identifier, clientMsgId (from 1 to 100 characters), to be reused only when it repeats the same request, for example after a timeout. If the message with that clientMsgId has already been sent successfully within the last SendIdempotencyHours hours, the bridge does not send it again and returns the same response with duplicate=true; if the first send is still in progress, it waits for its result. If the same clientMsgId arrives with a different recipient or content, the response is 409. Failed sends are not remembered, so the request can be repeated with the same clientMsgId. The identifiers are kept in memory and are reset when the program restarts.

/status reports the send status in the send block: messages sent, sends in the last hour, waiting requests, seconds until the next possible send, requests rejected because of a limit or an unregistered number, duplicates.

6. Database configuration

SQL Server, MySQL, MariaDB, SQLite via ODBC and embedded SQLite are supported.

```
TypeDB=InternalSqlite
#ConnStr=
# senza ConnStr:
# PathWork\Instance_<CodeInstance>\Albatros_Embedded.db
```

With InternalSqlite there is no need to install an ODBC driver for the database. If the file is locked by another program, the bridge makes a few quick retries and keeps the data in the queue.

Table	Content
hh_Albatros_Webhooks	Header of messages/events and full RawData.
hh_Albatros_WebHooks_Media	Media attachments associated with the header.
hh_Albatros_WebHooks_Locations	Location (latitude and longitude) of location messages, linked to the header.
hh_Albatros_WebHooks_Secrets	messageSecret of the messages. The linked number and the Key are not stored in this table: they are obtained with a JOIN on hh_Albatros_Webhooks via MsgID.
hh_Albatros_Webhooks_log	Diagnostic log.
hh_Albatros_SendQueue	Messages to send written by another program (sending from the database). Created only with DbSend=Y; see 15.

The tables are created automatically if missing. At startup the program also creates the IX_hh_Albatros_Webhooks_MsgID index on the MsgID column of the header, if it does not already exist, to speed up the search for original messages (required for edits). On a very large table the creation may take a few minutes, during which messages remain queued. If the database user does not have permission to create indexes, only a warning is shown and the program continues normally.

Each message receives a unique identifier (MsgUID) when it enters the queue. If a save has already been completed but the program stopped before deleting the queue file, on the next attempt the row is recognized and is not inserted a second time.

7. Encrypted database password and WhatsApp session

The encrypted password is bound to hardware components of the PC. To generate it:

```
Albatros-Bridge-odbc.exe --pwd
```

The password is requested twice with hidden input when PowerShell is available. At the end a PwdConnStr=ENC:... line is produced, to be copied into the configuration file. A password encrypted on one PC cannot be used on another PC (nor

after replacing the motherboard, CPU or system disk): in these cases it must be regenerated. The same encrypted value can be used for DbSendSecret: the program also offers it in the DbSendSecret=ENC:... line (see 15).

7.1 WhatsApp session encryption

The WhatsApp session files, in the auth_info folder, are always encrypted with a key bound to the hardware components of the PC, different from the one used for the database password. Encryption is automatic, always active and not configurable: it requires no action.

• Anyone who copies the auth_info folder to another PC cannot use it to connect with the WhatsApp number of the instance.

• Upgrading from a previous version: at the first startup, the existing plain-text files are encrypted automatically, without scanning the QR code again.

• Change of PC or hardware (motherboard, CPU, drive C, or a virtual machine moved to a host with a different CPU): the session can no longer be decrypted and must be linked again with a new QR code.

• In this case the folder is not deleted: it is renamed to auth_info.undecryptable_<date and time> and a new session is created. After verifying the new connection, the renamed folder can be deleted; on the phone, under Linked devices, it is advisable to remove the previous device.

• The EncryptAuth parameter is not supported: if present in the configuration file, it is ignored with a warning.

At startup, reading the hardware components may take a few seconds. If the first check fails, the program repeats it before considering the session undecryptable.

Encryption prevents the use of the session on another PC, but it does not replace the protection of the PC itself: anyone who accesses the PC with the permissions of the account that runs the bridge can use the session. Therefore, protect access to the server.

8. Starting the program

Configuration search: --config; PHONES_BRIDGE_CONFIG; Instance_Config.ini in the program folder; legacy path; finally default values with a warning.

At startup, the banner and the configuration summary are always shown. The summary also includes BindAddress, ApiToken (only whether it is set or not, never the value), MaxMediaMB, the send parameters (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec, SendCheckNumber, SendIdempotencyHours), sending from the database (DbSend and, if active, DbSendSecret: only whether it is encrypted or in plain text, never the value) and the Session files line, which confirms that the session is encrypted. The summary ends with DebugLog, DebugLogFilter, the indication by exclusion of the accepted media and, if conditions require it, the SECURITY warning.

```
   MaxMediaMB            : 100 MB
   MaxMemQueue           : 1500
   SendMinIntervalSec    : 3s between two sends (+ up to 30% random)
   SendMaxPerHour        : 60 sends per hour
   SendMaxWaitSec        : 60s maximum wait in the send queue, then HTTP 429
   SendCheckNumber       : Y (recipient verified before sending)
   SendIdempotencyHours  : 24h (repeated clientMsgId is not sent again)
   DbSend                : Y (table hh_Albatros_SendQueue, read every 5s, rows valid 60 minutes)
   DbSendSecret          : set (encrypted, bound to this PC)
   ConsoleLog            : Y (console debug enabled)
   DebugLog              : All (DEBUG, ERROR, INFO, WARNING)
   DebugLogFilter        : NO-VIDEO, NO-OTHER
   Accepted media        : AUDIO, PDF
⚠ SECURITY: the HTTP server is reachable from the network without ApiToken: ...
```

The summary also indicates the startup queue folder (Startup queue folder), described in 10.6.

User manuals: at startup the Italian and English manuals are saved in the program folder, with the same default names as --man_IT and --man_EN, only if they do not already exist: an existing file is never overwritten. For each file created an informational line is shown; if writing is not possible, a warning appears and the program continues. Automatic saving is disabled with the --Skip-Manuals parameter or with Skip-Manuals=Y in the configuration file.

If an unexpected unhandled error occurs, the program writes it to the log (on screen, in Connection.log and in the log table) and exits with exit code 1 after about 2 seconds. For continuous operation it is recommended to run it as a Windows service configured to restart automatically on failure.

9. First connection to WhatsApp

• If no valid credentials exist, the QR code is shown.

• The same QR code is saved in QrCode\latest.png and is also available from the /qr endpoint (with the token, if ApiToken is set).

• On the phone: Settings → Linked devices → Link a device.

• After the scan, WhatsApp requires the connection to be restarted: the bridge reconnects immediately.

• When the connection is open, the QR file is deleted.

• The files of the new session are created already encrypted (see 7.1).

• During an automatic reconnection, the QR code is neither shown nor saved unless it is really needed.

• After a logout, the session is removed and the QR code is requested again.

10. General operation

Each message/event is classified, associated with a previous message if applicable, serialized and placed in the save flow. Each item goes through the persistent queue on disk, and the file is deleted only after it has been successfully saved to the database.

10.1 Retries and queues

• Linear backoff with DbRetrySec and the DbRetryMaxSec cap; any new data triggers an immediate attempt anyway.

• The first successful access to the database resets the attempt counter.

• Queue files are written safely (temporary file and rename): a sudden interruption does not leave incomplete files.

• Database availability errors (network, server down, database busy) are retried according to DbMaxRetries and PendingMaxSec.

• An unreadable queue file or data rejected by the database (for example an invalid value) is moved immediately to pending_db\failed, without blocking the following items.

• Items that exceed the limits are moved to pending_db\failed; the cause is reported on screen and in the log.

• After a message has been saved successfully, ExeName can be launched (see later in this section).

To retry an item that ended up in failed, after fixing the cause, move the file from pending_db\failed to pending_db and restart the bridge: the files in pending_db are loaded at startup.

Post-insert executable. If ExeName is set, after each message saved to the database (not for messageSecrets) the bridge launches PathBin\ExeName without waiting for it to finish, passing three separate parameters:

```
--rec=<ID>          ID della riga in hh_Albatros_Webhooks
--istance=<istanza> codice istanza (CodeInstance) che ha registrato la riga
--Cell=<numero>     numero collegato all'istanza (PhoneConnected), solo cifre
```

Example of the command line received by the launched program:

```
GestioneMessaggi.exe --rec=45 --istance=Magazzino --Cell=393479528620
```

The values are those recorded in the row (ID, Instance and PhoneConnected columns). An instance code that contains spaces still arrives as a single parameter, in quotation marks. If the executable does not exist, a warning is shown only once and saving continues normally; the startup summary shows the parameter format next to Exe. In previous versions the only parameter was the record ID.

10.2 Edited messages

With Save_Secrets=S the messageSecret is captured before the header save filters, except for messages excluded by the configuration (Save_Sent=N, Save_StatusBroadcast=N, Save_protocolMessage=N); edits and deletions of messages excluded by Save_Sent and Save_StatusBroadcast are not saved. The current logic also handles LID-encrypted edits and, when necessary, retrieves the original message by searching for it in the database (ignoring status update rows, which have the same MsgID) or, if not yet saved, in the disk queue.

10.3 RelatedMsgID

Event	Source
Reply / quote	contextInfo.stanzaId
Reaction	reactionMessage.key.id
Protocol	protocolMessage.key.id
Edit	targetMessageKey.id

10.4 Media and locations

• The download of a media item is attempted up to 3 times, with short waits between attempts.

• If the download fails or the media exceeds MaxMediaMB, the message is saved anyway, without the attachment (MediaCount=0); an error is recorded in the log and the item returned by /messages contains the mediaError field with the cause.

• A location with invalid coordinates does not cause the message to be lost: it is saved without the row in hh_Albatros_WebHooks_Locations and with the locationError field; the original coordinates remain in RawData.

• Contacts and polls are recorded with type contact (text = display name) and poll (text = question); the full content remains in RawData.

• In case of other errors during processing, the message is not considered "already seen": a new delivery by WhatsApp is therefore processed normally.

10.5 Automatic reconnection

Cause of the disconnection	Behavior
No network or connection lost	New attempt with increasing wait: about 1, 2, 4, 8 ... seconds, up to a maximum of 60 seconds. The counter is reset when the connection reopens.
Restart required by WhatsApp (after scanning the QR code)	Immediate reconnection.
Session replaced	Another instance is using the same credentials: an error is shown and the next attempt takes place after 60 seconds.
Logout from the phone	The credentials are deleted and a new QR code is requested.

During the wait, /status reports connected=false, the attempt number (reconnectAttempt) and the remaining seconds (reconnectInSec). Send requests received during this phase get an HTTP 503 response.

10.6 Messages received before the linked number is available

At startup, if there are messages waiting on the server, WhatsApp may deliver them before the linked number is available. To avoid recording them with an empty PhoneConnected field, the bridge puts them in the startup queue and records them as soon as the number is known.

• Each event received during this phase (message, edit, deletion, status update) is written to a file in the instance's early_queue folder, in a way protected against sudden interruptions. If the program is closed, at the next startup the existing files are reloaded and wait for the linked number.

• As soon as the number is available, the events are recorded one at a time, from oldest to newest, according to WhatsApp's internal date/time. Within the same second, the message precedes its edit and the edit precedes the deletion; in any case edits, deletions, reactions and status updates are recorded after the message they refer to, so the messageSecret of the original message is already available when the edit is recorded.

• Events that arrive while the queue is being emptied are also queued: no event can be recorded before an older one.

• The file of an event is deleted only after it has been recorded. If the program stops after recording but before deletion, at restart the event is processed again without creating duplicate rows.

• An unreadable file or an event that could not be processed is moved to early_queue\failed; the cause is reported on screen and in the log.

• If the wait exceeds 30 seconds, a warning with the number of queued events is shown, repeated every 60 seconds. /status reports the number of queued events in the startupQueue field.

• For rows recorded from the queue, DateReceived corresponds to the time of recording; the WhatsApp date/time (DateWA) remains the original one.

• If an event stays in the startup queue for a long time, the media may no longer be downloadable because WhatsApp links expire: in this case the message is saved without the attachment, as described in 10.4.

11. HTTP interface (endpoints)

Base address: http://<address>:<Port> — default port 3000. If ApiToken is set, every request must contain the token (see 5.3). Data sent with POST is in JSON format (Content-Type: application/json).

Method	Endpoint	Description
GET	`/status`	Connection status, instance, DB, queue, startup queue (startupQueue), reconnection, sends (send), sending from the database (sendQueue) and statistics.
GET	`/qr`	Current QR code; 404 if not available.
GET	`/messages`	Returns and empties the message queue.
GET	`/messages?peek=1`	Returns the message queue without emptying it; each item has the seq number.
POST	`/messages/ack`	Acknowledges the processed messages: {"upToSeq": N} removes those with seq up to N.
GET	`/message-status`	Returns and empties the status updates.
GET	`/message-status?peek=1`	As above, without emptying (with seq).
POST	`/message-status/ack`	Acknowledges the processed status updates ({"upToSeq": N}).
GET	`/debug-log`	Diagnostic log status.
GET	`/db-queue`	Queue, retries and parameters.
POST	`/db-retry`	Forces an immediate attempt.
GET	`/config`	Active configuration without password or token.
GET	`/message-secret/:msgId`	MessageSecret of the message.
POST	`/send`	Sends text (send limits, number check and optional clientMsgId: see 5.5).
POST	`/send-media`	Sends media via base64 (same checks as /send).

Method	Endpoint	Description
POST	/send-queue/poll	Brings forward the read of the send table (see 15). Response 202; 409 DB_SEND_DISABLED if sending from the database is not active.

11.1 Reading messages without loss (peek / ack)

The classic read of /messages empties the queue at the moment it responds: if the client stops before processing the response, those messages are no longer returned (they remain saved in the database in any case). The two-phase read avoids the problem:

• GET /messages?peek=1: returns the queued messages, each with a sequential seq number, without removing them.

• The client processes the messages.

• POST /messages/ack with {"upToSeq": <seq of the last processed message>}: removes the acknowledged messages from the queue. If the client goes down before acknowledging, it finds them again on the next round.

```
GET  /messages?peek=1
{ "key": "Albatros", "messages": [ { "id": "3EB0...", "type": "text", "text": "ciao", ..., "seq": 41 },
                            { "id": "3EB0...", "type": "image", ..., "seq": 42 } ] }

POST /messages/ack        { "upToSeq": 42 }
{ "success": true, "removed": 2, "remaining": 0 }
```

The queue is in any case limited to MaxMemQueue items: if the client does not acknowledge for a long time, the oldest items are discarded. The same scheme applies to /message-status and /message-status/ack. Do not use the classic read and the two-phase read at the same time from different clients.

11.2 Sending messages

The recipient (number) can be a number with international prefix, also with +, spaces, hyphens or parentheses, or a complete WhatsApp identifier (@s.whatsapp.net, group @g.us, @lid).

```
POST /send
{ "number": "+39 340 123 4567", "message": "Testo del messaggio",
  "clientMsgId": "ORD-2026-000123" }

POST /send-media
{ "number": "393401234567", "mediaType": "document", "fileName": "fattura.pdf",
  "caption": "Fattura di ottobre", "base64Data": "JVBERi0xLjcK..." }

Risposta: { "success": true, "id": "<id del messaggio inviato>" }
Con clientMsgId: { "success": true, "id": "<id>", "clientMsgId": "ORD-2026-000123",
  "duplicate": false }
Errore: { "success": false, "code": "SEND_HOURLY_LIMIT",
  "error": "Hourly send limit reached", "retryAfterSec": 1250 }
```

mediaType	Sent as
image	Image, with optional caption.
audio	Audio (sent as audio/mpeg).
any other value	Document: fileName (default file.pdf) and optional caption.

The clientMsgId field is optional (see 5.5). The error responses of /send and /send-media contain, in addition to error, the code field, which the client can use instead of the text:

code (HTTP)	Meaning
INVALID_NUMBER (400)	number missing or invalid.
INVALID_MESSAGE (400)	message missing or empty (/send).
INVALID_MEDIA (400)	base64Data missing or empty (/send-media).

code (HTTP)	Meaning
INVALID_CLIENT_MSG_ID (400)	Invalid clientMsgId (not text, empty or longer than 100 characters).
CLIENT_MSG_ID_CONFLICT (409)	clientMsgId already used for a different message.
NOT_ON_WHATSAPP (422)	The number does not use WhatsApp (SendCheckNumber=Y). Do not retry.
SEND_HOURLY_LIMIT (429)	SendMaxPerHour reached: retry after retryAfterSec seconds.
SEND_QUEUE_BUSY (429)	Wait in queue longer than SendMaxWaitSec: retry after retryAfterSec seconds.
SEND_ERROR (500)	Error returned by WhatsApp during sending.
NOT_CONNECTED (503)	WhatsApp not connected.

11.3 Response codes

Code	Meaning
200	Request executed.
400	Missing or invalid data (for example invalid number, empty message, missing base64Data, non-numeric upToSeq, malformed JSON).
401	Missing or wrong token (only with ApiToken set).
404	Resource not available (for example no QR code present).
409	clientMsgId already used for a different message (/send, /send-media).
413	Request body too large (over 50 MB).
422	The recipient number does not use WhatsApp (SendCheckNumber=Y).
429	Send limit reached (SendMaxPerHour) or wait in queue longer than SendMaxWaitSec: retry after retryAfterSec seconds (Retry-After header).
500	Internal error or error returned by WhatsApp during sending.
503	WhatsApp not connected or database not available (/message-secret).

11.4 C#.NET client example

```
using System;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;

var http = new HttpClient { BaseAddress = new Uri("http://127.0.0.1:3000/") };
http.DefaultRequestHeaders.Authorization =
    new AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale");

// Lettura senza svuotare la coda
string json = await http.GetStringAsync("messages?peek=1");
// ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

// Conferma dei messaggi elaborati
var ack = new StringContent("{\"upToSeq\":" + ultimoSeq + "}", Encoding.UTF8, "application/json");
HttpResponseMessage r = await http.PostAsync("messages/ack", ack);
```

11.5 VB.NET client example

```
Imports System.Net.Http
Imports System.Net.Http.Headers
Imports System.Text

Dim http As New HttpClient()
http.BaseAddress = New Uri("http://127.0.0.1:3000/")
http.DefaultRequestHeaders.Authorization =
    New AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale")

' Lettura senza svuotare la coda
Dim json As String = Await http.GetStringAsync("messages?peek=1")
' ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

' Conferma dei messaggi elaborati
Dim ack As New StringContent("{""upToSeq"":" & ultimoSeq & "}", Encoding.UTF8, "application/json")
Dim r As HttpResponseMessage = Await http.PostAsync("messages/ack", ack)
```

```
' Invio con clientMsgId: nuovo per ogni messaggio, riutilizzato solo nei ritentativi
Dim idMsg As String = Guid.NewGuid().ToString()
Dim body As String = "{""number"":""393401234567"",""message"":""Testo"",""clientMsgId"":""" & idMsg & """}"
Dim s As HttpResponseMessage = Await http.PostAsync("send", New StringContent(body, Encoding.UTF8,
"application/json"))
Select Case CInt(s.StatusCode)
    Case 429 ' limite raggiunto: attendere e ritentare con lo stesso idMsg
        Dim attesa As TimeSpan = If(s.Headers.RetryAfter?.Delta, TimeSpan.FromSeconds(60))
    Case 422 ' numero non registrato su WhatsApp: non ritentare
End Select
```

12. Connection logs and on-screen log

Connection.log is located in PathWork\Instances_Log\<connected number>\Connection.log. It contains the connection log and a copy of the text output.

Content	Visibility
Banner	Always
Startup summary	Always
SECURITY warning	Always, when the server is reachable from the network without ApiToken
QR code	Always when actually required
Warning / Error	Always
Unhandled errors	Always, before the program closes
INFO / DEBUG	With ConsoleLog=Y
Normal logs	Compact date/time at the beginning

During a prolonged Internet outage, consecutive identical "Connection closed. Reconnecting: true" lines are recorded only once, both on screen and in the log.

13. Troubleshooting

Symptom	Action
Configuration file not found	Use --create_config= or --config=.
DB connection failed	Check TypeDB, ConnStr and PwdConnStr; the data remains in the queue.
ENC password cannot be decrypted	Regenerate it with --pwd on the PC that runs the bridge.
SQLite busy	Wait for the automatic retry; check which program is keeping the file open.
Slow first startup with a large database	Index on MsgID being created: wait; messages remain in the queue.
The client cannot reach the bridge	If the client is on another PC: BindAddress empty or 0.0.0.0, firewall rule for the port, ApiToken. If it is on the same PC: use http://127.0.0.1:<Port>.
HTTP 401 response	Token missing or different from ApiToken: check the Authorization: Bearer or X-Api-Key header.
HTTP 400 response on /send	Check number (number with prefix or JID) and message (not empty).
HTTP 503 response on /send	WhatsApp not connected: check /status (connected, reconnectInSec).
HTTP 429 response on /send	Send limit reached: wait retryAfterSec seconds before retrying. If it happens often, reduce the send rate or adjust SendMinIntervalSec, SendMaxPerHour and SendMaxWaitSec (see 5.5).
HTTP 422 response on /send	The number does not use WhatsApp: check the number and the international prefix. The check is disabled with SendCheckNumber=N.
HTTP 409 response on /send	The same clientMsgId was used for a different message: the client must generate a new clientMsgId for each message and reuse it only in retries.

Symptom	Action
SECURITY warning at startup	Set ApiToken or BindAddress=127.0.0.1.
Message saved without media	See the ERROR log (download failed or MaxMediaMB exceeded) and the mediaError field.
Files in pending_db\failed	Read the cause in the log; after fixing it, move the file to pending_db and restart.
"Connected number still not available" warning	The linked number is not yet available: received messages remain in the startup queue and are recorded as soon as the number is known (see 10.6). If the warning persists, check the connection with /status.
Files in early_queue\failed	Read the cause in the log; after fixing it, move the file to early_queue and restart.
Unwanted manuals in the program folder	Start with --Skip-Manuals or set Skip-Manuals=Y; the files already present can be deleted.
"session replaced" error	Another instance is using the same auth_info folder: leave only one active.
New QR code requested after a change of PC or hardware	Expected behavior: the session is bound to the PC (see 7.1). Scan the new QR code; after verifying, delete the auth_info.undecryptable_<date> folder and remove the old device from Linked devices on the phone.
auth_info folder copied from another PC does not work	Expected behavior: the encrypted files can only be used on the PC that created them. Link the instance with a new QR code.
"EncryptAuth is ignored" warning at startup	The session is always encrypted: remove the EncryptAuth line from the configuration file.
Error at startup about DbSendSecret	The secret is missing, is shorter than 32 characters or cannot be decrypted on this PC: sending from the database stays disabled. Set DbSendSecret or regenerate it with --pwd on the bridge PC.
Send table rows in status K	The writing program uses a secret different from DbSendSecret, or the data was modified after signing. With MySQL, check the utf8mb4 encoding.
Send table rows that remain in status N	Check DbSend=Y, the connection (connected in /status), that the PhoneSender of the rows is the linked number, and the send limits.
Send table rows in status X	Rows written too long before being read, or PC clocks not synchronized: increase DbSendMaxAgeMin (maximum 1440) or synchronize the clocks.
Send table rows in status U	The bridge stopped twice while sending the same row: check in hh_Albatros_Webhooks whether the message was sent and, if necessary, insert it again with a new ClientRef.
The program closes with an error	Read Connection.log; run the bridge as a service with automatic restart.
Console without INFO/DEBUG	Set ConsoleLog=Y.
QR code does not appear	Check whether the session is still valid.
Edit not decrypted	Check Save_Secrets=S and messageSecret.
No log in table	Set DebugLog=All or a list of types.
Unwanted media in DebugData	Configure DebugLogFilter.
WhatsApp number blocked (ban)	The ban is decided and applied by Meta, normally for violation of the WhatsApp rules (spam, advertising, bulk messaging): it must be handled directly with WhatsApp from the app on the phone. See the initial notices.
The program stops connecting after a WhatsApp update	Possible protocol change by WhatsApp (see the initial notices): check whether an updated version of the program is available.
ENC password or session cannot be decrypted without hardware changes	Check that PowerShell is available to the program's account and that the WMI service is running.

14. Best practices

• Use InternalSqlite for test installations when an external DB is not needed or when operating in a protected environment.

• In production, with an external database, prefer PwdConnStr=ENC:... (with InternalSqlite the password is not needed).

• Always set ApiToken when the bridge is reachable from other PCs; use BindAddress=127.0.0.1 when the client is on the same PC.

• In clients, prefer the two-phase read (?peek=1 and /ack) to the classic read.

• Keep Save_Secrets=S if you need edits.

• Monitor /status and /db-queue and periodically check pending_db\failed and early_queue\failed.

• Run the bridge as a Windows service with automatic restart on failure.

• Do not use the program for spam, commercial advertising or bulk messaging: they are prohibited by the WhatsApp rules and can cause the number to be banned.

• Keep the send limits active (SendMinIntervalSec, SendMaxPerHour) and send a unique clientMsgId for each message, reusing it only when the client repeats the same request.

• For sending from the database use an encrypted DbSendSecret (ENC:...), never write the secret to the database, and periodically delete the rows left in states E, K, X and U.

• The auth_info folder is encrypted and bound to the PC: copying it to another PC is useless (a new QR code is needed anyway). Still, restrict access to it to the accounts that run the bridge; restoring from a backup works only on the same PC.

• Use distinct CodeInstance values for multiple WhatsApp numbers; do not use the same auth_info on two instances.

• Enable DebugLog and ConsoleLog during troubleshooting.

• Set DebugLogFilter to limit diagnostic media, and MaxMediaMB according to the available memory.

• Adjust PendingMaxSec and DbMaxRetries to the operating context.

15. Sending from the database (send table)

In addition to the HTTP endpoints, the bridge can send the messages that another program writes to the hh_Albatros_SendQueue table. With DbSend=Y the bridge reads the table every DbSendPollSec seconds and sends the rows of the linked number only, as if they had arrived from /send or /send-media: same queue, same limits (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec) and same number check (SendCheckNumber).

Each row is signed with a secret known only to the bridge and to the writing program: anyone who accesses the database without the secret cannot create valid rows or modify their content.

15.1 Activation

• DbSend=Y enables the feature. The table is created automatically at startup; any creation error is reported but does not block the saving of messages.

• DbSendSecret is the signing secret (minimum 32 characters), in plain text or ENC:... generated with --pwd. Without a valid secret the feature stays disabled and startup reports it with an error. The same secret must be known to the program that writes the rows and must never be written to the database.

• DbSendPollSec (from 1 to 300, default 5): seconds between two reads. The table is read only when the bridge is connected and the linked number is known.

• DbSendMaxAgeMin (default 60): minutes of validity of a row; values below 30 become 30, values above 1440 become 1440.

• POST /send-queue/poll brings the read forward (response 202). Several requests in quick succession produce a single read.

```
DbSend=Y
DbSendSecret=ENC:...
DbSendPollSec=5
DbSendMaxAgeMin=60
```

15.2 Table structure

The columns marked as signed are included in the signature (see 15.4). The writer inserts the row with SendStatus = N; the bridge columns are managed by the bridge.

Column	Written by	Description
ID	database	Sequential identifier; rows are processed in ID order.
DateInsert	database	Insertion date (informational only).
PhoneSender	writer	Signed. Number that must send, digits only: the bridge reads only the rows of its own number.
ClientRef	writer	Signed. Unique reference per PhoneSender (unique index), maximum 100 characters.
Recipient	writer	Signed. Recipient: number with international prefix or JID (groups @g.us).
MsgKind	writer	Signed. Message type (see 15.3).
Title	writer	Signed. Name of the place, name of the contact or poll question.
MsgText	writer	Signed. Text, caption, address, vCard or poll options.
FileName	writer	Signed. Document name.
MimeType	writer	Signed. MIME type of the media; optional.
Latitude, Longitude	writer	Signed. Coordinates as text with a decimal point.
MediaData	writer	Signed through its SHA-256. Attached file in binary (not base64).
CreatedAtUtc	writer	Signed. Creation date in milliseconds UTC since 01/01/1970.
SignVer	writer	Signature format version: 1.
Signature	writer	HMAC-SHA256 signature in hexadecimal (64 characters).
SendStatus	writer / bridge	Row status (see 15.5); the writer inserts N.
ClaimToken, ClaimedBy, ClaimedAtUtc	bridge	Claiming of the row.
Attempts	bridge	Additional attempts already used (maximum 1).
SentMsgID	bridge	Message identifier, chosen before sending.
ErrorCode, ErrorText, ProcessedAtUtc	bridge	Outcome of a row that was not sent.

15.3 Message types (MsgKind)

MsgKind	Columns used	Notes
text	MsgText	Text-only message.
image	MediaData; MsgText (caption), MimeType	Image.
video	MediaData; MsgText (caption), MimeType	Default MIME video/mp4.
gif	MediaData (mp4); MsgText (caption)	Video played as an animated GIF.
audio	MediaData; MimeType	Default MIME audio/mpeg.
voice	MediaData; MimeType	Voice note; default MIME audio/ogg; codecs=opus.
document	MediaData, FileName; MsgText (caption), MimeType	MIME derived from the extension if MimeType is empty.
sticker	MediaData (webp)	Sticker.
location	Latitude, Longitude; Title (name), MsgText (address)	Latitude from -90 to 90, longitude from -180 to 180.
contact	MsgText (one or more vCards); Title (display name)	Each vCard from BEGIN:VCARD to END:VCARD.
poll	Title (question), MsgText (options, one per line)	From 2 to 12 options; single choice.
pollmulti	Title (question), MsgText (options, one per line)	Like poll, with multiple choices.

The media cannot exceed MaxMediaMB. Reactions, edits, deletions and forwards are not supported, because they require an already existing message.

15.4 Signature

Signature is the HMAC-SHA256, keyed with the DbSendSecret secret in UTF-8, of this UTF-8 text:

```
ALB-SEND-1
<lunghezza>:<PhoneSender>
<lunghezza>:<ClientRef>
<lunghezza>:<Recipient>
<lunghezza>:<MsgKind>
<lunghezza>:<Title>
<lunghezza>:<MsgText>
<lunghezza>:<FileName>
<lunghezza>:<MimeType>
<lunghezza>:<Latitude>
<lunghezza>:<Longitude>
<lunghezza>:<MediaSha256>
<lunghezza>:<CreatedAtUtc>
```

• Each line ends with the LF character only (Unix line break); <length> is the number of UTF-8 bytes of the value.

• A NULL value counts as an empty string. MediaSha256 is the lowercase hexadecimal SHA-256 of MediaData, empty if MediaData is NULL or empty. CreatedAtUtc is written in digits.

• Signature is in hexadecimal (uppercase or lowercase). The comparison is performed in constant time.

• The signed values must be identical to those written in the table: with MySQL the connection and the table must use utf8mb4, otherwise emoji and special characters change and the row ends up with an invalid signature.

15.5 Processing and outcomes

The bridge claims each row atomically (ClaimToken), without locks on the table, so two instances connected to the same number cannot send it twice. It then checks the number, signature and time validity, sends it and records the outcome. The row is claimed only if the send queue can send it within SendMaxWaitSec.

SendStatus	Meaning
N	To be sent (written by whoever queues the message).
C	In progress.
(row deleted)	Sent. The message remains in hh_Albatros_Webhooks with Flag_Base = N_TAB; the media is saved without being downloaded again.
E	Permanent error: ErrorCode and ErrorText indicate the reason.
K	Invalid signature: the row is not sent and the log reports a SECURITY error.
X	Expired (older than DbSendMaxAgeMin).
U	Uncertain outcome: the row is no longer sent.

• Bridge not connected or send queue full: the row returns to status N without using up attempts.

• Error during sending: only one additional attempt, with the same message identifier; then status E.

• Claimed without outcome (for example the bridge stopped): after 15 minutes plus SendMaxWaitSec, if the message is already in hh_Albatros_Webhooks (requires Save_Sent=S) the row is deleted; otherwise only one additional attempt is made; the second time the row moves to status U.

• If after sending the database is not reachable, the deletion of the row is repeated at the next read: the row is not sent again.

• Rows in states E, K, X and U remain in the table: they must be read and deleted by the program that wrote them.

ErrorCode	Status	Meaning
INVALID_SIGNATURE	K	Signature does not match: different secret or data modified after signing.
UNSUPPORTED_SIGN_VER	K	Signature version not supported.
EXPIRED	X	Row older than DbSendMaxAgeMin minutes.
CREATED_IN_FUTURE	E	CreatedAtUtc more than 5 minutes in the future (clocks not synchronized).
INVALID_CREATED_AT	E	CreatedAtUtc missing or invalid.
INVALID_RECIPIENT	E	Recipient missing or invalid.
INVALID_KIND	E	MsgKind not supported.
INVALID_MESSAGE	E	Text missing for the text type.
MISSING_MEDIA	E	MediaData missing for a type that requires it.
MISSING_FILE_NAME	E	FileName missing for the document type.
MEDIA_TOO_LARGE	E	MediaData exceeds MaxMediaMB.
MEDIA_READ_ERROR	E	MediaData not readable.
INVALID_LOCATION	E	Coordinates missing or out of range.
INVALID_CONTACT	E	No vCard in MsgText.
INVALID_POLL	E	Question missing or options outside 2-12.
NOT_ON_WHATSAPP	E	The recipient does not use WhatsApp.
SEND_ERROR	E	Send failed even on the additional attempt.
CLAIM_TIMEOUT	N	Claimed without outcome: put back on hold for the additional attempt.
UNCERTAIN	U	Second claim without outcome: the row is not sent again.

15.6 Monitoring

/status reports in the sendQueue block: activation, table, read interval, row validity, read in progress, last error and the counters of rows sent, with invalid signature, expired, in error, with uncertain outcome, retried and put back on hold.

/config reports dbSend, dbSendPollSec, dbSendMaxAgeMin and whether the secret is set (dbSendSecretSet) and encrypted (dbSendSecretEncrypted), never its value. The startup summary shows the DbSend and DbSendSecret lines (see 8).

15.7 Albatros SendQueue program

Albatros SendQueue queues messages in the send table. It consists of a program with a graphical interface (Albatros_SendQueue.exe) and the AlbatrosSendQueueLib.dll library, which contains all the logic: reading Instance_Config.ini, signing, writing to the table, row status and HTTP requests to the bridge.

The library can also be used by other .NET programs (for example a management application): the common parameters (database, DbSendSecret, sender number, bridge address) are set only once, by reading them from Instance_Config.ini or by passing them directly without a configuration file, and for each message type there is a dedicated send function (SendText, SendImage, SendDocument, SendLocation, SendPoll...).

The PhoneSender parameter, which the bridge ignores, is used as the sender number. The program and the library have their own user manual.

Appendix A. Configuration examples

A.1 Test on the same PC with embedded SQLite

```
Key=TEST
TypeDB=InternalSqlite
PathWork=c:\Albatros\Phone\
CodeInstance=Test
Port=3000
BindAddress=127.0.0.1
DebugLog=All
ConsoleLog=Y
```

A.2 Production with SQL Server and client on another PC

```
Key=Albatros
TypeDB=SqlServer
ConnStr=Driver={ODBC Driver 17 for SQL Server};Server=SRVSQL;Database=AlbatrosBridge;UID=albatros_bridge;
PwdConnStr=ENC:....
PathWork=c:\Albatros\Phone\
CodeInstance=393401234567
Port=3000
; BindAddress non indicato: ascolto su tutte le interfacce di rete
ApiToken=IlMioTokenLungoECasuale
MaxMediaMB=100
DebugLog=Error, Warning
DebugLogMaxRecords=10000
PendingMaxSec=172800
```

The values of Server, Database, UID and the token are only examples.

The password must not be specified in the connection string; it goes in PwdConnStr (preferably encrypted with --pwd).

▣ Manuel utilisateur – Albatros Bridge

Version 1.4.129.6121 – Build du 4 octobre 2026

⚠ Avertissements importants

Avertissement	Détail
Aucune association avec Meta	L'auteur n'est en aucune manière associé, affilié, autorisé, sponsorisé ou approuvé par Meta Platforms, Inc., par WhatsApp LLC ou par d'autres sociétés du groupe Meta. WhatsApp est une marque de ses titulaires respectifs et n'est citée que pour décrire la fonction du programme.
Procédure non officielle	La connexion à WhatsApp s'effectue au moyen d'une procédure non officielle (appareil connecté, comme WhatsApp Web, via des bibliothèques open source) et non au moyen des API officielles de WhatsApp Business. Meta, ou l'une des sociétés qui gèrent le produit WhatsApp, peut modifier à tout moment et sans préavis les protocoles, les schémas de connexion, le chiffrement et les règles d'utilisation : le programme peut donc cesser de fonctionner, en tout ou en partie, jusqu'à ce qu'il soit adapté.
Situation commune à tous les produits non officiels	Il existe dans le monde un très grand nombre de sociétés qui réalisent des logiciels ou proposent des services WhatsApp non officiels. Le problème décrit au point précédent est identique pour toutes : aucune d'entre elles ne peut garantir que les modifications décidées par Meta n'interrompront pas le fonctionnement, ni en prévoir le calendrier.
Risques pour le numéro	L'utilisation de clients non officiels peut ne pas être conforme aux conditions d'utilisation de WhatsApp et peut entraîner des limitations ou le blocage du numéro connecté. L'évaluation et le risque correspondant restent à la charge de l'utilisateur du programme.
Utilisation interdite : spam, publicité, envois de masse	L'utilisation du programme pour envoyer du spam, de la publicité commerciale ou des messages de masse est expressément interdite par les règles de WhatsApp. Une utilisation abusive du programme peut donc entraîner le blocage (bannissement) du numéro WhatsApp connecté. Dans ce cas, le blocage n'est en aucune manière imputable à l'auteur : il est appliqué directement par Meta en raison de l'utilisation abusive qui en a été faite.
Utilisation « en l'état »	Le programme est fourni « en l'état » (« as is »), sans garantie d'aucune sorte, expresse ou implicite, y compris les garanties de fonctionnement, d'adéquation à un usage particulier, de continuité du service et d'absence d'erreurs.
Exclusion de responsabilité	Dans les limites maximales autorisées par la loi applicable, aucune responsabilité ne peut être imputée à l'auteur pour des dommages directs ou indirects (perte de données ou de messages, manque à gagner, interruption d'activité, blocage du numéro ou autre) découlant de l'utilisation ou de l'impossibilité d'utiliser le programme, quelle que soit la manière dont on y a eu accès : à titre payant, à titre gracieux, en tant que logiciel gratuit ou de toute autre manière.
Données personnelles	Les messages enregistrés contiennent des données personnelles. La personne qui installe et utilise le programme est responsable de leur traitement, de la sécurité de la base de données et du respect de la réglementation sur la protection des données (RGPD).

```
**********************************
**        Albatros Bridge         **
** rel. 1.4.129.6121 - 04/10/2026 **
**    © Raffaele Romano 2026      **
**********************************
```

Sommaire

1. Présentation du programme

2. Configuration requise

3. Paramètres de ligne de commande

4. Principaux fichiers et dossiers

5. Fichier de configuration Instance_Config.ini

6. Configuration de la base de données

7. Mot de passe de la base de données et session WhatsApp chiffrés

8. Démarrage du programme

9. Première connexion à WhatsApp

10. Fonctionnement général

11. Interface HTTP (points de terminaison)

12. Journal des connexions et journal à l'écran

13. Diagnostic et résolution des problèmes

14. Bonnes pratiques

15. Envoi depuis la base de données (table des envois)

Annexe A. Exemples de configuration

1. ▪ Présentation du programme

Albatros Bridge est une passerelle (bridge) entre WhatsApp et une base de données. Il se connecte à WhatsApp au moyen d'un QR Code, comme WhatsApp Web, et gère la réception, la classification, la persistance et l'exposition HTTP des données.

• Réception : texte, images, vidéos, audio, documents, stickers, positions, statuts, réactions et messages techniques.

• Messages modifiés : gestion des modifications (edit), y compris les nouveaux messages chiffrés LID lorsque le messageSecret nécessaire est disponible.

• Envoi : messages texte et médias via des points de terminaison HTTP, avec contrôle des données reçues.

• Envoi depuis la base de données : un autre programme écrit des lignes signées dans la table hh_Albatros_SendQueue et le bridge les envoie (voir 15).

• Base de données : SQL Server, MySQL, MariaDB, SQLite via ODBC ou SQLite intégré.

• Plusieurs instances : chaque instance utilise son propre dossier et peut avoir sa propre session WhatsApp.

• Files d'attente persistantes : les données restent sur le disque si la base de données n'est pas disponible ; l'écriture sur disque est protégée contre les interruptions soudaines et un enregistrement répété ne crée pas de doublons.

• Ordre chronologique au démarrage : les messages reçus avant que le numéro connecté soit disponible sont mis en file d'attente sur le disque et enregistrés, du plus ancien au plus récent, dès que le numéro est connu.

• Médias fiables : téléchargement automatiquement retenté et limite de taille ; si le média n'est pas disponible, le message est enregistré quand même.

• Sécurité HTTP : adresse d'écoute configurable (BindAddress) et authentification par jeton (ApiToken).

• Session WhatsApp protégée : les fichiers de la session sont toujours chiffrés et liés au PC sur lequel le programme est installé.

• Reconnexion automatique à WhatsApp avec des délais d'attente progressifs.

• Journaux : journal de diagnostic dans une table, journal de connexion et copie de la sortie texte.

• Exécutable post-insertion : facultatif, après un enregistrement réussi.

2. ✓ Configuration requise

Exigence	Détail
Système d'exploitation	Windows 64 bits.
Base de données via ODBC	Pilote ODBC 64 bits installé pour SQL Server, MySQL, MariaDB ou SQLite externe.
SQLite intégré	InternalSqlite : ne nécessite pas de pilote ODBC pour la base de données.
Téléphone	WhatsApp actif et disponible pour scanner le QR Code lors de la première connexion.
Réseau	Accès Internet vers les serveurs WhatsApp. Si le client HTTP se trouve sur un autre PC, le port Port doit être accessible (Pare-feu Windows) et il est conseillé de définir ApiToken.
Client HTTP	Tout programme capable d'envoyer des requêtes HTTP avec des données JSON (par exemple un client VB.NET).

Exigence	Détail
Windows PowerShell	Windows PowerShell 5.1 (inclus dans Windows) présent et utilisable par le compte qui exécute le programme, avec le service WMI actif et sans stratégie de groupe bloquant l'exécution des scripts PowerShell. Il est indispensable pour le chiffrement du mot de passe de la base de données et de la session WhatsApp : sans PowerShell, la clé liée au PC ne peut pas être calculée de manière correcte et stable.

3. Paramètres de ligne de commande

Paramètre	Effet	Démarre le bridge ?
--h / --help	Affiche l'aide des paramètres et quitte.	Non
--version	Affiche la bannière et le numéro de version, puis quitte.	Non
--create_config <file>	Crée un fichier de configuration d'exemple, avec tous les paramètres commentés. N'écrase pas un fichier existant.	Non
--config <file|cartella>	Utilise le fichier indiqué ou Instance_Config.ini dans le dossier indiqué.	Oui
--pwd	Demande le mot de passe deux fois avec saisie masquée et génère PwdConnStr=ENC:... ; la même valeur chiffrée est également proposée comme DbSendSecret=ENC:... (voir 15).	Non
--pwd=<password>	Génère directement PwdConnStr chiffré à partir de la valeur indiquée.	Non
--man_IT [=<file|cartella>]	Enregistre le manuel utilisateur en italien (.docx) inclus dans le programme. Sans chemin, il l'enregistre dans le dossier courant ; avec un dossier, il l'enregistre dans celui-ci ; sinon, sous le nom indiqué. Nom par défaut : Manuale_Albatros_Bridge_<versione>.docx. N'écrase pas un fichier existant.	Non
--man_EN [=<file|cartella>]	Comme --man_IT, pour le manuel utilisateur en anglais. Nom par défaut : Manual_Albatros_Bridge_<versione>_EN.docx.	Non
--Skip-Manuals	N'enregistre pas automatiquement les manuels utilisateur dans le dossier du programme au démarrage (voir 8). Équivaut à Skip-Manuals=Y dans le fichier de configuration.	Oui
aucun paramètre	Démarrage standard avec recherche automatique de la configuration.	Oui

Les paramètres peuvent être indiqués sous la forme avec espace ou avec « = » lorsque celle-ci est prévue. Les noms ne distinguent pas les majuscules et les minuscules. Les chemins entre guillemets sont normalisés.

```
Albatros-Bridge-odbc.exe --help
Albatros-Bridge-odbc.exe --version
Albatros-Bridge-odbc.exe --config "C:\Albatros\Instance01"
Albatros-Bridge-odbc.exe --config="C:\Albatros\Instance01\Instance_Config.ini"
Albatros-Bridge-odbc.exe --create_config="C:\tmp\MyConfig.ini"
Albatros-Bridge-odbc.exe --pwd
Albatros-Bridge-odbc.exe --pwd=MiaPassword123
Albatros-Bridge-odbc.exe --man_IT
Albatros-Bridge-odbc.exe --man_EN="C:\Doc\"
Albatros-Bridge-odbc.exe --Skip-Manuals
```

Variable d'environnement : PHONES_BRIDGE_CONFIG peut indiquer le fichier ou le dossier de configuration lorsque --config n'est pas présent.

4. Principaux fichiers et dossiers

La structure est organisée sous PathWork. Chaque instance est séparée au moyen du préfixe Instance_ et de la valeur CodeInstance.

Élément	Description
Instance_Config.ini	Configuration de l'instance.
Manuale_Albatros_Bridge_<versione>.docx, Manual_Albatros_Bridge_<versione>_EN.docx	Manuels utilisateur en italien et en anglais, enregistrés automatiquement dans le dossier du programme au démarrage s'ils n'existent pas (voir 8).
PathWork	Dossier racine de travail. Par défaut : c:\Albatros\Phone\.
Instance_<CodeInstance>\	Dossier de l'instance.
...\QrCode\latest.png	QR Code actuel, uniquement lorsque le QR Code est effectivement affiché.
...\auth_info\	Identifiants de la session WhatsApp, toujours chiffrés et liés au PC (voir 7.1). Ne doit pas être utilisé par deux instances simultanément.
...\auth_info.undecryptable_<data>\	Session précédente non déchiffrable sur ce PC (voir 7.1) : mise de côté, non supprimée.
...\pending_db\	File d'attente persistante des données pas encore enregistrées (fichiers *_M.json pour les messages, *_S.json pour les messageSecret).
...\pending_db*.json.tmp	Fichier temporaire utilisé pendant l'écriture de la file d'attente ; s'il subsiste après une interruption soudaine, il est supprimé au démarrage.
...\pending_db\failed\	Éléments ayant dépassé les limites de nouvelles tentatives/de temps ou impossibles à enregistrer (fichier illisible ou donnée non valide).
...\early_queue\	File d'attente de démarrage : messages et événements reçus avant que le numéro connecté soit disponible, en attente d'enregistrement (voir 10.6).
...\early_queue\failed\	Éléments de la file d'attente de démarrage illisibles ou impossibles à traiter (voir 10.6).
...\Albatros_Embedded.db	Base de données SQLite intégrée, lorsqu'elle est utilisée sans ConnStr.
PathWork\Instances_Log\<numero>\Connection.log	Journal de la connexion et copie de la sortie texte.
%TEMP%\AlbatrosBridgeOdbc\	Zone temporaire pour l'ODBC intégré dans l'exécutable.

5. Fichier de configuration Instance_Config.ini

Le fichier utilise une paire Clé=Valeur ou Clé:Valeur par ligne. Les lignes qui commencent par # (dièse) ou ; (point-virgule) sont des commentaires. Les clés ne distinguent pas les majuscules et les minuscules.

Paramètre	Par défaut / exemple	Description
Key	Albatros	Libellé de l'instance (50 caractères maximum).
TypeDB	InternalSqlite	Type de base de données : MySql, SqlServer, MariaDB, SqLite, InternalSqlite.
ConnStr	vide	Chaîne de connexion ; peut être omise pour InternalSqlite.
PwdConnStr	vide	Mot de passe de la base de données, en clair ou ENC:.... Ignoré avec SQLite intégré.
PathWork	c:\Albatros\Phone\	Dossier de travail racine.
CodeInstance	Default	Code de l'instance (CodeIstance est également accepté).
PathAuth	automatique	Dossier alternatif pour les identifiants WhatsApp.
PathBin	c:\Albatros\Phone\bin\	Dossier de l'exécutable post-insertion (bin est également accepté).
ExeName	vide	Exécutable lancé après l'enregistrement réussi, avec les paramètres --rec, --istance et --Cell (voir 10.1). Exe est également accepté.
Save_Status	N	Enregistre les mises à jour de remise/lecture.

Paramètre	Par défaut / exemple	Description
Save_StatusBroadcast	N	Enregistre les statuts/stories status@broadcast. Avec N, leur messageSecret, leurs modifications et leurs suppressions ne sont pas enregistrés non plus.
Save_Sent	S	Enregistre les messages envoyés par le numéro connecté. Avec N, leur messageSecret ainsi que les modifications et suppressions effectuées par le numéro connecté ne sont pas enregistrés non plus.
Save_protocolMessage	N	Enregistre les messages techniques de protocole.
Save_Secrets	S	Enregistre les messageSecret, nécessaires pour les modifications chiffrées. Ceux des messages exclus par Save_Sent, Save_StatusBroadcast et Save_protocolMessage ne sont pas enregistrés.
ConsoleLog	N	Active l'affichage à l'écran des messages d'information/de débogage ; la bannière, le récapitulatif, le QR Code, les avertissements et les erreurs restent visibles.
DebugLog	vide	Active le journal de diagnostic dans hh_Albatros_Webhooks_log pour INFO/WARNING/ERROR/DEBUG.
DebugLogFilter	vide	Filtre uniquement le contenu multimédia présent dans DebugData.
DebugLogMaxRecords	10000	Nombre maximal d'enregistrements à conserver dans la table hh_Albatros_Webhooks_log. Valeur vide ou 0 = aucune limite.
DbRetrySec	30	Base du délai d'attente (backoff) linéaire vers la base de données.
DbRetryMaxSec	300	Plafond de l'attente entre deux nouvelles tentatives (86400 maximum).
DbMaxRetries	0	Nombre maximal de tentatives par élément ; 0 = illimité.
PendingMaxSec	172800	Durée maximale en attente (pending) ; 0 = aucune limite de temps.
Port	3000	Port HTTP.
BindAddress	vide	Adresse sur laquelle le serveur HTTP écoute. Vide = toutes les interfaces réseau (comme dans les versions précédentes) ; 127.0.0.1 = ce PC uniquement. Voir 5.3.
ApiToken	vide	Jeton exigé par tous les points de terminaison HTTP. Vide = aucune authentification. Voir 5.3.
MaxMediaMB	100	Taille maximale en Mo d'un média à télécharger ; 0 = aucune limite. Voir 5.4.
MaxMemQueue	500	Nombre maximal d'éléments dans les files d'attente HTTP en mémoire ; 0 désactive les files d'attente.
SendMinIntervalSec	3	Nombre minimal de secondes entre deux envois (décimales admises) ; l'intervalle est allongé aléatoirement jusqu'à 30 %. 0 = aucune attente. Voir 5.5.
SendMaxPerHour	60	Nombre maximal d'envois au cours de la dernière heure (fenêtre glissante) ; 0 = aucun plafond. Voir 5.5.
SendMaxWaitSec	60	Attente maximale d'une requête dans la file d'envoi, après quoi la réponse est 429 (3600 maximum) ; 0 = aucune attente. Voir 5.5.
SendCheckNumber	Y	Y = avant l'envoi, vérifie que le numéro utilise WhatsApp ; si ce n'est pas le cas, réponse 422. Voir 5.5.
SendIdempotencyHours	24	Durée de validité en heures de clientMsgId (720 maximum) ; 0 = clientMsgId ignoré. Voir 5.5.
Skip-Manuals	N	Y = au démarrage, n'enregistre pas automatiquement les manuels utilisateur dans le dossier du programme. Équivaut au paramètre --Skip-Manuals. Voir 8.
DbSend	N	Y = lit la table hh_Albatros_SendQueue et envoie les lignes du numéro connecté, avec les mêmes limites que les envois HTTP. Voir 15.
DbSendSecret	vide	Secret de la signature des lignes (32 caractères minimum), en clair ou ENC:... généré avec --pwd. Sans secret valide, DbSend reste désactivé. Voir 15.
DbSendPollSec	5	Nombre de secondes entre deux lectures de la table (de 1 à 300). POST /send-queue/poll anticipe la lecture. Voir 15.

Paramètre	Par défaut / exemple	Description
DbSendMaxAgeMin	60	Durée de validité en minutes d'une ligne à partir de CreatedAtUtc : les valeurs inférieures à 30 deviennent 30, celles supérieures à 1440 (24 heures) deviennent 1440. Voir 15.

Valeurs booléennes acceptées : S, Y, SI, SÌ, YES, TRUE, 1 (minuscules également). Les autres valeurs, y compris N ou vide, équivalent à Non.

5.1 DebugLog et DebugLogFilter

DebugLog concerne l'écriture dans la table hh_Albatros_Webhooks_log. DebugLogFilter ne détermine pas quelles lignes sont écrites : il agit exclusivement sur le contenu multimédia présent dans le champ DebugData.

Paramétrage	Résultat
DebugLog=All	INFO, WARNING, ERROR, DEBUG. All prévaut sur les autres valeurs.
DebugLog=Debug, Error	Uniquement DEBUG et ERROR.
DebugLog vide / N / NO / FALSE / 0	Journal de diagnostic dans une table désactivé.
DebugLogFilter=No-Video	Supprime les vidéos de DebugData.
No-Audio	Supprime les fichiers audio de DebugData.
No-PDF	Supprime les PDF de DebugData.
No-Other	Supprime les autres médias de DebugData.
No-Media	Exclut tous les médias de DebugData.

```
DebugLog=All
DebugLogFilter=No-Video, No-Other
ConsoleLog=Y
```

Le journal DEBUG d'un message n'est écrit qu'une seule fois, même si l'enregistrement dans la base de données est retenté plusieurs fois.

5.2 DebugLogMaxRecords

Limite le nombre de lignes conservées dans hh_Albatros_Webhooks_log. Après chaque écriture réussie du journal, les lignes les plus anciennes sont supprimées, dans une transaction distincte : si le nettoyage échoue, la ligne qui vient d'être écrite reste tout de même enregistrée et le nettoyage est retenté lors de l'écriture suivante du journal. Par défaut 10000 ; valeur vide, 0 ou non valide = aucune limite.

5.3 Sécurité de l'interface HTTP : BindAddress et ApiToken

Les points de terminaison HTTP permettent d'envoyer des messages depuis le numéro connecté, de lire les messages reçus et de télécharger le QR Code de connexion. C'est pourquoi il est important de limiter qui peut y accéder.

Situation	Configuration conseillée
Client sur le même PC que le bridge	BindAddress=127.0.0.1. Dans le client, utiliser http://127.0.0.1:<Port> (et non localhost, qui sous Windows peut être résolu d'abord comme adresse IPv6 ::1). ApiToken facultatif.
Client sur un autre PC ou sur une machine virtuelle	BindAddress vide (ou 0.0.0.0 ou l'adresse IP de la carte réseau), ApiToken obligatoire et règle du Pare-feu Windows qui n'ouvre le port qu'aux adresses des clients.

Si ApiToken est défini, **tous** les points de terminaison (y compris /qr) exigent le jeton, envoyé de l'une des deux manières suivantes ; sinon, la réponse est HTTP 401. La valeur du jeton n'est jamais affichée par /config.

```
Authorization: Bearer <token>
X-Api-Key: <token>
```

Utilisez un jeton long et aléatoire. Pour en générer un avec PowerShell :

```
$b = New-Object byte[] 32
[System.Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($b)
[Convert]::ToBase64String($b)
```

Exemple de règle de pare-feu qui n'ouvre le port 3000 qu'au PC du client (adresse d'exemple) :

```
New-NetFirewallRule -DisplayName "Albatros Bridge 3000" -Direction Inbound `
    -Protocol TCP -LocalPort 3000 -RemoteAddress 192.168.1.20 -Action Allow
```

Si le serveur est accessible depuis le réseau (BindAddress vide ou non local) et qu'ApiToken n'est pas défini, l'avertissement SECURITY est affiché au démarrage comme dernière ligne du récapitulatif.

5.4 MaxMediaMB

Limite la taille des médias téléchargés. Le contrôle est effectué avant le téléchargement, sur la taille déclarée par WhatsApp, puis de nouveau sur la taille réelle. Un média dépassant la limite n'est pas téléchargé : le message est enregistré sans pièce jointe et une erreur est consignée dans le journal. Par défaut 100 Mo ; 0 = aucune limite. Une valeur trop élevée augmente l'utilisation de la mémoire et de l'espace dans la file d'attente sur disque, car le média est conservé en attente (pending) jusqu'à son enregistrement dans la base de données.

5.5 Limites et contrôles des envois

Les limites réduisent le risque que le numéro soit bloqué en raison d'envois trop rapides ou trop nombreux, mais elles ne rendent pas licite l'utilisation du programme pour du spam, de la publicité ou des envois de masse (voir les avertissements initiaux).

File d'envoi. Les requêtes vers /send et /send-media entrent dans une file d'attente et partent dans l'ordre d'arrivée, espacées d'au moins SendMinIntervalSec secondes ; chaque intervalle est allongé aléatoirement jusqu'à 30 % pour éviter un rythme fixe. Pas plus de SendMaxPerHour messages ne partent en une heure. La requête HTTP attend son tour et reçoit la réponse une fois l'envoi effectué. Si son tour devait arriver après plus de SendMaxWaitSec secondes, le bridge répond immédiatement 429 avec le champ retryAfterSec et l'en-tête Retry-After, qui indiquent après combien de secondes réessayer. SendMaxWaitSec doit rester inférieur au délai d'expiration du client HTTP (en .NET, la valeur par défaut de HttpClient est de 100 secondes). Si SendMinIntervalSec et SendMaxPerHour valent tous deux 0, un avertissement SECURITY est affiché au démarrage.

Vérification du numéro. Avec SendCheckNumber=Y, avant de mettre le message en file d'attente, le bridge vérifie que le destinataire utilise WhatsApp ; si ce n'est pas le cas, il répond 422 (code NOT_ON_WHATSAPP) et le message n'est pas décompté du quota horaire. Le résultat est mémorisé pendant 24 heures si le numéro existe et pendant 10 minutes s'il n'existe pas. Les groupes (@g.us) et les identifiants @lid ne sont pas vérifiés. Si la vérification échoue pour des raisons techniques (par exemple un délai d'expiration), le message est envoyé quand même et un avertissement apparaît dans le journal.

Envoi sans doublons (clientMsgId). Le client peut indiquer dans le JSON un identifiant unique du message, clientMsgId (de 1 à 100 caractères), à réutiliser uniquement lorsqu'il répète la même requête, par exemple après un délai d'expiration. Si le message portant ce clientMsgId a déjà été envoyé avec succès au cours des SendIdempotencyHours dernières heures, le bridge ne l'envoie pas de nouveau et renvoie la même réponse avec duplicate=true ; si le premier envoi est encore en cours, il en attend le résultat. Si le même clientMsgId arrive avec un destinataire ou un contenu différent, la réponse est 409. Les envois qui ont échoué ne sont pas mémorisés : la requête peut donc être répétée avec le même clientMsgId. Les identifiants sont conservés en mémoire et sont remis à zéro au redémarrage du programme.

/status indique l'état des envois dans le bloc send : messages envoyés, envois de la dernière heure, requêtes en attente, secondes avant le prochain envoi possible, requêtes refusées pour cause de limite ou de numéro non enregistré, doublons.

6. Configuration de la base de données

SQL Server, MySQL, MariaDB, SQLite via ODBC et SQLite intégré sont pris en charge.

```
TypeDB=InternalSqlite
#ConnStr=
```

```
# senza ConnStr:
# PathWork\Instance_<CodeInstance>\Albatros_Embedded.db
```

Avec InternalSqlite, il n'est pas nécessaire d'installer un pilote ODBC pour la base de données. Si le fichier est occupé par un autre programme, le bridge effectue quelques nouvelles tentatives rapides et conserve les données en file d'attente.

Table	Contenu
hh_Albatros_Webhooks	En-tête des messages/événements et RawData complet.
hh_Albatros_WebHooks_Media	Pièces jointes multimédias associées à l'en-tête.
hh_Albatros_WebHooks_Locations	Position (latitude et longitude) des messages de type position, liée à l'en-tête.
hh_Albatros_WebHooks_Secrets	messageSecret des messages. Le numéro connecté et la Key ne sont pas stockés dans cette table : ils s'obtiennent par une JOIN sur hh_Albatros_Webhooks via MsgID.
hh_Albatros_Webhooks_log	Journal de diagnostic.
hh_Albatros_SendQueue	Messages à envoyer écrits par un autre programme (envoi depuis la base de données). Créée uniquement avec DbSend=Y ; voir 15.

Les tables sont créées automatiquement si elles sont absentes. Au démarrage, le programme crée également l'index IX_hh_Albatros_Webhooks_MsgID sur la colonne MsgID de l'en-tête, s'il n'existe pas déjà, pour accélérer la recherche des messages d'origine (nécessaire pour les modifications). Sur une table très volumineuse, la création peut prendre quelques minutes, pendant lesquelles les messages restent en file d'attente. Si l'utilisateur de la base de données n'a pas l'autorisation de créer des index, seul un avertissement est affiché et le programme poursuit normalement.

Chaque message reçoit un identifiant unique (MsgUID) au moment où il entre dans la file d'attente. Si un enregistrement a déjà été effectué mais que le programme s'est interrompu avant de supprimer le fichier de file d'attente, lors de la nouvelle tentative la ligne est reconnue et n'est pas insérée une seconde fois.

7. Mot de passe de la base de données et session WhatsApp chiffrés

Le mot de passe chiffré est lié à des éléments matériels du PC. Pour le générer :

```
Albatros-Bridge-odbc.exe --pwd
```

Le mot de passe est demandé deux fois avec saisie masquée lorsque PowerShell est disponible. À la fin, une ligne PwdConnStr=ENC:... est produite, à copier dans le fichier de configuration. Un mot de passe chiffré sur un PC n'est pas utilisable sur un autre PC (ni après le remplacement de la carte mère, du processeur ou du disque système) : dans ces cas, il doit être régénéré. La même valeur chiffrée peut être utilisée pour DbSendSecret : le programme la propose également dans la ligne DbSendSecret=ENC:... (voir 15).

7.1 Chiffrement de la session WhatsApp

Les fichiers de la session WhatsApp, dans le dossier auth_info, sont toujours chiffrés avec une clé liée aux éléments matériels du PC, différente de celle utilisée pour le mot de passe de la base de données. Le chiffrement est automatique, toujours actif et non configurable : il ne nécessite aucune intervention.

• Une personne qui copie le dossier auth_info sur un autre PC ne peut pas l'utiliser pour se connecter avec le numéro WhatsApp de l'instance.

• Mise à jour depuis une version précédente : au premier démarrage, les fichiers existants en clair sont chiffrés automatiquement, sans nouveau scan du QR Code.

• Changement de PC ou de matériel (carte mère, processeur, disque C, ou machine virtuelle déplacée sur un hôte doté d'un processeur différent) : la session n'est plus déchiffrable et doit être reconnectée avec un nouveau QR Code.

• Dans ce cas, le dossier n'est pas supprimé : il est renommé en auth_info.undecryptable_<date et heure> et une nouvelle session est créée. Après avoir vérifié la nouvelle connexion, le dossier renommé peut être supprimé ; sur le téléphone, dans Appareils connectés, il est conseillé de supprimer l'appareil précédent.

• Le paramètre EncryptAuth n'est pas pris en charge : s'il est présent dans le fichier de configuration, il est ignoré avec un avertissement.

Au démarrage, la lecture des éléments matériels peut prendre quelques secondes. Si la première vérification échoue, le programme la répète avant de considérer la session comme non déchiffrable.

Le chiffrement empêche l'utilisation de la session sur un autre PC, mais ne remplace pas la protection du PC lui-même : toute personne qui accède au PC avec les autorisations du compte qui exécute le bridge peut utiliser la session. Protégez donc l'accès au serveur.

8. Démarrage du programme

Recherche de la configuration : --config ; PHONES_BRIDGE_CONFIG ; Instance_Config.ini dans le dossier du programme ; chemin historique ; enfin, valeurs par défaut avec avertissement.

Au démarrage, la bannière et le récapitulatif de la configuration sont toujours affichés ; celui-ci comprend également BindAddress, ApiToken (uniquement s'il est défini ou non, jamais la valeur), MaxMediaMB, les paramètres des envois (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec, SendCheckNumber, SendIdempotencyHours), l'envoi depuis la base de données (DbSend et, s'il est actif, DbSendSecret : uniquement s'il est chiffré ou en clair, jamais la valeur) et la ligne Session files, qui confirme le chiffrement de la session. Le récapitulatif se termine par DebugLog, DebugLogFilter, l'indication par exclusion des médias acceptés et, si les conditions l'exigent, l'avertissement SECURITY.

```
   MaxMediaMB            : 100 MB
   MaxMemQueue           : 1500
   SendMinIntervalSec    : 3s between two sends (+ up to 30% random)
   SendMaxPerHour        : 60 sends per hour
   SendMaxWaitSec        : 60s maximum wait in the send queue, then HTTP 429
   SendCheckNumber       : Y (recipient verified before sending)
   SendIdempotencyHours  : 24h (repeated clientMsgId is not sent again)
   DbSend                : Y (table hh_Albatros_SendQueue, read every 5s, rows valid 60 minutes)
   DbSendSecret          : set (encrypted, bound to this PC)
   ConsoleLog            : Y (console debug enabled)
   DebugLog              : All (DEBUG, ERROR, INFO, WARNING)
   DebugLogFilter        : NO-VIDEO, NO-OTHER
   Accepted media        : AUDIO, PDF
⚠ SECURITY: the HTTP server is reachable from the network without ApiToken: ...
```

Le récapitulatif indique également le dossier de la file d'attente de démarrage (Startup queue folder), décrite en 10.6.

Manuels utilisateur : au démarrage, les manuels en italien et en anglais sont enregistrés dans le dossier du programme, sous les mêmes noms par défaut que --man_IT et --man_EN, uniquement s'ils n'existent pas déjà : un fichier existant n'est jamais écrasé. Pour chaque fichier créé, une ligne d'information est affichée ; si l'écriture est impossible, un avertissement apparaît et le programme poursuit. L'enregistrement automatique se désactive avec le paramètre --Skip-Manuals ou avec Skip-Manuals=Y dans le fichier de configuration.

Si une erreur imprévue non gérée se produit, le programme l'écrit dans le journal (à l'écran, dans Connection.log et dans la table de journal) et se termine avec le code de sortie 1 après environ 2 secondes. Pour un fonctionnement continu, il est conseillé de l'exécuter comme service Windows configuré pour redémarrer automatiquement en cas d'erreur.

9. Première connexion à WhatsApp

• S'il n'existe pas d'identifiants valides, le QR Code est affiché.

• Le même QR Code est enregistré dans QrCode\latest.png et est également disponible depuis le point de terminaison /qr (avec le jeton, si ApiToken est défini).

• Sur le téléphone : Paramètres → Appareils connectés → Connecter un appareil.

• Après le scan, WhatsApp demande un redémarrage de la connexion : le bridge se reconnecte immédiatement.

• Lorsque la connexion est ouverte, le fichier du QR Code est supprimé.

• Les fichiers de la nouvelle session sont créés déjà chiffrés (voir 7.1).

• Lors d'une reconnexion automatique, le QR Code n'est ni affiché ni enregistré s'il n'est pas réellement nécessaire.

• Après une déconnexion (logout), la session est supprimée et le QR Code est de nouveau demandé.

10. Fonctionnement général

Chaque message/événement est classé, éventuellement associé à un message précédent, sérialisé et inséré dans le flux d'enregistrement. Chaque élément passe par la file d'attente persistante sur disque et le fichier n'est supprimé qu'après l'enregistrement réussi dans la base de données.

10.1 Nouvelles tentatives et files d'attente

• Délai d'attente (backoff) linéaire avec DbRetrySec et plafond DbRetryMaxSec ; chaque nouvelle donnée provoque de toute façon une tentative immédiate.

• Le premier accès réussi à la base de données remet à zéro le compteur de tentatives.

• Les fichiers de file d'attente sont écrits de manière sûre (fichier temporaire puis renommage) : une interruption soudaine ne laisse pas de fichiers incomplets.

• Les erreurs de disponibilité de la base de données (réseau, serveur éteint, base de données occupée) sont retentées selon DbMaxRetries et PendingMaxSec.

• Un fichier de file d'attente illisible ou une donnée refusée par la base de données (par exemple une valeur non valide) est immédiatement déplacé dans pending_db\failed, sans bloquer les éléments suivants.

• Les éléments qui dépassent les limites sont déplacés dans pending_db\failed ; la cause est indiquée à l'écran et dans le journal.

• Après l'enregistrement réussi d'un message, ExeName peut être lancé (voir plus loin dans ce paragraphe).

Pour retenter un élément placé dans failed, après en avoir corrigé la cause, déplacez le fichier de pending_db\failed vers pending_db et redémarrez le bridge : les fichiers présents dans pending_db sont chargés au démarrage.

Exécutable post-insertion. Si ExeName est défini, après chaque message enregistré dans la base de données (pas pour les messageSecret), le bridge lance PathBin\ExeName sans attendre sa fin, en lui passant trois paramètres distincts :

```
--rec=<ID>          ID della riga in hh_Albatros_Webhooks
--istance=<istanza> codice istanza (CodeInstance) che ha registrato la riga
--Cell=<numero>     numero collegato all'istanza (PhoneConnected), solo cifre
```

Exemple de ligne de commande reçue par le programme lancé :

```
GestioneMessaggi.exe --rec=45 --istance=Magazzino --Cell=393479528620
```

Les valeurs sont celles enregistrées dans la ligne (colonnes ID, Instance et PhoneConnected). Un code d'instance contenant des espaces arrive tout de même comme paramètre unique, entre guillemets. Si l'exécutable n'existe pas, un avertissement est affiché une seule fois et les enregistrements se poursuivent normalement ; le récapitulatif au démarrage affiche, à côté de Exe, le format des paramètres. Dans les versions précédentes, l'unique paramètre était l'ID de l'enregistrement.

10.2 Messages modifiés

Avec Save_Secrets=S, le messageSecret est capturé avant les filtres d'enregistrement de l'en-tête, sauf pour les messages exclus par la configuration (Save_Sent=N, Save_StatusBroadcast=N, Save_protocolMessage=N) ; les modifications et les suppressions des messages exclus par Save_Sent et Save_StatusBroadcast ne sont pas enregistrées. La logique actuelle gère également les modifications chiffrées LID et, si nécessaire, récupère le message

d'origine en le recherchant dans la base de données (en ignorant les lignes de mise à jour d'état, qui ont le même MsgID) ou, s'il n'est pas encore enregistré, dans la file d'attente sur disque.

10.3 RelatedMsgID

Événement	Origine
Réponse / citation	contextInfo.stanzaId
Réaction	reactionMessage.key.id
Protocole	protocolMessage.key.id
Modification	targetMessageKey.id

10.4 Médias et positions

• Le téléchargement d'un média est tenté jusqu'à 3 fois, avec de courtes attentes entre deux tentatives.

• Si le téléchargement échoue ou si le média dépasse MaxMediaMB, le message est enregistré quand même, sans pièce jointe (MediaCount=0) ; une erreur est consignée dans le journal et l'élément renvoyé par /messages contient le champ mediaError avec la cause.

• Une position dont les coordonnées ne sont pas valides ne fait pas perdre le message : il est enregistré sans la ligne dans hh_Albatros_WebHooks_Locations et avec le champ locationError ; les coordonnées d'origine restent dans RawData.

• Les contacts et les sondages sont enregistrés avec le type contact (texte = nom affiché) et poll (texte = question) ; le contenu complet reste dans RawData.

• En cas d'autres erreurs pendant le traitement, le message n'est pas considéré comme « déjà vu » : une nouvelle remise de la part de WhatsApp est donc traitée normalement.

10.5 Reconnexion automatique

Cause de la fermeture	Comportement
Réseau absent ou connexion perdue	Nouvelle tentative avec une attente croissante : environ 1, 2, 4, 8 ... secondes, jusqu'à 60 secondes maximum. Le compteur est remis à zéro lorsque la connexion se rouvre.
Redémarrage demandé par WhatsApp (après le scan du QR Code)	Reconnexion immédiate.
Session remplacée	Une autre instance utilise les mêmes identifiants : une erreur est affichée et la tentative suivante a lieu après 60 secondes.
Déconnexion depuis le téléphone	Les identifiants sont supprimés et un nouveau QR Code est demandé.

Pendant l'attente, /status indique connected=false, le numéro de la tentative (reconnectAttempt) et les secondes restantes (reconnectInSec). Les requêtes d'envoi reçues pendant cette phase répondent HTTP 503.

10.6 Messages reçus avant que le numéro connecté soit disponible

Au démarrage, s'il y a des messages en attente sur le serveur, WhatsApp peut les remettre avant que le numéro connecté soit disponible. Pour ne pas les enregistrer avec le champ PhoneConnected vide, le bridge les place dans la file d'attente de démarrage et les enregistre dès que le numéro est connu.

• Chaque événement reçu pendant cette phase (message, modification, suppression, mise à jour d'état) est écrit dans un fichier du dossier early_queue de l'instance, de manière protégée contre les interruptions soudaines. Si le programme est fermé, au démarrage suivant les fichiers présents sont rechargés et restent en attente du numéro connecté.

• Dès que le numéro est disponible, les événements sont enregistrés un par un, du plus ancien au plus récent, selon la date/heure interne de WhatsApp. Dans la même seconde, le message précède sa modification et la modification précède la suppression ; dans tous les cas, les modifications, suppressions, réactions et mises à jour d'état sont enregistrées après

le message auquel elles se rapportent, de sorte que le messageSecret du message d'origine est déjà disponible lors de l'enregistrement de la modification.

• Les événements qui arrivent pendant que la file d'attente est vidée sont eux aussi mis en file d'attente : aucun événement ne peut être enregistré avant un événement plus ancien.

• Le fichier d'un événement n'est supprimé qu'après l'enregistrement. Si le programme s'interrompt après l'enregistrement mais avant la suppression, au redémarrage l'événement est traité de nouveau sans créer de lignes en double.

• Un fichier illisible ou un événement qui n'a pas pu être traité est déplacé dans early_queue\failed ; la cause est indiquée à l'écran et dans le journal.

• Si l'attente dépasse 30 secondes, un avertissement indiquant le nombre d'événements en file d'attente est affiché, puis répété toutes les 60 secondes. /status indique le nombre d'événements en file d'attente dans le champ startupQueue.

• Pour les lignes enregistrées à partir de la file d'attente, DateReceived correspond au moment de l'enregistrement ; la date/heure de WhatsApp (DateWA) reste celle d'origine.

• Si un événement reste longtemps dans la file d'attente de démarrage, le média risque de ne plus pouvoir être téléchargé, car les liens de WhatsApp expirent : dans ce cas, le message est enregistré sans pièce jointe, comme décrit en 10.4.

11. Interface HTTP (points de terminaison)

Adresse de base : http://<adresse>:<Port> — port par défaut 3000. Si ApiToken est défini, chaque requête doit contenir le jeton (voir 5.3). Les données envoyées avec POST sont au format JSON (Content-Type: application/json).

Méthode	Point de terminaison	Description
GET	`/status`	État de la connexion, instance, base de données, file d'attente, file d'attente de démarrage (startupQueue), reconnexion, envois (send), envoi depuis la base de données (sendQueue) et statistiques.
GET	`/qr`	QR Code actuel ; 404 s'il n'est pas disponible.
GET	`/messages`	Renvoie et vide la file d'attente des messages.
GET	`/messages?peek=1`	Renvoie la file d'attente des messages sans la vider ; chaque élément porte le numéro seq.
POST	`/messages/ack`	Confirme les messages traités : {"upToSeq": N} supprime ceux dont le seq va jusqu'à N.
GET	`/message-status`	Renvoie et vide les mises à jour d'état.
GET	`/message-status?peek=1`	Comme ci-dessus, sans vider (avec seq).
POST	`/message-status/ack`	Confirme les mises à jour d'état traitées ({"upToSeq": N}).
GET	`/debug-log`	État du journal de diagnostic.
GET	`/db-queue`	File d'attente, nouvelles tentatives et paramètres.
POST	`/db-retry`	Force une tentative immédiate.
GET	`/config`	Configuration active sans mot de passe ni jeton.
GET	`/message-secret/:msgId`	MessageSecret du message.
POST	`/send`	Envoie du texte (limites d'envoi, vérification du numéro et clientMsgId facultatif : voir 5.5).
POST	`/send-media`	Envoie un média en base64 (mêmes contrôles que /send).
POST	`/send-queue/poll`	Anticipe la lecture de la table des envois (voir 15). Réponse 202 ; 409 DB_SEND_DISABLED si l'envoi depuis la base de données n'est pas actif.

11.1 Lecture des messages sans perte (peek / ack)

La lecture classique de /messages vide la file d'attente au moment où elle répond : si le client s'interrompt avant d'avoir traité la réponse, ces messages ne sont plus renvoyés (ils restent néanmoins enregistrés dans la base de données). La lecture en deux phases évite ce problème :

• GET /messages?peek=1 : renvoie les messages en file d'attente, chacun avec un numéro progressif seq, sans les supprimer.

• Le client traite les messages.

• POST /messages/ack avec {"upToSeq": <seq du dernier message traité>} : supprime de la file d'attente les messages confirmés. Si le client tombe avant la confirmation, il les retrouve au cycle suivant.

```
GET  /messages?peek=1
{ "key": "Albatros", "messages": [ { "id": "3EB0...", "type": "text", "text": "ciao", ..., "seq": 41 },
                                   { "id": "3EB0...", "type": "image", ..., "seq": 42 } ] }

POST /messages/ack        { "upToSeq": 42 }
{ "success": true, "removed": 2, "remaining": 0 }
```

La file d'attente reste néanmoins limitée à MaxMemQueue éléments : si le client ne confirme pas pendant longtemps, les éléments les plus anciens sont écartés. Le même schéma s'applique à /message-status et /message-status/ack. N'utilisez pas simultanément la lecture classique et la lecture en deux phases depuis des clients différents.

11.2 Envoi de messages

Le destinataire (number) peut être un numéro avec indicatif international, même avec +, espaces, tirets ou parenthèses, ou un identifiant WhatsApp complet (@s.whatsapp.net, groupe @g.us, @lid).

```
POST /send
{ "number": "+39 340 123 4567", "message": "Testo del messaggio",
  "clientMsgId": "ORD-2026-000123" }

POST /send-media
{ "number": "393401234567", "mediaType": "document", "fileName": "fattura.pdf",
  "caption": "Fattura di ottobre", "base64Data": "JVBERi0xLjcK..." }

Risposta: { "success": true, "id": "<id del messaggio inviato>" }
Con clientMsgId: { "success": true, "id": "<id>", "clientMsgId": "ORD-2026-000123",
  "duplicate": false }
Errore: { "success": false, "code": "SEND_HOURLY_LIMIT",
  "error": "Hourly send limit reached", "retryAfterSec": 1250 }
```

mediaType	Envoi
image	Image, avec légende (caption) éventuelle.
audio	Audio (envoyé en tant que audio/mpeg).
toute autre valeur	Document : fileName (par défaut file.pdf) et légende (caption) éventuelle.

Le champ clientMsgId est facultatif (voir 5.5). Les réponses d'erreur de /send et /send-media contiennent, en plus de error, le champ code, que le client peut utiliser à la place du texte :

code (HTTP)	Signification
INVALID_NUMBER (400)	number absent ou non valide.
INVALID_MESSAGE (400)	message absent ou vide (/send).
INVALID_MEDIA (400)	base64Data absent ou vide (/send-media).
INVALID_CLIENT_MSG_ID (400)	clientMsgId non valide (pas du texte, vide ou de plus de 100 caractères).
CLIENT_MSG_ID_CONFLICT (409)	clientMsgId déjà utilisé pour un message différent.
NOT_ON_WHATSAPP (422)	Le numéro n'utilise pas WhatsApp (SendCheckNumber=Y). Ne pas réessayer.
SEND_HOURLY_LIMIT (429)	SendMaxPerHour atteint : réessayer après retryAfterSec secondes.
SEND_QUEUE_BUSY (429)	Attente en file d'attente supérieure à SendMaxWaitSec : réessayer après retryAfterSec secondes.
SEND_ERROR (500)	Erreur renvoyée par WhatsApp pendant l'envoi.
NOT_CONNECTED (503)	WhatsApp non connecté.

11.3 Codes de réponse

Code	Signification
200	Requête exécutée.
400	Données absentes ou non valides (par exemple number non valide, message vide, base64Data absent, upToSeq non numérique, JSON mal formé).
401	Jeton absent ou erroné (uniquement si ApiToken est défini).
404	Ressource non disponible (par exemple QR Code absent).
409	clientMsgId déjà utilisé pour un message différent (/send, /send-media).
413	Corps de la requête trop volumineux (plus de 50 Mo).
422	Le numéro destinataire n'utilise pas WhatsApp (SendCheckNumber=Y).
429	Limite d'envoi atteinte (SendMaxPerHour) ou attente en file d'attente supérieure à SendMaxWaitSec : réessayer après retryAfterSec secondes (en-tête Retry-After).
500	Erreur interne ou erreur renvoyée par WhatsApp pendant l'envoi.
503	WhatsApp non connecté ou base de données non disponible (/message-secret).

11.4 Exemple de client C#.NET

```
using System;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;

var http = new HttpClient { BaseAddress = new Uri("http://127.0.0.1:3000/") };
http.DefaultRequestHeaders.Authorization =
    new AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale");

// Lettura senza svuotare la coda
string json = await http.GetStringAsync("messages?peek=1");
// ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

// Conferma dei messaggi elaborati
var ack = new StringContent("{\"upToSeq\":" + ultimoSeq + "}", Encoding.UTF8, "application/json");
HttpResponseMessage r = await http.PostAsync("messages/ack", ack);
```

11.5 Exemple de client VB.NET

```
Imports System.Net.Http
Imports System.Net.Http.Headers
Imports System.Text

Dim http As New HttpClient()
http.BaseAddress = New Uri("http://127.0.0.1:3000/")
http.DefaultRequestHeaders.Authorization =
    New AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale")

' Lettura senza svuotare la coda
Dim json As String = Await http.GetStringAsync("messages?peek=1")
' ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

' Conferma dei messaggi elaborati
Dim ack As New StringContent("{""upToSeq"":" & ultimoSeq & "}", Encoding.UTF8, "application/json")
Dim r As HttpResponseMessage = Await http.PostAsync("messages/ack", ack)

' Invio con clientMsgId: nuovo per ogni messaggio, riutilizzato solo nei ritentativi
Dim idMsg As String = Guid.NewGuid().ToString()
Dim body As String = "{""number"":""393401234567"",""message"":""Testo"",""clientMsgId"":""" & idMsg & """}"
Dim s As HttpResponseMessage = Await http.PostAsync("send", New StringContent(body, Encoding.UTF8,
"application/json"))
Select Case CInt(s.StatusCode)
    Case 429 ' limite raggiunto: attendere e ritentare con lo stesso idMsg
        Dim attesa As TimeSpan = If(s.Headers.RetryAfter?.Delta, TimeSpan.FromSeconds(60))
    Case 422 ' numero non registrato su WhatsApp: non ritentare
End Select
```

12. Journal des connexions et journal à l'écran

Connection.log se trouve dans PathWork\Instances_Log\<numéro connecté>\Connection.log. Il contient le journal de connexion et une copie de la sortie texte.

Contenu	Visibilité
Bannière	Toujours
Récapitulatif de démarrage	Toujours
Avertissement SECURITY	Toujours, lorsque le serveur est accessible depuis le réseau sans ApiToken
QR Code	Toujours, lorsqu'il est réellement demandé
Warning / Error	Toujours
Erreurs non gérées	Toujours, avant la fermeture du programme
INFO / DEBUG	Avec ConsoleLog=Y
Journaux normaux	Date/heure compacte au début

Lors d'une perte prolongée de la connexion Internet, les lignes identiques consécutives "Connection closed. Reconnecting: true" ne sont enregistrées qu'une seule fois, à l'écran comme dans le journal.

13. Diagnostic et résolution des problèmes

Symptôme	Action
Fichier de configuration introuvable	Utiliser --create_config= ou --config=.
Échec de la connexion à la base de données	Vérifier TypeDB, ConnStr et PwdConnStr ; les données restent en file d'attente.
Mot de passe ENC non déchiffrable	Régénérer avec --pwd sur le PC qui exécute le bridge.
SQLite occupé	Attendre la nouvelle tentative automatique ; vérifier quel programme maintient le fichier ouvert.
Premier démarrage lent avec une base de données volumineuse	Création de l'index sur MsgID en cours : patienter ; les messages restent en file d'attente.
Le client n'atteint pas le bridge	Si le client se trouve sur un autre PC : BindAddress vide ou 0.0.0.0, règle de pare-feu pour le port, ApiToken. S'il se trouve sur le même PC : utiliser http://127.0.0.1:<Port>.
Réponse HTTP 401	Jeton absent ou différent d'ApiToken : vérifier l'en-tête Authorization: Bearer ou X-Api-Key.
Réponse HTTP 400 sur /send	Contrôler number (numéro avec indicatif ou JID) et message (non vide).
Réponse HTTP 503 sur /send	WhatsApp non connecté : contrôler /status (connected, reconnectInSec).
Réponse HTTP 429 sur /send	Limite d'envoi atteinte : attendre retryAfterSec secondes avant de réessayer. Si cela se produit souvent, réduire le rythme des envois ou ajuster SendMinIntervalSec, SendMaxPerHour et SendMaxWaitSec (voir 5.5).
Réponse HTTP 422 sur /send	Le numéro n'utilise pas WhatsApp : contrôler le numéro et l'indicatif international. La vérification se désactive avec SendCheckNumber=N.
Réponse HTTP 409 sur /send	Le même clientMsgId a été utilisé pour un message différent : le client doit générer un nouveau clientMsgId pour chaque message et ne le réutiliser que lors des nouvelles tentatives.
Avertissement SECURITY au démarrage	Définir ApiToken ou BindAddress=127.0.0.1.
Message enregistré sans média	Consulter le journal ERROR (échec du téléchargement ou MaxMediaMB dépassé) et le champ mediaError.
Fichiers dans pending_db\failed	Lire la cause dans le journal ; après correction, déplacer le fichier dans pending_db et redémarrer.
Avertissement "Connected number still not available"	Le numéro connecté n'est pas encore disponible : les messages reçus restent dans la file d'attente de démarrage et sont enregistrés dès que le numéro est connu (voir 10.6). Si l'avertissement persiste, contrôler la connexion avec /status.

Symptôme	Action
Fichiers dans early_queue\failed	Lire la cause dans le journal ; après correction, déplacer le fichier dans early_queue et redémarrer.
Manuels non souhaités dans le dossier du programme	Démarrer avec --Skip-Manuals ou définir Skip-Manuals=Y ; les fichiers déjà présents peuvent être supprimés.
Erreur "session replaced"	Une autre instance utilise le même dossier auth_info : n'en laisser qu'une seule active.
Nouveau QR Code demandé après un changement de PC ou de matériel	Comportement prévu : la session est liée au PC (voir 7.1). Scanner le nouveau QR Code ; après vérification, supprimer le dossier auth_info.undecryptable_<date> et supprimer l'ancien appareil dans Appareils connectés sur le téléphone.
Le dossier auth_info copié depuis un autre PC ne fonctionne pas	Comportement prévu : les fichiers chiffrés ne sont utilisables que sur le PC qui les a créés. Connecter l'instance avec un nouveau QR Code.
Avertissement "EncryptAuth is ignored" au démarrage	La session est toujours chiffrée : supprimer la ligne EncryptAuth du fichier de configuration.
Erreur au démarrage concernant DbSendSecret	Le secret est absent, fait moins de 32 caractères ou n'est pas déchiffrable sur ce PC : l'envoi depuis la base de données reste désactivé. Définir DbSendSecret ou le régénérer avec --pwd sur le PC du bridge.
Lignes de la table des envois à l'état K	Le programme qui écrit utilise un secret différent de DbSendSecret, ou les données ont été modifiées après la signature. Avec MySQL, vérifier l'encodage utf8mb4.
Lignes de la table des envois qui restent à l'état N	Vérifier DbSend=Y, la connexion (connected dans /status), que le PhoneSender des lignes correspond bien au numéro connecté, ainsi que les limites d'envoi.
Lignes de la table des envois à l'état X	Lignes écrites trop longtemps avant la lecture ou horloges des PC non synchronisées : augmenter DbSendMaxAgeMin (1440 maximum) ou synchroniser les horloges.
Lignes de la table des envois à l'état U	Le bridge s'est arrêté deux fois pendant l'envoi de la même ligne : vérifier dans hh_Albatros_Webhooks si le message est parti et, si nécessaire, le réinsérer avec un nouveau ClientRef.
Le programme se ferme avec une erreur	Lire Connection.log ; exécuter le bridge comme service avec redémarrage automatique.
Console sans INFO/DEBUG	Définir ConsoleLog=Y.
Le QR Code n'apparaît pas	Vérifier si la session est encore valide.
Modification non déchiffrée	Vérifier Save_Secrets=S et messageSecret.
Journal dans la table absent	Définir DebugLog=All ou une liste de types.
Médias indésirables dans DebugData	Configurer DebugLogFilter.
Numéro WhatsApp bloqué (bannissement)	Le blocage est décidé et appliqué par Meta, en règle générale pour violation des règles de WhatsApp (spam, publicité, envois de masse) : il doit être traité directement avec WhatsApp depuis l'application sur le téléphone. Voir les avertissements initiaux.
Le programme ne parvient plus à se connecter après une mise à jour de WhatsApp	Modification possible du protocole par WhatsApp (voir les avertissements initiaux) : vérifier la disponibilité d'une version mise à jour du programme.
Mot de passe ENC ou session non déchiffrables sans changement de matériel	Vérifier que PowerShell est disponible pour le compte du programme et que le service WMI est actif.

14. Bonnes pratiques

• Utilisez InternalSqlite pour les installations de test lorsqu'une base de données externe n'est pas nécessaire ou que vous travaillez dans un environnement protégé.

• En production, avec une base de données externe, privilégiez PwdConnStr=ENC:... (avec InternalSqlite, le mot de passe n'est pas nécessaire).

• Définissez toujours ApiToken lorsque le bridge est accessible depuis d'autres PC ; utilisez BindAddress=127.0.0.1 lorsque le client se trouve sur le même PC.

• Dans les clients, privilégiez la lecture en deux phases (?peek=1 et /ack) à la lecture classique.

• Conservez Save_Secrets=S si les modifications sont nécessaires.

• Surveillez /status et /db-queue et contrôlez régulièrement pending_db\failed et early_queue\failed.

• Exécutez le bridge comme service Windows avec redémarrage automatique en cas d'erreur.

• N'utilisez pas le programme pour du spam, de la publicité commerciale ou des envois de masse : ils sont interdits par les règles de WhatsApp et peuvent entraîner le blocage du numéro.

• Laissez actives les limites d'envoi (SendMinIntervalSec, SendMaxPerHour) et envoyez un clientMsgId unique pour chaque message, en ne le réutilisant que lorsque le client répète la même requête.

• Pour l'envoi depuis la base de données, utilisez un DbSendSecret chiffré (ENC:...), n'écrivez jamais le secret dans la base de données et supprimez régulièrement les lignes restées aux états E, K, X et U.

• Le dossier auth_info est chiffré et lié au PC : le copier sur un autre PC ne sert à rien (un nouveau QR Code est de toute façon nécessaire). Limitez néanmoins son accès aux comptes qui exécutent le bridge ; la restauration depuis une sauvegarde ne fonctionne que sur le même PC.

• Utilisez des CodeInstance distincts pour plusieurs numéros WhatsApp ; n'utilisez pas le même auth_info sur deux instances.

• Activez DebugLog et ConsoleLog pendant le diagnostic.

• Définissez DebugLogFilter pour limiter les médias de diagnostic, et MaxMediaMB en fonction de la mémoire disponible.

• Adaptez PendingMaxSec et DbMaxRetries au contexte d'exploitation.

15. Envoi depuis la base de données (table des envois)

Outre les points de terminaison HTTP, le bridge peut envoyer les messages qu'un autre programme écrit dans la table hh_Albatros_SendQueue. Avec DbSend=Y, le bridge lit la table toutes les DbSendPollSec secondes et envoie uniquement les lignes du numéro connecté, comme si elles provenaient de /send ou de /send-media : même file d'attente, mêmes limites (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec) et même vérification du numéro (SendCheckNumber).

Chaque ligne est signée avec un secret connu uniquement du bridge et du programme qui écrit : une personne qui accède à la base de données sans le secret ne peut ni créer de lignes valides ni en modifier le contenu.

15.1 Activation

• DbSend=Y active la fonction. La table est créée automatiquement au démarrage ; une éventuelle erreur de création est signalée mais ne bloque pas l'enregistrement des messages.

• DbSendSecret est le secret de la signature (32 caractères minimum), en clair ou ENC:... généré avec --pwd. Sans secret valide, la fonction reste désactivée et le démarrage le signale par une erreur. Le même secret doit être connu du programme qui écrit les lignes et ne doit jamais être écrit dans la base de données.

• DbSendPollSec (de 1 à 300, par défaut 5) : secondes entre deux lectures. La table n'est lue que lorsque le bridge est connecté et que le numéro connecté est connu.

• DbSendMaxAgeMin (par défaut 60) : durée de validité d'une ligne en minutes ; les valeurs inférieures à 30 deviennent 30, celles supérieures à 1440 deviennent 1440.

• POST /send-queue/poll anticipe la lecture (réponse 202). Plusieurs requêtes rapprochées ne produisent qu'une seule lecture.

```
DbSend=Y
DbSendSecret=ENC:...
DbSendPollSec=5
DbSendMaxAgeMin=60
```

15.2 Structure de la table

Les colonnes indiquées comme signées entrent dans la signature (voir 15.4). Le programme qui écrit insère la ligne avec SendStatus = N ; les colonnes du bridge sont gérées par le bridge.

Colonne	Écrite par	Description
ID	base de données	Identifiant progressif ; les lignes sont traitées dans l'ordre des ID.
DateInsert	base de données	Date d'insertion (à titre informatif uniquement).
PhoneSender	programme qui écrit	Signée. Numéro qui doit envoyer, chiffres uniquement : le bridge ne lit que les lignes de son propre numéro.
ClientRef	programme qui écrit	Signée. Référence unique par PhoneSender (index unique), 100 caractères maximum.
Recipient	programme qui écrit	Signée. Destinataire : numéro avec indicatif international ou JID (groupes @g.us).
MsgKind	programme qui écrit	Signée. Type de message (voir 15.3).
Title	programme qui écrit	Signée. Nom du lieu, nom du contact ou question du sondage.
MsgText	programme qui écrit	Signée. Texte, légende, adresse, vCard ou options du sondage.
FileName	programme qui écrit	Signée. Nom du document.
MimeType	programme qui écrit	Signée. Type MIME du média ; facultatif.
Latitude, Longitude	programme qui écrit	Signées. Coordonnées sous forme de texte avec le point comme séparateur décimal.
MediaData	programme qui écrit	Signée au moyen de son SHA-256. Fichier joint en binaire (pas en base64).
CreatedAtUtc	programme qui écrit	Signée. Date de création en millisecondes UTC depuis le 01/01/1970.
SignVer	programme qui écrit	Version du format de la signature : 1.
Signature	programme qui écrit	Signature HMAC-SHA256 en hexadécimal (64 caractères).
SendStatus	programme qui écrit / bridge	État de la ligne (voir 15.5) ; le programme qui écrit insère N.
ClaimToken, ClaimedBy, ClaimedAtUtc	bridge	Prise en charge de la ligne.
Attempts	bridge	Tentatives supplémentaires déjà utilisées (1 maximum).
SentMsgID	bridge	Identifiant du message, choisi avant l'envoi.
ErrorCode, ErrorText, ProcessedAtUtc	bridge	Résultat d'une ligne non envoyée.

15.3 Types de message (MsgKind)

MsgKind	Colonnes utilisées	Remarques
text	MsgText	Message texte uniquement.
image	MediaData ; MsgText (légende), MimeType	Image.
video	MediaData ; MsgText (légende), MimeType	MIME par défaut video/mp4.
gif	MediaData (mp4) ; MsgText (légende)	Vidéo lue comme GIF animé.
audio	MediaData ; MimeType	MIME par défaut audio/mpeg.

MsgKind	Colonnes utilisées	Remarques
voice	MediaData ; MimeType	Note vocale ; MIME par défaut audio/ogg; codecs=opus.
document	MediaData, FileName ; MsgText (légende), MimeType	MIME déduit de l'extension si MimeType est vide.
sticker	MediaData (webp)	Sticker.
location	Latitude, Longitude ; Title (nom), MsgText (adresse)	Latitude de -90 à 90, longitude de -180 à 180.
contact	MsgText (une ou plusieurs vCard) ; Title (nom affiché)	Chaque vCard va de BEGIN:VCARD à END:VCARD.
poll	Title (question), MsgText (options, une par ligne)	De 2 à 12 options ; un seul choix.
pollmulti	Title (question), MsgText (options, une par ligne)	Comme poll, avec choix multiples.

Le média ne peut pas dépasser MaxMediaMB. Les réactions, modifications, suppressions et transferts ne sont pas gérés, car ils nécessitent un message déjà existant.

15.4 Signature

Signature est le HMAC-SHA256, avec pour clé le secret DbSendSecret en UTF-8, du texte UTF-8 suivant :

```
ALB-SEND-1
<lunghezza>:<PhoneSender>
<lunghezza>:<ClientRef>
<lunghezza>:<Recipient>
<lunghezza>:<MsgKind>
<lunghezza>:<Title>
<lunghezza>:<MsgText>
<lunghezza>:<FileName>
<lunghezza>:<MimeType>
<lunghezza>:<Latitude>
<lunghezza>:<Longitude>
<lunghezza>:<MediaSha256>
<lunghezza>:<CreatedAtUtc>
```

• Chaque ligne se termine uniquement par le caractère LF (saut de ligne Unix) ; <lunghezza> est la longueur de la valeur en octets UTF-8.

• Une valeur NULL vaut une chaîne vide. MediaSha256 est le SHA-256 hexadécimal en minuscules de MediaData, vide si MediaData est NULL ou vide. CreatedAtUtc est écrit en chiffres.

• Signature est en hexadécimal (majuscules ou minuscules). La comparaison s'effectue en temps constant.

• Les valeurs signées doivent être identiques à celles écrites dans la table : avec MySQL, la connexion et la table doivent utiliser utf8mb4, sinon les emoji et les caractères spéciaux sont altérés et la signature de la ligne est considérée comme non valide.

15.5 Traitement et résultats

Le bridge prend en charge chaque ligne de manière atomique (ClaimToken), sans verrou sur la table, de sorte que deux instances connectées au même numéro ne peuvent pas l'envoyer deux fois. Il contrôle ensuite le numéro, la signature et la validité dans le temps, envoie le message et enregistre le résultat. La ligne n'est prise en charge que si la file d'envoi peut la faire partir dans le délai SendMaxWaitSec.

SendStatus	Signification
N	À envoyer (écrite par le programme qui met le message en file d'attente).
C	En cours de traitement.
(ligne supprimée)	Envoyée. Le message reste dans hh_Albatros_Webhooks avec Flag_Base = N_TAB ; le média est enregistré sans être téléchargé de nouveau.
E	Erreur définitive : ErrorCode et ErrorText indiquent le motif.

SendStatus	Signification
K	Signature non valide : la ligne n'est pas envoyée et le journal indique une erreur SECURITY.
X	Expirée (plus ancienne que DbSendMaxAgeMin).
U	Résultat incertain : la ligne n'est plus envoyée.

• Bridge non connecté ou file d'envoi pleine : la ligne revient à l'état N sans consommer de tentative.

• Erreur pendant l'envoi : une seule tentative supplémentaire, avec le même identifiant de message ; ensuite état E.

• Prise en charge sans résultat (par exemple arrêt du bridge) : après 15 minutes plus SendMaxWaitSec, si le message figure déjà dans hh_Albatros_Webhooks (nécessite Save_Sent=S), la ligne est supprimée ; sinon, une seule tentative supplémentaire est effectuée ; la deuxième fois, la ligne passe à l'état U.

• Si, après l'envoi, la base de données n'est pas accessible, la suppression de la ligne est répétée lors de la lecture suivante : la ligne n'est pas renvoyée.

• Les lignes aux états E, K, X et U restent dans la table : elles doivent être lues et supprimées par le programme qui les a écrites.

ErrorCode	État	Signification
INVALID_SIGNATURE	K	Signature non concordante : secret différent ou données modifiées après la signature.
UNSUPPORTED_SIGN_VER	K	Version de la signature non gérée.
EXPIRED	X	Ligne plus ancienne que DbSendMaxAgeMin minutes.
CREATED_IN_FUTURE	E	CreatedAtUtc situé plus de 5 minutes dans le futur (horloges non synchronisées).
INVALID_CREATED_AT	E	CreatedAtUtc absent ou non valide.
INVALID_RECIPIENT	E	Destinataire absent ou non valide.
INVALID_KIND	E	MsgKind non géré.
INVALID_MESSAGE	E	Texte absent pour le type text.
MISSING_MEDIA	E	MediaData absent pour un type qui l'exige.
MISSING_FILE_NAME	E	FileName absent pour le type document.
MEDIA_TOO_LARGE	E	MediaData dépasse MaxMediaMB.
MEDIA_READ_ERROR	E	MediaData illisible.
INVALID_LOCATION	E	Coordonnées absentes ou hors plage.
INVALID_CONTACT	E	Aucune vCard dans MsgText.
INVALID_POLL	E	Question absente ou nombre d'options hors de la plage 2-12.
NOT_ON_WHATSAPP	E	Le destinataire n'utilise pas WhatsApp.
SEND_ERROR	E	Échec de l'envoi, y compris lors de la tentative supplémentaire.
CLAIM_TIMEOUT	N	Prise en charge sans résultat : remise en attente pour la tentative supplémentaire.
UNCERTAIN	U	Deuxième prise en charge sans résultat : la ligne n'est pas renvoyée.

15.6 Contrôle

/status indique dans le bloc sendQueue : l'activation, la table, l'intervalle de lecture, la validité des lignes, la lecture en cours, la dernière erreur et les compteurs des lignes envoyées, à signature non valide, expirées, en erreur, à résultat incertain, retentées et remises en attente.

/config indique dbSend, dbSendPollSec, dbSendMaxAgeMin et si le secret est défini (dbSendSecretSet) et chiffré (dbSendSecretEncrypted), jamais sa valeur. Le récapitulatif au démarrage affiche les lignes DbSend et DbSendSecret (voir 8).

15.7 Programme Albatros SendQueue

Albatros SendQueue met des messages en file d'attente dans la table des envois. Il se compose d'un programme doté d'une interface graphique (Albatros_SendQueue.exe) et de la bibliothèque AlbatrosSendQueueLib.dll, qui contient toute la logique : lecture d'Instance_Config.ini, signature, écriture dans la table, état de la ligne et requêtes HTTP vers le bridge.

La bibliothèque peut également être utilisée par d'autres programmes .NET (par exemple un logiciel de gestion) : les paramètres communs (base de données, DbSendSecret, numéro expéditeur, adresse du bridge) se définissent une seule fois, en les lisant dans Instance_Config.ini ou en les passant directement sans fichier de configuration, et il existe une fonction d'envoi dédiée pour chaque type de message (SendText, SendImage, SendDocument, SendLocation, SendPoll...).

Comme numéro expéditeur, on utilise le paramètre PhoneSender, que le bridge ignore. Le programme et la bibliothèque disposent chacun de leur propre manuel utilisateur.

Annexe A. Exemples de configuration

A.1 Test sur le même PC avec SQLite intégré

```
Key=TEST
TypeDB=InternalSqlite
PathWork=c:\Albatros\Phone\
CodeInstance=Test
Port=3000
BindAddress=127.0.0.1
DebugLog=All
ConsoleLog=Y
```

A.2 Production avec SQL Server et client sur un autre PC

```
Key=Albatros
TypeDB=SqlServer
ConnStr=Driver={ODBC Driver 17 for SQL Server};Server=SRVSQL;Database=AlbatrosBridge;UID=albatros_bridge;
PwdConnStr=ENC:....
PathWork=c:\Albatros\Phone\
CodeInstance=393401234567
Port=3000
; BindAddress non indicato: ascolto su tutte le interfacce di rete
ApiToken=IlMioTokenLungoECasuale
MaxMediaMB=100
DebugLog=Error, Warning
DebugLogMaxRecords=10000
PendingMaxSec=172800
```

Les valeurs de Server, Database, UID et du jeton ne sont que des exemples.

Le mot de passe ne doit pas être indiqué dans la chaîne de connexion : il se saisit dans PwdConnStr (de préférence chiffré avec --pwd).

▣ Manuale Utente – Albatros Bridge

Versione 1.4.129.6121 – Build 4 ottobre 2026

⚠ Avvertenze importanti

Avvertenza	Dettaglio
Nessuna associazione con Meta	L'autore non è in alcun modo associato, affiliato, autorizzato, sponsorizzato o approvato da Meta Platforms, Inc., da WhatsApp LLC o da altre società del gruppo Meta. WhatsApp è un marchio dei rispettivi titolari ed è citato solo per descrivere la funzione del programma.
Procedura non ufficiale	Il collegamento a WhatsApp avviene con una procedura non ufficiale (dispositivo collegato, come WhatsApp Web, tramite librerie open source) e non con le API ufficiali di WhatsApp Business. Meta, o una delle società che gestiscono il prodotto WhatsApp, può cambiare in qualsiasi momento e senza preavviso protocolli, schemi di connessione, cifratura e regole d'uso: il programma può quindi smettere di funzionare, in tutto o in parte, finché non viene adeguato.
Situazione comune a tutti i prodotti non ufficiali	Nel mondo esistono moltissime società che realizzano software o offrono servizi WhatsApp non ufficiali. Il problema descritto al punto precedente è identico per tutte: nessuna di esse può garantire che le modifiche decise da Meta non interrompano il funzionamento, né prevederne i tempi.
Rischi per il numero	L'uso di client non ufficiali può non essere conforme alle condizioni d'uso di WhatsApp e può comportare limitazioni o il blocco del numero collegato. La valutazione e il relativo rischio restano a carico di chi usa il programma.
Uso vietato: spam, pubblicità, invii massivi	L'uso del programma per inviare spam, pubblicità commerciale o messaggi massivi è espressamente vietato dalle regole di WhatsApp. Un uso improprio del programma può quindi portare al blocco (ban) del numero WhatsApp collegato. In questo caso il blocco non è in alcun modo riferibile all'autore: è applicato direttamente da Meta a causa dell'uso improprio che ne è stato fatto.
Uso «così com'è»	Il programma è fornito «così com'è» («as is»), senza garanzie di alcun tipo, espresse o implicite, comprese quelle di funzionamento, idoneità a uno scopo specifico, continuità del servizio e assenza di errori.
Esclusione di responsabilità	Nei limiti massimi consentiti dalla legge applicabile, nessuna responsabilità è imputabile all'autore per danni diretti o indiretti (perdita di dati o messaggi, mancato guadagno, interruzione dell'attività, blocco del numero o altro) derivanti dall'uso o dall'impossibilità di usare il programma, qualunque sia la forma con cui se ne è ottenuto l'accesso: a pagamento, in omaggio, come software gratuito o in qualsiasi altro modo.
Dati personali	I messaggi salvati contengono dati personali. Chi installa e usa il programma è responsabile del loro trattamento, della sicurezza del database e del rispetto della normativa sulla protezione dei dati (GDPR).

```
**********************************
**        Albatros Bridge        **
** rel. 1.4.129.6121 - 04/10/2026 **
**    © Raffaele Romano 2026     **
**********************************
```

Indice

1. Cos'è il programma

2. Requisiti

3. Parametri da riga di comando

4. File e cartelle principali

5. File di configurazione Instance_Config.ini

6. Configurazione del database

7. Password del database e sessione WhatsApp cifrate

8. Avvio del programma

9. Primo collegamento a WhatsApp

10. Funzionamento generale

11. Interfaccia HTTP (endpoint)

12. Log delle connessioni e log a video

13. Diagnosi e risoluzione problemi

14. Buone pratiche

15. Invio da database (tabella degli invii)

Appendice A. Esempi di configurazione

1. ▪ Cos'è il programma

Albatros Bridge è un bridge tra WhatsApp e un database. Si collega a WhatsApp tramite QR Code, come WhatsApp Web, e gestisce la ricezione, la classificazione, la persistenza e l'esposizione HTTP dei dati.

• Ricezione: testo, immagini, video, audio, documenti, sticker, posizioni, stati, reazioni e messaggi tecnici.

• Messaggi modificati: gestione degli edit, inclusi i nuovi messaggi cifrati LID quando il messageSecret necessario è disponibile.

• Invio: messaggi di testo e media tramite endpoint HTTP, con controllo dei dati ricevuti.

• Invio da database: un altro programma scrive righe firmate nella tabella hh_Albatros_SendQueue e il bridge le invia (vedere 15).

• Database: SQL Server, MySQL, MariaDB, SQLite via ODBC oppure SQLite incorporato.

• Più istanze: ogni istanza usa una propria cartella e può avere una propria sessione WhatsApp.

• Code persistenti: i dati restano su disco se il database non è disponibile; la scrittura su disco è protetta da interruzioni improvvise e un salvataggio ripetuto non crea doppioni.

• Ordine cronologico all'avvio: i messaggi ricevuti prima che il numero collegato sia disponibile vengono accodati su disco e registrati, dal più vecchio al più nuovo, appena il numero è noto.

• Media affidabili: download ritentato automaticamente e limite di dimensione; se il media non è disponibile il messaggio viene salvato comunque.

• Sicurezza HTTP: indirizzo di ascolto configurabile (BindAddress) e autenticazione con token (ApiToken).

• Sessione WhatsApp protetta: i file della sessione sono sempre cifrati e legati al PC su cui è installato il programma.

• Riconnessione automatica a WhatsApp con attese progressive.

• Log: log diagnostico su tabella, log connessione e copia dell'output testuale.

• Eseguibile post-inserimento: opzionale, dopo un salvataggio riuscito.

2. ✓ Requisiti

Requisito	Dettaglio
Sistema operativo	Windows 64 bit.
Database via ODBC	Driver ODBC a 64 bit installato per SQL Server, MySQL, MariaDB o SQLite esterno.
SQLite incorporato	InternalSqlite: non richiede un driver ODBC per il database.
Telefono	WhatsApp attivo e disponibile per la scansione del QR al primo collegamento.
Rete	Accesso a Internet verso i server WhatsApp. Se il client HTTP è su un altro PC, la porta Port deve essere raggiungibile (firewall di Windows) ed è consigliato impostare ApiToken.
Client HTTP	Qualsiasi programma in grado di inviare richieste HTTP con dati JSON (ad esempio un client VB.NET).
Windows PowerShell	Windows PowerShell 5.1 (incluso in Windows) presente e utilizzabile dall'account che esegue il programma, con il servizio WMI attivo e senza criteri di gruppo che blocchino l'esecuzione degli script PowerShell. È indispensabile per la cifratura della password del database e della sessione WhatsApp: senza PowerShell la chiave legata al PC non può essere calcolata in modo corretto e stabile.

3. Parametri da riga di comando

Parametro	Effetto	Avvia bridge?
--h / --help	Mostra l'help dei parametri e termina.	No
--version	Mostra banner e numero di versione e termina.	No
--create_config <file>	Crea un file di configurazione di esempio, con tutti i parametri commentati. Non sovrascrive un file esistente.	No
--config <file|cartella>	Usa il file indicato oppure Instance_Config.ini nella cartella indicata.	Sì
--pwd	Chiede la password due volte con input nascosto e genera PwdConnStr=ENC:...; lo stesso valore cifrato viene proposto anche come DbSendSecret=ENC:... (vedere 15).	No
--pwd=<password>	Genera direttamente PwdConnStr cifrata dal valore indicato.	No
--man_IT [=<file|cartella>]	Salva il manuale utente in italiano (.docx) incluso nel programma. Senza percorso lo salva nella cartella corrente; con una cartella lo salva al suo interno; altrimenti con il nome indicato. Nome predefinito: Manuale_Albatros_Bridge_<versione>.docx. Non sovrascrive un file esistente.	No
--man_EN [=<file|cartella>]	Come --man_IT, per il manuale utente in inglese. Nome predefinito: Manual_Albatros_Bridge_<versione>_EN.docx.	No
--Skip-Manuals	Non salva automaticamente i manuali utente nella cartella del programma all'avvio (vedere 8). Equivale a Skip-Manuals=Y nel file di configurazione.	Sì
nessun parametro	Avvio standard con ricerca automatica della configurazione.	Sì

I parametri possono essere indicati nella forma con spazio oppure con '=' quando prevista. I nomi non distinguono maiuscole/minuscole. I percorsi racchiusi tra virgolette vengono normalizzati.

```
Albatros-Bridge-odbc.exe --help
Albatros-Bridge-odbc.exe --version
Albatros-Bridge-odbc.exe --config "C:\Albatros\Instance01"
Albatros-Bridge-odbc.exe --config="C:\Albatros\Instance01\Instance_Config.ini"
Albatros-Bridge-odbc.exe --create_config="C:\tmp\MyConfig.ini"
Albatros-Bridge-odbc.exe --pwd
Albatros-Bridge-odbc.exe --pwd=MiaPassword123
Albatros-Bridge-odbc.exe --man_IT
Albatros-Bridge-odbc.exe --man_EN="C:\Doc\"
Albatros-Bridge-odbc.exe --Skip-Manuals
```

Variabile d'ambiente: PHONES_BRIDGE_CONFIG può indicare il file o la cartella di configurazione quando --config non è presente.

4. File e cartelle principali

La struttura viene organizzata sotto PathWork. Ogni istanza è separata tramite il prefisso Instance_ e il valore CodeInstance.

Elemento	Descrizione
Instance_Config.ini	Configurazione dell'istanza.
Manuale_Albatros_Bridge_<versione>.docx, Manual_Albatros_Bridge_<versione>_EN.docx	Manuali utente in italiano e in inglese, salvati automaticamente nella cartella del programma all'avvio se non esistono (vedere 8).
PathWork	Cartella radice del lavoro. Default: c:\Albatros\Phone\.
Instance_<CodeInstance>\	Cartella dell'istanza.
...\QrCode\latest.png	QR Code corrente, solo quando il QR è effettivamente mostrato.

Elemento	Descrizione
...\auth_info\	Credenziali della sessione WhatsApp, sempre cifrate e legate al PC (vedere 7.1). Non deve essere usata da due istanze contemporaneamente.
...\auth_info.undecryptable_<data>\	Sessione precedente non decifrabile su questo PC (vedere 7.1): spostata da parte, non cancellata.
...\pending_db\	Coda persistente dei dati non ancora salvati (file *_M.json per i messaggi, *_S.json per i messageSecret).
...\pending_db*.json.tmp	File temporaneo usato durante la scrittura della coda; se resta dopo un'interruzione improvvisa viene eliminato all'avvio.
...\pending_db\failed\	Elementi che hanno superato i limiti di retry/tempo oppure non salvabili (file illeggibile o dato non valido).
...\early_queue\	Coda di avvio: messaggi ed eventi ricevuti prima che il numero collegato sia disponibile, in attesa di essere registrati (vedere 10.6).
...\early_queue\failed\	Elementi della coda di avvio illeggibili o non elaborabili (vedere 10.6).
...\Albatros_Embedded.db	Database SQLite incorporato, quando usato senza ConnStr.
PathWork\Instances_Log\<numero>\Connection.log	Log della connessione e copia dell'output testuale.
%TEMP%\AlbatrosBridgeOdbc\	Area temporanea per l'ODBC incorporato nell'eseguibile.

5. File di configurazione Instance_Config.ini

Il file usa una coppia Chiave=Valore oppure Chiave:Valore per riga. Le righe che iniziano con # (cancelletto) o ; (punto e virgola) sono commenti. Le chiavi sono case-insensitive.

Parametro	Default / esempio	Descrizione
Key	Albatros	Etichetta dell'istanza (massimo 50 caratteri).
TypeDB	InternalSqlite	Tipo database: MySql, SqlServer, MariaDB, SqLite, InternalSqlite.
ConnStr	vuota	Connection string; per InternalSqlite può essere omessa.
PwdConnStr	vuota	Password DB, in chiaro oppure ENC:.... Ignorata con SQLite incorporato.
PathWork	c:\Albatros\Phone\	Cartella di lavoro radice.
CodeInstance	Default	Codice dell'istanza (accettato anche CodeIstance).
PathAuth	automatico	Cartella alternativa per le credenziali WhatsApp.
PathBin	c:\Albatros\Phone\bin\	Cartella dell'eseguibile post-inserimento (accettato anche bin).
ExeName	vuoto	Eseguibile lanciato dopo il salvataggio riuscito, con i parametri --rec, --istance e --Cell (vedere 10.1). Accettato anche Exe.
Save_Status	N	Salva aggiornamenti di consegna/lettura.
Save_StatusBroadcast	N	Salva stati/storie status@broadcast. Con N non vengono salvati neppure il loro messageSecret, le modifiche e le cancellazioni.
Save_Sent	S	Salva i messaggi inviati dal numero collegato. Con N non vengono salvati neppure il loro messageSecret e le modifiche e cancellazioni fatte dal numero collegato.
Save_protocolMessage	N	Salva i messaggi tecnici di protocollo.
Save_Secrets	S	Salva i messageSecret, necessari per gli edit cifrati. Non vengono salvati quelli dei messaggi esclusi da Save_Sent, Save_StatusBroadcast e Save_protocolMessage.
ConsoleLog	N	Abilita a video i messaggi informativi/debug; banner, riepilogo, QR, warning ed errori restano visibili.
DebugLog	vuoto	Abilita il log diagnostico su hh_Albatros_Webhooks_log per INFO/WARNING/ERROR/DEBUG.
DebugLogFilter	vuoto	Filtra solo il contenuto multimediale presente in DebugData.
DebugLogMaxRecords	10000	Numero massimo di record da mantenere nella tabella hh_Albatros_Webhooks_log. Valore vuoto oppure 0 = nessun limite.
DbRetrySec	30	Base del backoff lineare verso il DB.
DbRetryMaxSec	300	Tetto massimo dell'attesa tra retry (massimo 86400).

Parametro	Default / esempio	Descrizione
DbMaxRetries	0	Tentativi massimi per elemento; 0 = illimitati.
PendingMaxSec	172800	Tempo massimo in pending; 0 = nessun limite temporale.
Port	3000	Porta HTTP.
BindAddress	vuoto	Indirizzo su cui ascolta il server HTTP. Vuoto = tutte le interfacce di rete (come nelle versioni precedenti); 127.0.0.1 = solo questo PC. Vedere 5.3.
ApiToken	vuoto	Token richiesto da tutti gli endpoint HTTP. Vuoto = nessuna autenticazione. Vedere 5.3.
MaxMediaMB	100	Dimensione massima in MB di un media da scaricare; 0 = nessun limite. Vedere 5.4.
MaxMemQueue	500	Numero massimo di elementi nelle code HTTP in memoria; 0 disattiva le code.
SendMinIntervalSec	3	Secondi minimi tra due invii (anche decimali); l'intervallo viene allungato a caso fino al 30%. 0 = nessuna attesa. Vedere 5.5.
SendMaxPerHour	60	Invii massimi nell'ultima ora (finestra mobile); 0 = nessun tetto. Vedere 5.5.
SendMaxWaitSec	60	Attesa massima di una richiesta nella coda di invio, poi risposta 429 (massimo 3600); 0 = nessuna attesa. Vedere 5.5.
SendCheckNumber	Y	Y = prima di inviare verifica che il numero usi WhatsApp; se non lo usa, risposta 422. Vedere 5.5.
SendIdempotencyHours	24	Ore di validità di clientMsgId (massimo 720); 0 = clientMsgId ignorato. Vedere 5.5.
Skip-Manuals	N	Y = all'avvio non salva automaticamente i manuali utente nella cartella del programma. Equivale al parametro --Skip-Manuals. Vedere 8.
DbSend	N	Y = legge la tabella hh_Albatros_SendQueue e invia le righe del numero collegato, con gli stessi limiti degli invii HTTP. Vedere 15.
DbSendSecret	vuoto	Segreto della firma delle righe (minimo 32 caratteri), in chiaro oppure ENC:... generato con --pwd. Senza un segreto valido DbSend resta disattivato. Vedere 15.
DbSendPollSec	5	Secondi tra due letture della tabella (da 1 a 300). POST /send-queue/poll anticipa la lettura. Vedere 15.
DbSendMaxAgeMin	60	Minuti di validità di una riga da CreatedAtUtc: valori sotto 30 diventano 30, oltre 1440 (24 ore) diventano 1440. Vedere 15.

Valori booleani accettati: S, Y, SI, SÌ, YES, TRUE, 1 (anche minuscoli). Gli altri valori, compreso N o vuoto, equivalgono a No.

5.1 DebugLog e DebugLogFilter

DebugLog riguarda la scrittura nella tabella hh_Albatros_Webhooks_log. DebugLogFilter non decide quali righe vengono scritte: agisce esclusivamente sul contenuto multimediale presente nel campo DebugData.

Impostazione	Risultato
DebugLog=All	INFO, WARNING, ERROR, DEBUG. All prevale sugli altri valori.
DebugLog=Debug, Error	Solo DEBUG ed ERROR.
DebugLog vuoto / N / NO / FALSE / 0	Log diagnostico su tabella disattivato.
DebugLogFilter=No-Video	Rimuove i video da DebugData.
No-Audio	Rimuove gli audio da DebugData.
No-PDF	Rimuove i PDF da DebugData.
No-Other	Rimuove gli altri media da DebugData.
No-Media	Esclude tutti i media da DebugData.

```
DebugLog=All
DebugLogFilter=No-Video, No-Other
ConsoleLog=Y
```

Il log DEBUG di un messaggio viene scritto una sola volta, anche se il salvataggio nel database viene ritentato più volte.

5.2 DebugLogMaxRecords

Limita il numero di righe conservate in hh_Albatros_Webhooks_log. Dopo ogni scrittura riuscita del log vengono eliminate le righe più vecchie, in una transazione separata: se la pulizia non riesce, la riga appena scritta resta comunque salvata e la pulizia viene ritentata al log successivo. Default 10000; valore vuoto, 0 o non valido = nessun limite.

5.3 Sicurezza dell'interfaccia HTTP: BindAddress e ApiToken

Gli endpoint HTTP permettono di inviare messaggi dal numero collegato, leggere i messaggi ricevuti e scaricare il QR di collegamento. Per questo è importante limitare chi può raggiungerli.

Situazione	Configurazione consigliata
Client sullo stesso PC del bridge	BindAddress=127.0.0.1. Nel client usare http://127.0.0.1:<Port> (non localhost, che su Windows può essere risolto prima come indirizzo IPv6 ::1). ApiToken facoltativo.
Client su un altro PC o su una macchina virtuale	BindAddress vuoto (oppure 0.0.0.0 o l'indirizzo IP della scheda di rete), ApiToken obbligatorio e regola del firewall di Windows che apre la porta solo agli indirizzi dei client.

Se ApiToken è impostato, **tutti** gli endpoint (compreso /qr) richiedono il token, inviato in uno dei due modi seguenti; in caso contrario la risposta è HTTP 401. Il valore del token non viene mai mostrato da /config.

```
Authorization: Bearer <token>
X-Api-Key: <token>
```

Usare un token lungo e casuale. Per generarne uno con PowerShell:

```
$b = New-Object byte[] 32
[System.Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($b)
[Convert]::ToBase64String($b)
```

Esempio di regola del firewall che apre la porta 3000 solo al PC del client (indirizzo di esempio):

```
New-NetFirewallRule -DisplayName "Albatros Bridge 3000" -Direction Inbound `
    -Protocol TCP -LocalPort 3000 -RemoteAddress 192.168.1.20 -Action Allow
```

Se il server è raggiungibile dalla rete (BindAddress vuoto o non locale) e ApiToken non è impostato, all'avvio viene mostrato l'avviso SECURITY come ultima riga del riepilogo.

5.4 MaxMediaMB

Limita la dimensione dei media scaricati. Il controllo avviene prima del download, sulla dimensione dichiarata da WhatsApp, e di nuovo sulla dimensione reale. Un media oltre il limite non viene scaricato: il messaggio viene salvato senza allegato e nel log viene registrato un errore. Default 100 MB; 0 = nessun limite. Un valore troppo alto aumenta l'uso di memoria e di spazio nella coda su disco, perché il media viene conservato in pending fino al salvataggio nel database.

5.5 Limiti e controlli sugli invii

I limiti riducono il rischio che il numero venga bloccato per invii troppo rapidi o troppo numerosi, ma non rendono lecito l'uso del programma per spam, pubblicità o invii massivi (vedere le avvertenze iniziali).

Coda di invio. Le richieste a /send e /send-media entrano in una coda e partono nell'ordine di arrivo, distanziate di almeno SendMinIntervalSec secondi; ogni intervallo viene allungato a caso fino al 30% per evitare un ritmo fisso. In un'ora non partono più di SendMaxPerHour messaggi. La richiesta HTTP resta in attesa del proprio turno e riceve la risposta a invio avvenuto. Se il turno arriverebbe dopo più di SendMaxWaitSec secondi, il bridge risponde subito 429 con il campo retryAfterSec e l'header Retry-After, che indicano dopo quanti secondi ritentare. SendMaxWaitSec deve restare sotto il timeout del client HTTP (in .NET il default di HttpClient è 100 secondi). Se SendMinIntervalSec e SendMaxPerHour sono entrambi 0, all'avvio viene mostrato un avviso SECURITY.

Verifica del numero. Con SendCheckNumber=Y, prima di mettere il messaggio in coda il bridge verifica che il destinatario usi WhatsApp; se non lo usa risponde 422 (code NOT_ON_WHATSAPP) e il messaggio non consuma la quota oraria.

L'esito viene ricordato per 24 ore se il numero esiste e per 10 minuti se non esiste. Gruppi (@g.us) e identificativi @lid non vengono verificati. Se la verifica non riesce per motivi tecnici (ad esempio un timeout), il messaggio viene inviato comunque e nel log compare un avviso.

Invio senza doppioni (clientMsgId). Il client può indicare nel JSON un identificativo univoco del messaggio, clientMsgId (da 1 a 100 caratteri), da riutilizzare solo quando ripete la stessa richiesta, ad esempio dopo un timeout. Se il messaggio con quel clientMsgId è già stato inviato con successo nelle ultime SendIdempotencyHours ore, il bridge non lo invia di nuovo e restituisce la stessa risposta con duplicate=true; se il primo invio è ancora in corso, ne attende l'esito. Se lo stesso clientMsgId arriva con destinatario o contenuto diversi, la risposta è 409. Gli invii non riusciti non vengono ricordati, quindi la richiesta può essere ripetuta con lo stesso clientMsgId. Gli identificativi sono tenuti in memoria e si azzerano al riavvio del programma.

/status riporta lo stato degli invii nel blocco send: messaggi inviati, invii dell'ultima ora, richieste in attesa, secondi al prossimo invio possibile, richieste rifiutate per limite o per numero non registrato, duplicati.

6. Configurazione del database

Sono supportati SQL Server, MySQL, MariaDB, SQLite via ODBC e SQLite incorporato.

```
TypeDB=InternalSqlite
#ConnStr=
# senza ConnStr:
# PathWork\Instance_<CodeInstance>\Albatros_Embedded.db
```

Con InternalSqlite non è necessario installare un driver ODBC per il database. Se il file è occupato da un altro programma, il bridge effettua alcuni ritentativi rapidi e mantiene i dati in coda.

Tabella	Contenuto
hh_Albatros_Webhooks	Testata dei messaggi/eventi e RawData integrale.
hh_Albatros_WebHooks_Media	Allegati multimediali associati alla testata.
hh_Albatros_WebHooks_Locations	Posizione (latitudine e longitudine) dei messaggi di tipo posizione, collegata alla testata.
hh_Albatros_WebHooks_Secrets	messageSecret dei messaggi. Il numero collegato e la Key non sono memorizzati in questa tabella: si ricavano con una JOIN su hh_Albatros_Webhooks tramite MsgID.
hh_Albatros_Webhooks_log	Log diagnostico.
hh_Albatros_SendQueue	Messaggi da inviare scritti da un altro programma (invio da database). Creata solo con DbSend=Y; vedere 15.

Le tabelle vengono create automaticamente se mancano. All'avvio il programma crea anche l'indice IX_hh_Albatros_Webhooks_MsgID sulla colonna MsgID della testata, se non esiste già, per velocizzare la ricerca dei messaggi originali (necessaria per gli edit). Su una tabella molto grande la creazione può richiedere alcuni minuti, durante i quali i messaggi restano in coda. Se l'utente del database non ha il permesso di creare indici, viene mostrato solo un avviso e il programma prosegue normalmente.

Ogni messaggio riceve un identificativo univoco (MsgUID) nel momento in cui entra in coda. Se un salvataggio è già stato completato ma il programma si è interrotto prima di cancellare il file di coda, al nuovo tentativo la riga viene riconosciuta e non viene inserita una seconda volta.

7. Password del database e sessione WhatsApp cifrate

La password cifrata è legata a elementi hardware del PC. Per generarla:

```
Albatros-Bridge-odbc.exe --pwd
```

La password viene richiesta due volte con input nascosto quando PowerShell è disponibile. Al termine viene prodotta una riga PwdConnStr=ENC:... da copiare nel file di configurazione. Una password cifrata su un PC non è utilizzabile su un altro PC (né dopo la sostituzione di scheda madre, CPU o disco di sistema): in questi casi va rigenerata. Lo stesso valore cifrato può essere usato per DbSendSecret: il programma lo propone anche nella riga DbSendSecret=ENC:... (vedere 15).

7.1 Cifratura della sessione WhatsApp

I file della sessione WhatsApp, nella cartella auth_info, sono sempre cifrati con una chiave legata agli elementi hardware del PC, diversa da quella usata per la password del database. La cifratura è automatica, sempre attiva e non configurabile: non richiede alcun intervento.

• Chi copia la cartella auth_info su un altro PC non può usarla per collegarsi con il numero WhatsApp dell'istanza.

• Aggiornamento da una versione precedente: al primo avvio i file esistenti in chiaro vengono cifrati automaticamente, senza una nuova scansione del QR.

• Cambio di PC o di hardware (scheda madre, CPU, disco C, oppure macchina virtuale spostata su un host con CPU diversa): la sessione non è più decifrabile e va ricollegata con un nuovo QR.

• In questo caso la cartella non viene cancellata: viene rinominata in auth_info.undecryptable_<data e ora> e viene creata una nuova sessione. Dopo aver verificato il nuovo collegamento la cartella rinominata può essere eliminata; sul telefono, in Dispositivi collegati, conviene rimuovere il dispositivo precedente.

• Il parametro EncryptAuth non è supportato: se presente nel file di configurazione viene ignorato con un avviso.

All'avvio la lettura degli elementi hardware può richiedere qualche secondo. Se la prima verifica non riesce, il programma la ripete prima di considerare la sessione non decifrabile.

La cifratura impedisce l'uso della sessione su un altro PC, ma non sostituisce la protezione del PC stesso: chi accede al PC con i permessi dell'account che esegue il bridge può utilizzare la sessione. Proteggere quindi l'accesso al server.

8. Avvio del programma

Ricerca della configurazione: --config; PHONES_BRIDGE_CONFIG; Instance_Config.ini nella cartella del programma; percorso storico; infine valori di default con avviso.

All'avvio vengono sempre mostrati banner e riepilogo della configurazione, che comprende anche BindAddress, ApiToken (solo se impostato o no, mai il valore), MaxMediaMB, i parametri degli invii (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec, SendCheckNumber, SendIdempotencyHours), l'invio da database (DbSend e, se attivo, DbSendSecret: solo se è cifrato o in chiaro, mai il valore) e la riga Session files, che conferma la cifratura della sessione. Il riepilogo termina con DebugLog, DebugLogFilter, l'indicazione per esclusione dei media accettati e, se le condizioni lo richiedono, l'avviso SECURITY.

```
   MaxMediaMB            : 100 MB
   MaxMemQueue           : 1500
   SendMinIntervalSec    : 3s between two sends (+ up to 30% random)
   SendMaxPerHour        : 60 sends per hour
   SendMaxWaitSec        : 60s maximum wait in the send queue, then HTTP 429
   SendCheckNumber       : Y (recipient verified before sending)
   SendIdempotencyHours  : 24h (repeated clientMsgId is not sent again)
   DbSend                : Y (table hh_Albatros_SendQueue, read every 5s, rows valid 60 minutes)
   DbSendSecret          : set (encrypted, bound to this PC)
   ConsoleLog            : Y (console debug enabled)
   DebugLog              : All (DEBUG, ERROR, INFO, WARNING)
   DebugLogFilter        : NO-VIDEO, NO-OTHER
   Accepted media        : AUDIO, PDF
⚠ SECURITY: the HTTP server is reachable from the network without ApiToken: ...
```

Il riepilogo indica anche la cartella della coda di avvio (Startup queue folder), descritta in 10.6.

Manuali utente: all'avvio i manuali in italiano e in inglese vengono salvati nella cartella del programma, con gli stessi nomi predefiniti di --man_IT e --man_EN, solo se non esistono già: un file esistente non viene mai sovrascritto. Per ogni file creato viene mostrata una riga informativa; se la scrittura non è possibile compare un avviso e il programma prosegue. Il salvataggio automatico si disattiva con il parametro --Skip-Manuals oppure con Skip-Manuals=Y nel file di configurazione.

Se si verifica un errore imprevisto non gestito, il programma lo scrive nel log (a video, in Connection.log e nella tabella di log) e termina con codice di uscita 1 dopo circa 2 secondi. Per un funzionamento continuo è consigliato eseguirlo come servizio Windows configurato per il riavvio automatico in caso di errore.

9. Primo collegamento a WhatsApp

• Se non esistono credenziali valide viene mostrato il QR.

• Lo stesso QR viene salvato in QrCode\latest.png ed è disponibile anche dall'endpoint /qr (con il token, se ApiToken è impostato).

• Sul telefono: Impostazioni → Dispositivi collegati → Collega un dispositivo.

• Dopo la scansione WhatsApp richiede un riavvio della connessione: il bridge si ricollega immediatamente.

• Quando la connessione è aperta, il file QR viene eliminato.

• I file della nuova sessione vengono creati già cifrati (vedere 7.1).

• Durante una riconnessione automatica il QR non viene mostrato né salvato se non è realmente necessario.

• Dopo un logout, la sessione viene rimossa e viene nuovamente richiesto il QR.

10. Funzionamento generale

Ogni messaggio/evento viene classificato, eventualmente associato a un messaggio precedente, serializzato e inserito nel flusso di salvataggio. Ogni elemento passa dalla coda persistente su disco e il file viene cancellato solo dopo il salvataggio riuscito nel database.

10.1 Retry e code

• Backoff lineare con DbRetrySec e tetto DbRetryMaxSec; ogni nuovo dato provoca comunque un tentativo immediato.

• Il primo accesso riuscito al database azzera il contatore dei tentativi.

• I file di coda vengono scritti in modo sicuro (file temporaneo e rinomina): un'interruzione improvvisa non lascia file incompleti.

• Gli errori di disponibilità del database (rete, server spento, database occupato) vengono ritentati secondo DbMaxRetries e PendingMaxSec.

• Un file di coda illeggibile o un dato rifiutato dal database (ad esempio un valore non valido) viene spostato subito in pending_db\failed, senza bloccare gli elementi successivi.

• Gli elementi che superano i limiti vengono spostati in pending_db\failed; la causa è riportata a video e nel log.

• Dopo il salvataggio riuscito di un messaggio può essere avviato ExeName (vedere più avanti in questo paragrafo).

Per ritentare un elemento finito in failed, dopo averne corretto la causa, spostare il file da pending_db\failed a pending_db e riavviare il bridge: i file presenti in pending_db vengono caricati all'avvio.

Eseguibile post-inserimento. Se ExeName è impostato, dopo ogni messaggio salvato nel database (non per i messageSecret) il bridge avvia PathBin\ExeName senza attendere che termini, passando tre parametri separati:

```
--rec=<ID>          ID della riga in hh_Albatros_Webhooks
--istance=<istanza> codice istanza (CodeInstance) che ha registrato la riga
--Cell=<numero>     numero collegato all'istanza (PhoneConnected), solo cifre
```

Esempio di riga di comando ricevuta dal programma avviato:

```
GestioneMessaggi.exe --rec=45 --istance=Magazzino --Cell=393479528620
```

I valori sono quelli registrati nella riga (colonne ID, Instance e PhoneConnected). Un codice istanza che contiene spazi arriva comunque come unico parametro, tra virgolette. Se l'eseguibile non esiste viene mostrato un avviso una sola volta e i salvataggi proseguono normalmente; il riepilogo all'avvio mostra, accanto a Exe, il formato dei parametri. Nelle versioni precedenti l'unico parametro era l'ID del record.

10.2 Messaggi modificati

Con Save_Secrets=S il messageSecret viene catturato prima dei filtri di salvataggio della testata, tranne che per i messaggi esclusi dalla configurazione (Save_Sent=N, Save_StatusBroadcast=N, Save_protocolMessage=N); le modifiche e le cancellazioni dei messaggi esclusi da Save_Sent e Save_StatusBroadcast non vengono salvate. La logica corrente gestisce anche gli edit cifrati LID e, quando necessario, recupera il messaggio originale cercandolo nel database (ignorando le righe di aggiornamento stato, che hanno lo stesso MsgID) oppure, se non ancora salvato, nella coda su disco.

10.3 RelatedMsgID

Evento	Origine
Risposta / citazione	contextInfo.stanzaId
Reazione	reactionMessage.key.id
Protocollo	protocolMessage.key.id
Modifica	targetMessageKey.id

10.4 Media e posizioni

• Il download di un media viene tentato fino a 3 volte, con brevi attese tra un tentativo e l'altro.

• Se il download non riesce o il media supera MaxMediaMB, il messaggio viene salvato comunque, senza allegato (MediaCount=0); nel log viene registrato un errore e l'elemento restituito da /messages contiene il campo mediaError con la causa.

• Una posizione con coordinate non valide non fa perdere il messaggio: viene salvato senza la riga in hh_Albatros_WebHooks_Locations e con il campo locationError; le coordinate originali restano in RawData.

• Contatti e sondaggi vengono registrati con tipo contact (testo = nome visualizzato) e poll (testo = domanda); il contenuto completo resta in RawData.

• In caso di altri errori durante l'elaborazione, il messaggio non viene considerato "già visto": una nuova consegna da parte di WhatsApp viene quindi elaborata normalmente.

10.5 Riconnessione automatica

Causa della chiusura	Comportamento
Rete assente o connessione persa	Nuovo tentativo con attesa crescente: circa 1, 2, 4, 8 ... secondi, fino a un massimo di 60 secondi. Il contatore si azzera quando la connessione si riapre.
Riavvio richiesto da WhatsApp (dopo la scansione del QR)	Riconnessione immediata.
Sessione sostituita	Un'altra istanza sta usando le stesse credenziali: viene mostrato un errore e il tentativo successivo avviene dopo 60 secondi.
Logout dal telefono	Le credenziali vengono eliminate e viene richiesto un nuovo QR.

Durante l'attesa /status riporta connected=false, il numero del tentativo (reconnectAttempt) e i secondi mancanti (reconnectInSec). Le richieste di invio ricevute in questa fase rispondono HTTP 503.

10.6 Messaggi ricevuti prima della disponibilità del numero collegato

All'avvio, se ci sono messaggi in attesa sul server, WhatsApp può consegnarli prima che il numero collegato sia disponibile. Per non registrarli con il campo PhoneConnected vuoto, il bridge li mette nella coda di avvio e li registra appena il numero è noto.

• Ogni evento ricevuto in questa fase (messaggio, modifica, cancellazione, aggiornamento di stato) viene scritto in un file nella cartella early_queue dell'istanza, in modo protetto da interruzioni improvvise. Se il programma viene chiuso, all'avvio successivo i file presenti vengono ricaricati e restano in attesa del numero collegato.

• Appena il numero è disponibile gli eventi vengono registrati uno alla volta, dal più vecchio al più nuovo, secondo la data/ora interna di WhatsApp. Nello stesso secondo il messaggio precede la sua modifica e la modifica precede la cancellazione; in ogni caso modifiche, cancellazioni, reazioni e aggiornamenti di stato vengono registrati dopo il messaggio a cui si riferiscono, così il messageSecret del messaggio originale è già disponibile quando si registra la modifica.

• Anche gli eventi che arrivano mentre la coda viene svuotata vengono accodati: nessun evento può essere registrato prima di uno più vecchio.

• Il file di un evento viene cancellato solo dopo la registrazione. Se il programma si interrompe dopo la registrazione ma prima della cancellazione, al riavvio l'evento viene elaborato di nuovo senza creare righe doppie.

• Un file illeggibile o un evento che non è stato possibile elaborare viene spostato in early_queue\failed; la causa è riportata a video e nel log.

• Se l'attesa supera 30 secondi viene mostrato un avviso con il numero di eventi in coda, ripetuto ogni 60 secondi. /status riporta il numero di eventi in coda nel campo startupQueue.

• Per le righe registrate dalla coda, DateReceived corrisponde al momento della registrazione; la data/ora di WhatsApp (DateWA) resta quella originale.

• Se un evento resta a lungo nella coda di avvio, il media potrebbe non essere più scaricabile perché i collegamenti di WhatsApp scadono: in questo caso il messaggio viene salvato senza allegato, come descritto in 10.4.

11. Interfaccia HTTP (endpoint)

Indirizzo base: http://<indirizzo>:<Port> — default porta 3000. Se ApiToken è impostato, ogni richiesta deve contenere il token (vedere 5.3). I dati inviati con POST sono in formato JSON (Content-Type: application/json).

Metodo	Endpoint	Descrizione
GET	`/status`	Stato connessione, istanza, DB, coda, coda di avvio (startupQueue), riconnessione, invii (send), invio da database (sendQueue) e statistiche.
GET	`/qr`	QR corrente; 404 se non disponibile.
GET	`/messages`	Restituisce e svuota la coda messaggi.
GET	`/messages?peek=1`	Restituisce la coda messaggi senza svuotarla; ogni elemento ha il numero seq.
POST	`/messages/ack`	Conferma i messaggi elaborati: {"upToSeq": N} rimuove quelli con seq fino a N.
GET	`/message-status`	Restituisce e svuota gli aggiornamenti degli stati.
GET	`/message-status?peek=1`	Come sopra, senza svuotare (con seq).
POST	`/message-status/ack`	Conferma gli aggiornamenti di stato elaborati ({"upToSeq": N}).
GET	`/debug-log`	Stato del log diagnostico.
GET	`/db-queue`	Coda, retry e parametri.
POST	`/db-retry`	Forza un tentativo immediato.
GET	`/config`	Configurazione attiva senza password né token.
GET	`/message-secret/:msgId`	MessageSecret del messaggio.
POST	`/send`	Invia testo (limiti di invio, verifica del numero e clientMsgId facoltativo: vedere 5.5).
POST	`/send-media`	Invia media tramite base64 (stessi controlli di /send).
POST	`/send-queue/poll`	Anticipa la lettura della tabella degli invii (vedere 15). Risposta 202; 409 DB_SEND_DISABLED se l'invio da database non è attivo.

11.1 Lettura dei messaggi senza perdite (peek / ack)

La lettura classica di /messages svuota la coda nel momento in cui risponde: se il client si interrompe prima di aver elaborato la risposta, quei messaggi non vengono più restituiti (restano comunque salvati nel database). La lettura a due fasi evita il problema:

• GET /messages?peek=1: restituisce i messaggi in coda, ciascuno con un numero progressivo seq, senza rimuoverli.

• Il client elabora i messaggi.

• POST /messages/ack con {"upToSeq": <seq dell'ultimo messaggio elaborato>}: rimuove dalla coda i messaggi confermati. Se il client cade prima della conferma, al giro successivo li ritrova.

```
GET  /messages?peek=1
{ "key": "Albatros", "messages": [ { "id": "3EB0...", "type": "text", "text": "ciao", ..., "seq": 41 },
                                   { "id": "3EB0...", "type": "image", ..., "seq": 42 } ] }

POST /messages/ack        { "upToSeq": 42 }
{ "success": true, "removed": 2, "remaining": 0 }
```

La coda resta comunque limitata a MaxMemQueue elementi: se il client non conferma per molto tempo, gli elementi più vecchi vengono scartati. Lo stesso schema vale per /message-status e /message-status/ack. Non usare la lettura classica e la lettura a due fasi contemporaneamente da client diversi.

11.2 Invio di messaggi

Il destinatario (number) può essere un numero con prefisso internazionale, anche con +, spazi, trattini o parentesi, oppure un identificativo WhatsApp completo (@s.whatsapp.net, gruppo @g.us, @lid).

```
POST /send
{ "number": "+39 340 123 4567", "message": "Testo del messaggio",
  "clientMsgId": "ORD-2026-000123" }

POST /send-media
{ "number": "393401234567", "mediaType": "document", "fileName": "fattura.pdf",
  "caption": "Fattura di ottobre", "base64Data": "JVBERi0xLjcK..." }

Risposta: { "success": true, "id": "<id del messaggio inviato>" }
Con clientMsgId: { "success": true, "id": "<id>", "clientMsgId": "ORD-2026-000123",
  "duplicate": false }
Errore: { "success": false, "code": "SEND_HOURLY_LIMIT",
  "error": "Hourly send limit reached", "retryAfterSec": 1250 }
```

mediaType	Invio
image	Immagine, con eventuale caption.
audio	Audio (inviato come audio/mpeg).
qualsiasi altro valore	Documento: fileName (default file.pdf) ed eventuale caption.

Il campo clientMsgId è facoltativo (vedere 5.5). Le risposte di errore di /send e /send-media contengono, oltre a error, il campo code, che il client può usare al posto del testo:

code (HTTP)	Significato
INVALID_NUMBER (400)	number mancante o non valido.
INVALID_MESSAGE (400)	message mancante o vuoto (/send).
INVALID_MEDIA (400)	base64Data mancante o vuoto (/send-media).
INVALID_CLIENT_MSG_ID (400)	clientMsgId non valido (non testo, vuoto o oltre 100 caratteri).
CLIENT_MSG_ID_CONFLICT (409)	clientMsgId già usato per un messaggio diverso.
NOT_ON_WHATSAPP (422)	Il numero non usa WhatsApp (SendCheckNumber=Y). Non ritentare.
SEND_HOURLY_LIMIT (429)	Raggiunto SendMaxPerHour: ritentare dopo retryAfterSec secondi.
SEND_QUEUE_BUSY (429)	Attesa in coda oltre SendMaxWaitSec: ritentare dopo retryAfterSec secondi.
SEND_ERROR (500)	Errore restituito da WhatsApp durante l'invio.

code (HTTP)	Significato
NOT_CONNECTED (503)	WhatsApp non connesso.

11.3 Codici di risposta

Codice	Significato
200	Richiesta eseguita.
400	Dati mancanti o non validi (ad esempio number non valido, message vuoto, base64Data assente, upToSeq non numerico, JSON malformato).
401	Token mancante o errato (solo con ApiToken impostato).
404	Risorsa non disponibile (ad esempio QR non presente).
409	clientMsgId già usato per un messaggio diverso (/send, /send-media).
413	Corpo della richiesta troppo grande (oltre 50 MB).
422	Il numero destinatario non usa WhatsApp (SendCheckNumber=Y).
429	Limite di invio raggiunto (SendMaxPerHour) oppure attesa in coda oltre SendMaxWaitSec: ritentare dopo retryAfterSec secondi (header Retry-After).
500	Errore interno o errore restituito da WhatsApp durante l'invio.
503	WhatsApp non connesso oppure database non disponibile (/message-secret).

11.4 Esempio di client C#.NET

```
using System;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;

var http = new HttpClient { BaseAddress = new Uri("http://127.0.0.1:3000/") };
http.DefaultRequestHeaders.Authorization =
    new AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale");

// Lettura senza svuotare la coda
string json = await http.GetStringAsync("messages?peek=1");
// ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

// Conferma dei messaggi elaborati
var ack = new StringContent("{\"upToSeq\":" + ultimoSeq + "}", Encoding.UTF8, "application/json");
HttpResponseMessage r = await http.PostAsync("messages/ack", ack);
```

11.5 Esempio di client VB.NET

```
Imports System.Net.Http
Imports System.Net.Http.Headers
Imports System.Text

Dim http As New HttpClient()
http.BaseAddress = New Uri("http://127.0.0.1:3000/")
http.DefaultRequestHeaders.Authorization =
    New AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale")

' Lettura senza svuotare la coda
Dim json As String = Await http.GetStringAsync("messages?peek=1")
' ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

' Conferma dei messaggi elaborati
Dim ack As New StringContent("{""upToSeq"":" & ultimoSeq & "}", Encoding.UTF8, "application/json")
Dim r As HttpResponseMessage = Await http.PostAsync("messages/ack", ack)

' Invio con clientMsgId: nuovo per ogni messaggio, riutilizzato solo nei ritentativi
Dim idMsg As String = Guid.NewGuid().ToString()
Dim body As String = "{""number"":""393401234567"",""message"":""Testo"",""clientMsgId"":""" & idMsg & """}"
Dim s As HttpResponseMessage = Await http.PostAsync("send", New StringContent(body, Encoding.UTF8,
"application/json"))
Select Case CInt(s.StatusCode)
    Case 429 ' limite raggiunto: attendere e ritentare con lo stesso idMsg
```

```
        Dim attesa As TimeSpan = If(s.Headers.RetryAfter?.Delta, TimeSpan.FromSeconds(60))
    Case 422 ' numero non registrato su WhatsApp: non ritentare
End Select
```

12. Log delle connessioni e log a video

Connection.log si trova in PathWork\Instances_Log\<numero connesso>\Connection.log. Contiene il log di connessione e una copia dell'output testuale.

Contenuto	Visibilità
Banner	Sempre
Riepilogo di avvio	Sempre
Avviso SECURITY	Sempre, quando il server è raggiungibile dalla rete senza ApiToken
QR	Sempre quando realmente richiesto
Warning / Error	Sempre
Errori non gestiti	Sempre, prima della chiusura del programma
INFO / DEBUG	Con ConsoleLog=Y
Normali log	Data/ora compatta all'inizio

Durante una perdita prolungata di Internet, le righe identiche consecutive "Connection closed. Reconnecting: true" vengono registrate una sola volta, sia a video sia nel log.

13. Diagnosi e risoluzione problemi

Sintomo	Azione
File di configurazione non trovato	Usare --create_config= oppure --config=.
Connessione DB fallita	Verificare TypeDB, ConnStr e PwdConnStr; i dati restano in coda.
Password ENC non decifrabile	Rigenerare con --pwd sul PC che esegue il bridge.
SQLite occupato	Attendere il retry automatico; verificare quale programma tiene aperto il file.
Primo avvio lento con un database grande	Creazione dell'indice su MsgID in corso: attendere; i messaggi restano in coda.
Il client non raggiunge il bridge	Se il client è su un altro PC: BindAddress vuoto o 0.0.0.0, regola del firewall per la porta, ApiToken. Se è sullo stesso PC: usare http://127.0.0.1:<Port>.
Risposta HTTP 401	Token mancante o diverso da ApiToken: verificare l'header Authorization: Bearer oppure X-Api-Key.
Risposta HTTP 400 su /send	Controllare number (numero con prefisso o JID) e message (non vuoto).
Risposta HTTP 503 su /send	WhatsApp non connesso: controllare /status (connected, reconnectInSec).
Risposta HTTP 429 su /send	Limite di invio raggiunto: attendere retryAfterSec secondi prima di ritentare. Se succede spesso, ridurre il ritmo degli invii o adeguare SendMinIntervalSec, SendMaxPerHour e SendMaxWaitSec (vedere 5.5).
Risposta HTTP 422 su /send	Il numero non usa WhatsApp: controllare il numero e il prefisso internazionale. La verifica si disattiva con SendCheckNumber=N.
Risposta HTTP 409 su /send	Lo stesso clientMsgId è stato usato per un messaggio diverso: il client deve generare un clientMsgId nuovo per ogni messaggio e riutilizzarlo solo nei ritentativi.
Avviso SECURITY all'avvio	Impostare ApiToken oppure BindAddress=127.0.0.1.
Messaggio salvato senza media	Vedere il log ERROR (download non riuscito o MaxMediaMB superato) e il campo mediaError.
File in pending_db\failed	Leggere la causa nel log; dopo la correzione spostare il file in pending_db e riavviare.

Sintomo	Azione
Avviso "Connected number still not available"	Il numero collegato non è ancora disponibile: i messaggi ricevuti restano nella coda di avvio e vengono registrati appena il numero è noto (vedere 10.6). Se l'avviso persiste, controllare la connessione con /status.
File in early_queue\failed	Leggere la causa nel log; dopo la correzione spostare il file in early_queue e riavviare.
Manuali non desiderati nella cartella del programma	Avviare con --Skip-Manuals oppure impostare Skip-Manuals=Y; i file già presenti possono essere eliminati.
Errore "session replaced"	Un'altra istanza usa la stessa cartella auth_info: lasciarne attiva una sola.
Nuovo QR richiesto dopo un cambio di PC o di hardware	Comportamento previsto: la sessione è legata al PC (vedere 7.1). Scansionare il nuovo QR; dopo la verifica eliminare la cartella auth_info.undecryptable_<data> e rimuovere il vecchio dispositivo da Dispositivi collegati sul telefono.
Cartella auth_info copiata da un altro PC non funziona	Comportamento previsto: i file cifrati sono utilizzabili solo sul PC che li ha creati. Collegare l'istanza con un nuovo QR.
Avviso "EncryptAuth is ignored" all'avvio	La sessione è sempre cifrata: togliere la riga EncryptAuth dal file di configurazione.
Errore all'avvio su DbSendSecret	Il segreto manca, è più corto di 32 caratteri o non è decifrabile su questo PC: l'invio da database resta disattivato. Impostare DbSendSecret oppure rigenerarlo con --pwd sul PC del bridge.
Righe della tabella degli invii in stato K	Il programma che scrive usa un segreto diverso da DbSendSecret, oppure i dati sono stati modificati dopo la firma. Con MySQL verificare la codifica utf8mb4.
Righe della tabella degli invii che restano in stato N	Verificare DbSend=Y, il collegamento (connected in /status), che PhoneSender delle righe sia il numero collegato e i limiti di invio.
Righe della tabella degli invii in stato X	Righe scritte troppo prima della lettura oppure orologi dei PC non allineati: aumentare DbSendMaxAgeMin (massimo 1440) o sincronizzare gli orologi.
Righe della tabella degli invii in stato U	Il bridge si è fermato due volte durante l'invio della stessa riga: verificare in hh_Albatros_Webhooks se il messaggio è partito e, se necessario, reinserirlo con un nuovo ClientRef.
Il programma si chiude con un errore	Leggere Connection.log; eseguire il bridge come servizio con riavvio automatico.
Console senza INFO/DEBUG	Impostare ConsoleLog=Y.
QR non appare	Verificare se la sessione è ancora valida.
Edit non decifrato	Verificare Save_Secrets=S e messageSecret.
Log tabella assente	Impostare DebugLog=All o un elenco di tipi.
Media indesiderati in DebugData	Configurare DebugLogFilter.
Numero WhatsApp bloccato (ban)	Il blocco è deciso e applicato da Meta, di norma per violazione delle regole di WhatsApp (spam, pubblicità, invii massivi): va gestito direttamente con WhatsApp dall'app sul telefono. Vedere le avvertenze iniziali.
Il programma smette di collegarsi dopo un aggiornamento di WhatsApp	Possibile modifica del protocollo da parte di WhatsApp (vedere le avvertenze iniziali): verificare la disponibilità di una versione aggiornata del programma.
Password ENC o sessione non decifrabili senza cambi di hardware	Verificare che PowerShell sia disponibile per l'account del programma e che il servizio WMI sia attivo.

14. Buone pratiche

• Usare InternalSqlite per installazioni di test quando non serve un DB esterno o si opera in un ambiente protetto.

• In produzione, con un database esterno, preferire PwdConnStr=ENC:... (con InternalSqlite la password non è necessaria).

• Impostare sempre ApiToken quando il bridge è raggiungibile da altri PC; usare BindAddress=127.0.0.1 quando il client è sullo stesso PC.

• Nei client, preferire la lettura a due fasi (?peek=1 e /ack) alla lettura classica.

• Mantenere Save_Secrets=S se servono gli edit.

• Monitorare /status e /db-queue e controllare periodicamente pending_db\failed ed early_queue\failed.

• Eseguire il bridge come servizio Windows con riavvio automatico in caso di errore.

• Non usare il programma per spam, pubblicità commerciale o invii massivi: sono vietati dalle regole di WhatsApp e possono causare il blocco del numero.

• Mantenere attivi i limiti di invio (SendMinIntervalSec, SendMaxPerHour) e inviare un clientMsgId univoco per ogni messaggio, riutilizzandolo solo quando il client ripete la stessa richiesta.

• Per l'invio da database usare DbSendSecret cifrato (ENC:...), non scrivere mai il segreto nel database e cancellare periodicamente le righe rimaste negli stati E, K, X e U.

• La cartella auth_info è cifrata e legata al PC: copiarla su un altro PC non serve (occorre comunque un nuovo QR). Limitarne comunque l'accesso agli account che eseguono il bridge; il ripristino da backup funziona solo sullo stesso PC.

• Usare CodeInstance distinti per più numeri WhatsApp; non usare la stessa auth_info su due istanze.

• Attivare DebugLog e ConsoleLog durante la diagnosi.

• Impostare DebugLogFilter per limitare i media diagnostici e MaxMediaMB in base alla memoria disponibile.

• Adeguare PendingMaxSec e DbMaxRetries al contesto operativo.

15. Invio da database (tabella degli invii)

Oltre agli endpoint HTTP, il bridge può inviare i messaggi che un altro programma scrive nella tabella hh_Albatros_SendQueue. Con DbSend=Y il bridge legge la tabella ogni DbSendPollSec secondi e invia le righe del solo numero collegato, come se fossero arrivate da /send o /send-media: stessa coda, stessi limiti (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec) e stessa verifica del numero (SendCheckNumber).

Ogni riga è firmata con un segreto noto solo al bridge e al programma che scrive: chi accede al database senza il segreto non può creare righe valide né modificarne il contenuto.

15.1 Attivazione

• DbSend=Y attiva la funzione. La tabella viene creata automaticamente all'avvio; un eventuale errore di creazione viene segnalato ma non blocca il salvataggio dei messaggi.

• DbSendSecret è il segreto della firma (minimo 32 caratteri), in chiaro oppure ENC:... generato con --pwd. Senza un segreto valido la funzione resta disattivata e l'avvio lo segnala con un errore. Lo stesso segreto deve essere noto al programma che scrive le righe e non va mai scritto nel database.

• DbSendPollSec (da 1 a 300, default 5): secondi tra due letture. La tabella viene letta solo quando il bridge è collegato e il numero collegato è noto.

• DbSendMaxAgeMin (default 60): minuti di validità di una riga; valori sotto 30 diventano 30, oltre 1440 diventano 1440.

• POST /send-queue/poll anticipa la lettura (risposta 202). Più richieste ravvicinate producono una sola lettura.

```
DbSend=Y
DbSendSecret=ENC:...
```

```
DbSendPollSec=5
DbSendMaxAgeMin=60
```

15.2 Struttura della tabella

Le colonne indicate come firmate entrano nella firma (vedere 15.4). Chi scrive inserisce la riga con SendStatus = N; le colonne del bridge vengono gestite dal bridge.

Colonna	Scritta da	Descrizione
ID	database	Identificativo progressivo; le righe vengono lavorate in ordine di ID.
DateInsert	database	Data di inserimento (solo informativa).
PhoneSender	chi scrive	Firmata. Numero che deve inviare, solo cifre: il bridge legge solo le righe del proprio numero.
ClientRef	chi scrive	Firmata. Riferimento univoco per PhoneSender (indice univoco), massimo 100 caratteri.
Recipient	chi scrive	Firmata. Destinatario: numero con prefisso internazionale oppure JID (gruppi @g.us).
MsgKind	chi scrive	Firmata. Tipo di messaggio (vedere 15.3).
Title	chi scrive	Firmata. Nome del luogo, nome del contatto oppure domanda del sondaggio.
MsgText	chi scrive	Firmata. Testo, didascalia, indirizzo, vCard oppure opzioni del sondaggio.
FileName	chi scrive	Firmata. Nome del documento.
MimeType	chi scrive	Firmata. Tipo MIME del media; facoltativo.
Latitude, Longitude	chi scrive	Firmate. Coordinate come testo con il punto decimale.
MediaData	chi scrive	Firmata tramite il suo SHA-256. File allegato in binario (non base64).
CreatedAtUtc	chi scrive	Firmata. Data di creazione in millisecondi UTC dal 01/01/1970.
SignVer	chi scrive	Versione del formato della firma: 1.
Signature	chi scrive	Firma HMAC-SHA256 in esadecimale (64 caratteri).
SendStatus	chi scrive / bridge	Stato della riga (vedere 15.5); chi scrive inserisce N.
ClaimToken, ClaimedBy, ClaimedAtUtc	bridge	Presa in carico della riga.
Attempts	bridge	Tentativi aggiuntivi già usati (massimo 1).
SentMsgID	bridge	Identificativo del messaggio, scelto prima dell'invio.
ErrorCode, ErrorText, ProcessedAtUtc	bridge	Esito di una riga non inviata.

15.3 Tipi di messaggio (MsgKind)

MsgKind	Colonne usate	Note
text	MsgText	Messaggio di solo testo.
image	MediaData; MsgText (didascalia), MimeType	Immagine.
video	MediaData; MsgText (didascalia), MimeType	MIME predefinito video/mp4.
gif	MediaData (mp4); MsgText (didascalia)	Video riprodotto come GIF animata.
audio	MediaData; MimeType	MIME predefinito audio/mpeg.
voice	MediaData; MimeType	Nota vocale; MIME predefinito audio/ogg; codecs=opus.
document	MediaData, FileName; MsgText (didascalia), MimeType	MIME ricavato dall'estensione se MimeType è vuoto.
sticker	MediaData (webp)	Adesivo.
location	Latitude, Longitude; Title (nome), MsgText (indirizzo)	Latitudine da -90 a 90, longitudine da -180 a 180.
contact	MsgText (una o più vCard); Title (nome visualizzato)	Ogni vCard da BEGIN:VCARD a END:VCARD.
poll	Title (domanda), MsgText (opzioni, una per riga)	Da 2 a 12 opzioni; una sola scelta.
pollmulti	Title (domanda), MsgText (opzioni, una per riga)	Come poll, con scelte multiple.

Il media non può superare MaxMediaMB. Reazioni, modifiche, cancellazioni e inoltri non sono gestiti, perché richiedono un messaggio già esistente.

15.4 Firma

Signature è l'HMAC-SHA256, con chiave il segreto DbSendSecret in UTF-8, di questo testo UTF-8:

```
ALB-SEND-1
<lunghezza>:<PhoneSender>
<lunghezza>:<ClientRef>
<lunghezza>:<Recipient>
<lunghezza>:<MsgKind>
<lunghezza>:<Title>
<lunghezza>:<MsgText>
<lunghezza>:<FileName>
<lunghezza>:<MimeType>
<lunghezza>:<Latitude>
<lunghezza>:<Longitude>
<lunghezza>:<MediaSha256>
<lunghezza>:<CreatedAtUtc>
```

• Ogni riga termina con il solo carattere LF (a capo Unix); <lunghezza> è il numero di byte UTF-8 del valore.

• Un valore NULL vale come stringa vuota. MediaSha256 è lo SHA-256 esadecimale minuscolo di MediaData, vuoto se MediaData è NULL o vuoto. CreatedAtUtc è scritto in cifre.

• Signature è in esadecimale (maiuscolo o minuscolo). Il confronto avviene a tempo costante.

• I valori firmati devono essere identici a quelli scritti nella tabella: con MySQL la connessione e la tabella devono usare utf8mb4, altrimenti emoji e caratteri speciali cambiano e la riga risulta con firma non valida.

15.5 Elaborazione ed esiti

Per ogni riga il bridge la prende in carico in modo atomico (ClaimToken), senza blocchi sulla tabella, così due istanze collegate allo stesso numero non possono inviarla due volte. Poi controlla numero, firma e validità nel tempo, invia e registra l'esito. La riga viene presa in carico solo se la coda di invio può farla partire entro SendMaxWaitSec.

SendStatus	Significato
N	Da inviare (scritta da chi accoda il messaggio).
C	In lavorazione.
(riga cancellata)	Inviata. Il messaggio resta in hh_Albatros_Webhooks con Flag_Base = N_TAB; il media viene salvato senza essere riscaricato.
E	Errore definitivo: ErrorCode ed ErrorText indicano il motivo.
K	Firma non valida: la riga non viene inviata e il log riporta un errore SECURITY.
X	Scaduta (più vecchia di DbSendMaxAgeMin).
U	Esito incerto: la riga non viene più inviata.

• Bridge non connesso o coda di invio piena: la riga torna in stato N senza consumare tentativi.

• Errore durante l'invio: un solo tentativo aggiuntivo, con lo stesso identificativo del messaggio; poi stato E.

• Presa in carico senza esito (ad esempio arresto del bridge): dopo 15 minuti più SendMaxWaitSec, se il messaggio risulta già in hh_Albatros_Webhooks (richiede Save_Sent=S) la riga viene cancellata; altrimenti viene fatto un solo tentativo aggiuntivo; la seconda volta la riga passa allo stato U.

• Se dopo l'invio il database non è raggiungibile, la cancellazione della riga viene ripetuta alla lettura successiva: la riga non viene reinviata.

• Le righe negli stati E, K, X e U restano nella tabella: vanno lette e cancellate dal programma che le ha scritte.

ErrorCode	Stato	Significato
INVALID_SIGNATURE	K	Firma non corrispondente: segreto diverso o dati modificati dopo la firma.
UNSUPPORTED_SIGN_VER	K	Versione della firma non gestita.
EXPIRED	X	Riga più vecchia di DbSendMaxAgeMin minuti.
CREATED_IN_FUTURE	E	CreatedAtUtc oltre 5 minuti nel futuro (orologi non allineati).
INVALID_CREATED_AT	E	CreatedAtUtc mancante o non valido.
INVALID_RECIPIENT	E	Destinatario mancante o non valido.
INVALID_KIND	E	MsgKind non gestito.
INVALID_MESSAGE	E	Testo mancante per il tipo text.
MISSING_MEDIA	E	MediaData mancante per un tipo che lo richiede.
MISSING_FILE_NAME	E	FileName mancante per il tipo document.
MEDIA_TOO_LARGE	E	MediaData oltre MaxMediaMB.
MEDIA_READ_ERROR	E	MediaData non leggibile.
INVALID_LOCATION	E	Coordinate mancanti o fuori intervallo.
INVALID_CONTACT	E	Nessuna vCard in MsgText.
INVALID_POLL	E	Domanda mancante oppure opzioni fuori da 2-12.
NOT_ON_WHATSAPP	E	Il destinatario non usa WhatsApp.
SEND_ERROR	E	Invio non riuscito anche al tentativo aggiuntivo.
CLAIM_TIMEOUT	N	Presa in carico senza esito: rimessa in attesa per il tentativo aggiuntivo.
UNCERTAIN	U	Seconda presa in carico senza esito: la riga non viene reinviata.

15.6 Controllo

/status riporta nel blocco sendQueue: attivazione, tabella, intervallo di lettura, validità delle righe, lettura in corso, ultimo errore e i contatori delle righe inviate, con firma non valida, scadute, in errore, con esito incerto, ritentate e rimesse in attesa.

/config riporta dbSend, dbSendPollSec, dbSendMaxAgeMin e se il segreto è impostato (dbSendSecretSet) e cifrato (dbSendSecretEncrypted), mai il suo valore. Il riepilogo all'avvio mostra le righe DbSend e DbSendSecret (vedere 8).

15.7 Programma Albatros SendQueue

Albatros SendQueue accoda messaggi nella tabella degli invii. È composto da un programma con interfaccia grafica (Albatros_SendQueue.exe) e dalla libreria AlbatrosSendQueueLib.dll, che contiene tutta la logica: lettura di Instance_Config.ini, firma, scrittura nella tabella, stato della riga e richieste HTTP al bridge.

La libreria può essere usata anche da altri programmi .NET (per esempio un gestionale): i parametri comuni (database, DbSendSecret, numero mittente, indirizzo del bridge) si impostano una sola volta, leggendoli da Instance_Config.ini oppure passandoli direttamente senza file di configurazione, e per ogni tipo di messaggio esiste una funzione di invio dedicata (SendText, SendImage, SendDocument, SendLocation, SendPoll...).

Come numero mittente si usa il parametro PhoneSender, che il bridge ignora. Programma e libreria hanno un proprio manuale utente.

Appendice A. Esempi di configurazione

A.1 Test sullo stesso PC con SQLite incorporato

```
Key=TEST
TypeDB=InternalSqlite
PathWork=c:\Albatros\Phone\
CodeInstance=Test
Port=3000
BindAddress=127.0.0.1
DebugLog=All
ConsoleLog=Y
```

A.2 Produzione con SQL Server e client su un altro PC

```
Key=Albatros
TypeDB=SqlServer
ConnStr=Driver={ODBC Driver 17 for SQL Server};Server=SRVSQL;Database=AlbatrosBridge;UID=albatros_bridge;
PwdConnStr=ENC:....
PathWork=c:\Albatros\Phone\
CodeInstance=393401234567
Port=3000
; BindAddress non indicato: ascolto su tutte le interfacce di rete
ApiToken=IlMioTokenLungoECasuale
MaxMediaMB=100
DebugLog=Error, Warning
DebugLogMaxRecords=10000
PendingMaxSec=172800
```

I valori di Server, Database, UID e del token sono solo esempi.

Nella connection string non va indicata la password, che si inserisce in PwdConnStr (preferibilmente cifrata con --pwd).

▣ Manual de utilizare – Albatros Bridge

Versiunea 1.4.129.6121 – Build 4 octombrie 2026

⚠ Avertismente importante

Avertisment	Detalii
Nicio asociere cu Meta	Autorul nu este în niciun fel asociat, afiliat, autorizat, sponsorizat sau aprobat de Meta Platforms, Inc., de WhatsApp LLC sau de alte societăți din grupul Meta. WhatsApp este o marcă a titularilor respectivi și este menționată doar pentru a descrie funcția programului.
Procedură neoficială	Conectarea la WhatsApp se face printr-o procedură neoficială (dispozitiv conectat, ca WhatsApp Web, prin biblioteci open source) și nu prin API-urile oficiale WhatsApp Business. Meta sau una dintre societățile care gestionează produsul WhatsApp poate modifica oricând și fără preaviz protocoalele, schemele de conectare, criptarea și regulile de utilizare: prin urmare, programul poate înceta să funcționeze, total sau parțial, până când este adaptat.
Situație comună tuturor produselor neoficiale	În lume există foarte multe societăți care realizează software sau oferă servicii WhatsApp neoficiale. Problema descrisă la punctul precedent este identică pentru toate: niciuna dintre ele nu poate garanta că modificările decise de Meta nu vor întrerupe funcționarea și nici nu le poate prevedea momentul.
Riscuri pentru număr	Utilizarea clienților neoficiali poate să nu fie conformă cu condițiile de utilizare WhatsApp și poate duce la limitări sau la blocarea numărului conectat. Evaluarea și riscul aferent rămân în sarcina celui care folosește programul.
Utilizare interzisă: spam, publicitate, trimiteri în masă	Utilizarea programului pentru trimiterea de spam, publicitate comercială sau mesaje în masă este interzisă în mod expres de regulile WhatsApp. O utilizare necorespunzătoare a programului poate duce, așadar, la blocarea (ban) numărului WhatsApp conectat. În acest caz, blocarea nu este în niciun fel imputabilă autorului: este aplicată direct de Meta din cauza utilizării necorespunzătoare.
Utilizare „ca atare”	Programul este furnizat „ca atare” („as is”), fără garanții de niciun fel, exprese sau implicite, inclusiv cele de funcționare, adecvare la un anumit scop, continuitate a serviciului și absență a erorilor.
Excluderea răspunderii	În limitele maxime permise de legea aplicabilă, autorului nu i se poate imputa nicio răspundere pentru daune directe sau indirecte (pierderea de date sau mesaje, beneficii nerealizate, întreruperea activității, blocarea numărului sau altele) care decurg din utilizarea sau din imposibilitatea de a utiliza programul, indiferent de modul în care s-a obținut accesul la acesta: contra cost, cadou, ca software gratuit sau în orice alt mod.
Date cu caracter personal	Mesajele salvate conțin date cu caracter personal. Cel care instalează și folosește programul răspunde de prelucrarea acestora, de securitatea bazei de date și de respectarea normelor privind protecția datelor (GDPR).

```
**********************************
**        Albatros Bridge         **
** rel. 1.4.129.6121 - 04/10/2026 **
**    © Raffaele Romano 2026      **
**********************************
```

Cuprins

1. Ce este programul

2. Cerințe

3. Parametri din linia de comandă

4. Fișiere și foldere principale

5. Fișierul de configurare Instance_Config.ini

6. Configurarea bazei de date

7. Parola bazei de date și sesiunea WhatsApp criptate

8. Pornirea programului

9. Prima conectare la WhatsApp

10. Funcționare generală

11. Interfața HTTP (endpoint-uri)

12. Jurnalul conexiunilor și jurnalul pe ecran

13. Diagnosticare și rezolvarea problemelor

14. Bune practici

15. Trimitere din baza de date (tabelul trimiterilor)

Anexa A. Exemple de configurare

1. ▪ Ce este programul

Albatros Bridge este o punte (bridge) între WhatsApp și o bază de date. Se conectează la WhatsApp prin cod QR, ca WhatsApp Web, și gestionează primirea, clasificarea, persistența și expunerea HTTP a datelor.

• Primire: text, imagini, video, audio, documente, stickere, locații, stări (status), reacții și mesaje tehnice.

• Mesaje modificate: gestionarea editărilor, inclusiv a noilor mesaje criptate LID atunci când messageSecret necesar este disponibil.

• Trimitere: mesaje text și media prin endpoint-uri HTTP, cu verificarea datelor primite.

• Trimitere din baza de date: un alt program scrie rânduri semnate în tabelul hh_Albatros_SendQueue, iar bridge-ul le trimite (vedeți 15).

• Bază de date: SQL Server, MySQL, MariaDB, SQLite prin ODBC sau SQLite încorporat.

• Mai multe instanțe: fiecare instanță folosește un folder propriu și poate avea propria sesiune WhatsApp.

• Cozi persistente: datele rămân pe disc dacă baza de date nu este disponibilă; scrierea pe disc este protejată împotriva întreruperilor bruște, iar o salvare repetată nu creează duplicate.

• Ordine cronologică la pornire: mesajele primite înainte ca numărul conectat să fie disponibil sunt puse în coadă pe disc și înregistrate, de la cel mai vechi la cel mai nou, imediat ce numărul este cunoscut.

• Media fiabile: descărcarea este reîncercată automat și există o limită de dimensiune; dacă fișierul media nu este disponibil, mesajul este salvat oricum.

• Securitate HTTP: adresă de ascultare configurabilă (BindAddress) și autentificare cu token (ApiToken).

• Sesiune WhatsApp protejată: fișierele sesiunii sunt întotdeauna criptate și legate de PC-ul pe care este instalat programul.

• Reconectare automată la WhatsApp cu așteptări progresive.

• Jurnale: jurnal de diagnosticare în tabel, jurnal de conexiune și copie a ieșirii text.

• Executabil post-inserare: opțional, după o salvare reușită.

2. ✓ Cerințe

Cerință	Detalii
Sistem de operare	Windows pe 64 de biți.
Bază de date prin ODBC	Driver ODBC pe 64 de biți instalat pentru SQL Server, MySQL, MariaDB sau SQLite extern.
SQLite încorporat	InternalSqlite: nu necesită un driver ODBC pentru baza de date.
Telefon	WhatsApp activ și disponibil pentru scanarea codului QR la prima conectare.
Rețea	Acces la Internet către serverele WhatsApp. Dacă clientul HTTP se află pe un alt PC, portul Port trebuie să fie accesibil (Paravanul de protecție Windows) și se recomandă setarea ApiToken.
Client HTTP	Orice program capabil să trimită cereri HTTP cu date JSON (de exemplu, un client VB.NET).

Cerință	Detalii
Windows PowerShell	Windows PowerShell 5.1 (inclus în Windows) prezent și utilizabil de contul care rulează programul, cu serviciul WMI activ și fără politici de grup care să blocheze executarea scripturilor PowerShell. Este indispensabil pentru criptarea parolei bazei de date și a sesiunii WhatsApp: fără PowerShell, cheia legată de PC nu poate fi calculată corect și stabil.

3. Parametri din linia de comandă

Parametru	Efect	Pornește bridge-ul?
--h / --help	Afișează ajutorul pentru parametri și se încheie.	Nu
--version	Afișează bannerul și numărul versiunii și se încheie.	Nu
--create_config <fișier>	Creează un fișier de configurare exemplu, cu toți parametrii comentați. Nu suprascrie un fișier existent.	Nu
--config <fișier|folder>	Folosește fișierul indicat sau Instance_Config.ini din folderul indicat.	Da
--pwd	Cere parola de două ori, cu introducere ascunsă, și generează PwdConnStr=ENC:...; aceeași valoare criptată este propusă și ca DbSendSecret=ENC:... (vedeți 15).	Nu
--pwd=<parolă>	Generează direct PwdConnStr criptat din valoarea indicată.	Nu
--man_IT [=<fișier|folder>]	Salvează manualul de utilizare în italiană (.docx) inclus în program. Fără cale, îl salvează în folderul curent; cu un folder, îl salvează în interiorul acestuia; altfel, cu numele indicat. Nume implicit: Manuale_Albatros_Bridge_<versione>.docx. Nu suprascrie un fișier existent.	Nu
--man_EN [=<fișier|folder>]	Ca --man_IT, pentru manualul de utilizare în engleză. Nume implicit: Manual_Albatros_Bridge_<versione>_EN.docx.	Nu
--Skip-Manuals	Nu salvează automat manualele de utilizare în folderul programului la pornire (vedeți 8). Echivalent cu Skip-Manuals=Y în fișierul de configurare.	Da
niciun parametru	Pornire standard cu căutarea automată a configurației.	Da

Parametrii pot fi indicați în forma cu spațiu sau cu „=” atunci când este prevăzut. Numele nu fac distincție între majuscule și minuscule. Căile încadrate între ghilimele sunt normalizate.

```
Albatros-Bridge-odbc.exe --help
Albatros-Bridge-odbc.exe --version
Albatros-Bridge-odbc.exe --config "C:\Albatros\Instance01"
Albatros-Bridge-odbc.exe --config="C:\Albatros\Instance01\Instance_Config.ini"
Albatros-Bridge-odbc.exe --create_config="C:\tmp\MyConfig.ini"
Albatros-Bridge-odbc.exe --pwd
Albatros-Bridge-odbc.exe --pwd=MiaPassword123
Albatros-Bridge-odbc.exe --man_IT
Albatros-Bridge-odbc.exe --man_EN="C:\Doc\"
Albatros-Bridge-odbc.exe --Skip-Manuals
```

Variabilă de mediu: PHONES_BRIDGE_CONFIG poate indica fișierul sau folderul de configurare atunci când --config nu este prezent.

4. Fișiere și foldere principale

Structura este organizată sub PathWork. Fiecare instanță este separată prin prefixul Instance_ și valoarea CodeInstance.

Element	Descriere
Instance_Config.ini	Configurația instanței.

Element	Descriere
Manuale_Albatros_Bridge_<versione>.docx, Manual_Albatros_Bridge_<versione>_EN.docx	Manualele de utilizare în italiană și în engleză, salvate automat în folderul programului la pornire, dacă nu există (vedeți 8).
PathWork	Folderul rădăcină de lucru. Implicit: c:\Albatros\Phone\.
Instance_<CodeInstance>\	Folderul instanței.
...\QrCode\latest.png	Codul QR curent, numai atunci când codul QR este efectiv afișat.
...\auth_info\	Credențialele sesiunii WhatsApp, întotdeauna criptate și legate de PC (vedeți 7.1). Nu trebuie folosit de două instanțe simultan.
...\auth_info.undecryptable_<data>\	Sesiune anterioară care nu poate fi decriptată pe acest PC (vedeți 7.1): mutată deoparte, nu ștearsă.
...\pending_db\	Coadă persistentă a datelor încă nesalvate (fișiere *_M.json pentru mesaje, *_S.json pentru messageSecret).
...\pending_db*.json.tmp	Fișier temporar folosit în timpul scrierii cozii; dacă rămâne după o întrerupere bruscă, este șters la pornire.
...\pending_db\failed\	Elemente care au depășit limitele de reîncercări/timp sau care nu pot fi salvate (fișier ilizibil sau date nevalide).
...\early_queue\	Coadă de pornire: mesaje și evenimente primite înainte ca numărul conectat să fie disponibil, în așteptarea înregistrării (vedeți 10.6).
...\early_queue\failed\	Elemente ale cozii de pornire ilizibile sau care nu pot fi procesate (vedeți 10.6).
...\Albatros_Embedded.db	Baza de date SQLite încorporată, când este folosită fără ConnStr.
PathWork\Instances_Log\<număr>\Connection.log	Jurnalul conexiunii și copia ieșirii text.
%TEMP%\AlbatrosBridgeOdbc\	Zonă temporară pentru ODBC-ul încorporat în executabil.

5. Fișierul de configurare Instance_Config.ini

Fișierul folosește câte o pereche Cheie=Valoare sau Cheie:Valoare pe rând. Rândurile care încep cu # (diez) sau ; (punct și virgulă) sunt comentarii. Cheile nu fac distincție între majuscule și minuscule.

Parametru	Implicit / exemplu	Descriere
Key	Albatros	Eticheta instanței (maximum 50 de caractere).
TypeDB	InternalSqlite	Tipul bazei de date: MySql, SqlServer, MariaDB, SqLite, InternalSqlite.
ConnStr	gol	Șir de conexiune (connection string); pentru InternalSqlite poate fi omis.
PwdConnStr	gol	Parola bazei de date, în clar sau ENC:.... Ignorată cu SQLite încorporat.
PathWork	c:\Albatros\Phone\	Folderul de lucru rădăcină.
CodeInstance	Default	Codul instanței (este acceptat și CodeIstance).
PathAuth	automat	Folder alternativ pentru credențialele WhatsApp.
PathBin	c:\Albatros\Phone\bin\	Folderul executabilului post-inserare (este acceptat și bin).
ExeName	gol	Executabil lansat după salvarea reușită, cu parametrii --rec, --istance și --Cell (vedeți 10.1). Este acceptat și Exe.
Save_Status	N	Salvează actualizările de livrare/citire.
Save_StatusBroadcast	N	Salvează stările/poveștile status@broadcast. Cu N nu sunt salvate nici messageSecret-ul lor, nici modificările și ștergerile.
Save_Sent	S	Salvează mesajele trimise de numărul conectat. Cu N nu sunt salvate nici messageSecret-ul lor, nici modificările și ștergerile făcute de numărul conectat.
Save_protocolMessage	N	Salvează mesajele tehnice de protocol.
Save_Secrets	S	Salvează messageSecret, necesare pentru editările criptate. Nu sunt salvate cele ale mesajelor excluse de Save_Sent, Save_StatusBroadcast și Save_protocolMessage.
ConsoleLog	N	Activează pe ecran mesajele informative/de depanare; bannerul, rezumatul, codul QR, avertismentele și erorile rămân vizibile.

Parametru	Implicit / exemplu	Descriere
DebugLog	gol	Activează jurnalul de diagnosticare în hh_Albatros_Webhooks_log pentru INFO/WARNING/ERROR/DEBUG.
DebugLogFilter	gol	Filtrează doar conținutul multimedia prezent în DebugData.
DebugLogMaxRecords	10000	Numărul maxim de înregistrări păstrate în tabelul hh_Albatros_Webhooks_log. Valoare goală sau 0 = nicio limită.
DbRetrySec	30	Baza backoff-ului liniar către baza de date.
DbRetryMaxSec	300	Plafonul maxim al așteptării între reîncercări (maximum 86400).
DbMaxRetries	0	Numărul maxim de încercări per element; 0 = nelimitat.
PendingMaxSec	172800	Timpul maxim în pending; 0 = nicio limită de timp.
Port	3000	Portul HTTP.
BindAddress	gol	Adresa pe care ascultă serverul HTTP. Gol = toate interfețele de rețea (ca în versiunile anterioare); 127.0.0.1 = doar acest PC. Vedeți 5.3.
ApiToken	gol	Token cerut de toate endpoint-urile HTTP. Gol = nicio autentificare. Vedeți 5.3.
MaxMediaMB	100	Dimensiunea maximă în MB a unui fișier media de descărcat; 0 = nicio limită. Vedeți 5.4.
MaxMemQueue	500	Numărul maxim de elemente în cozile HTTP din memorie; 0 dezactivează cozile.
SendMinIntervalSec	3	Numărul minim de secunde între două trimiteri (și zecimale); intervalul este prelungit aleatoriu cu până la 30%. 0 = nicio așteptare. Vedeți 5.5.
SendMaxPerHour	60	Numărul maxim de trimiteri în ultima oră (fereastră glisantă); 0 = niciun plafon. Vedeți 5.5.
SendMaxWaitSec	60	Așteptarea maximă a unei cereri în coada de trimitere, apoi răspuns 429 (maximum 3600); 0 = nicio așteptare. Vedeți 5.5.
SendCheckNumber	Y	Y = înainte de trimitere verifică dacă numărul folosește WhatsApp; dacă nu îl folosește, răspuns 422. Vedeți 5.5.
SendIdempotencyHours	24	Orele de valabilitate ale clientMsgId (maximum 720); 0 = clientMsgId ignorat. Vedeți 5.5.
Skip-Manuals	N	Y = la pornire nu salvează automat manualele de utilizare în folderul programului. Echivalent cu parametrul --Skip-Manuals. Vedeți 8.
DbSend	N	Y = citește tabelul hh_Albatros_SendQueue și trimite rândurile numărului conectat, cu aceleași limite ca trimiterile HTTP. Vedeți 15.
DbSendSecret	gol	Secretul semnăturii rândurilor (minimum 32 de caractere), în clar sau ENC:... generat cu --pwd. Fără un secret valid, DbSend rămâne dezactivat. Vedeți 15.
DbSendPollSec	5	Secunde între două citiri ale tabelului (de la 1 la 300). POST /send-queue/poll anticipează citirea. Vedeți 15.
DbSendMaxAgeMin	60	Minute de valabilitate ale unui rând de la CreatedAtUtc: valorile sub 30 devin 30, cele peste 1440 (24 de ore) devin 1440. Vedeți 15.

Valori booleene acceptate: S, Y, SI, SÌ, YES, TRUE, 1 (și cu minuscule). Celelalte valori, inclusiv N sau gol, echivalează cu Nu.

5.1 DebugLog și DebugLogFilter

DebugLog privește scrierea în tabelul hh_Albatros_Webhooks_log. DebugLogFilter nu decide ce rânduri sunt scrise: acționează exclusiv asupra conținutului multimedia prezent în câmpul DebugData.

Setare	Rezultat
DebugLog=All	INFO, WARNING, ERROR, DEBUG. All are prioritate față de celelalte valori.
DebugLog=Debug, Error	Doar DEBUG și ERROR.
DebugLog gol / N / NO / FALSE / 0	Jurnalul de diagnosticare în tabel este dezactivat.
DebugLogFilter=No-Video	Elimină fișierele video din DebugData.

Setare	Rezultat
No-Audio	Elimină fișierele audio din DebugData.
No-PDF	Elimină fișierele PDF din DebugData.
No-Other	Elimină celelalte fișiere media din DebugData.
No-Media	Exclude toate fișierele media din DebugData.

```
DebugLog=All
DebugLogFilter=No-Video, No-Other
ConsoleLog=Y
```

Jurnalul DEBUG al unui mesaj este scris o singură dată, chiar dacă salvarea în baza de date este reîncercată de mai multe ori.

5.2 DebugLogMaxRecords

Limitează numărul de rânduri păstrate în hh_Albatros_Webhooks_log. După fiecare scriere reușită a jurnalului, rândurile cele mai vechi sunt șterse, într-o tranzacție separată: dacă curățarea nu reușește, rândul tocmai scris rămâne oricum salvat, iar curățarea este reîncercată la următoarea scriere în jurnal. Implicit 10000; valoare goală, 0 sau nevalidă = nicio limită.

5.3 Securitatea interfeței HTTP: BindAddress și ApiToken

Endpoint-urile HTTP permit trimiterea de mesaje de pe numărul conectat, citirea mesajelor primite și descărcarea codului QR de conectare. De aceea este important să limitați cine le poate accesa.

Situație	Configurație recomandată
Client pe același PC cu bridge-ul	BindAddress=127.0.0.1. În client folosiți http://127.0.0.1:<Port> (nu localhost, care pe Windows poate fi rezolvat mai întâi ca adresă IPv6 ::1). ApiToken opțional.
Client pe un alt PC sau pe o mașină virtuală	BindAddress gol (sau 0.0.0.0 ori adresa IP a plăcii de rețea), ApiToken obligatoriu și o regulă în Paravanul de protecție Windows care deschide portul doar pentru adresele clienților.

Dacă ApiToken este setat, **toate** endpoint-urile (inclusiv /qr) cer token-ul, trimis în unul dintre cele două moduri de mai jos; în caz contrar, răspunsul este HTTP 401. Valoarea token-ului nu este afișată niciodată de /config.

```
Authorization: Bearer <token>
X-Api-Key: <token>
```

Folosiți un token lung și aleatoriu. Pentru a genera unul cu PowerShell:

```
$b = New-Object byte[] 32
[System.Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($b)
[Convert]::ToBase64String($b)
```

Exemplu de regulă de paravan de protecție care deschide portul 3000 doar pentru PC-ul clientului (adresă de exemplu):

```
New-NetFirewallRule -DisplayName "Albatros Bridge 3000" -Direction Inbound `
    -Protocol TCP -LocalPort 3000 -RemoteAddress 192.168.1.20 -Action Allow
```

Dacă serverul este accesibil din rețea (BindAddress gol sau nelocal) și ApiToken nu este setat, la pornire este afișat avertismentul SECURITY ca ultim rând al rezumatului.

5.4 MaxMediaMB

Limitează dimensiunea fișierelor media descărcate. Verificarea are loc înainte de descărcare, pe dimensiunea declarată de WhatsApp, și din nou pe dimensiunea reală. Un fișier media peste limită nu este descărcat: mesajul este salvat fără atașament și în jurnal este înregistrată o eroare. Implicit 100 MB; 0 = nicio limită. O valoare prea mare crește utilizarea memoriei și a spațiului din coada de pe disc, deoarece fișierul media este păstrat în pending până la salvarea în baza de date.

5.5 Limite și verificări ale trimiterilor

Limitele reduc riscul ca numărul să fie blocat din cauza trimiterilor prea rapide sau prea numeroase, dar nu fac permisă utilizarea programului pentru spam, publicitate sau trimiteri în masă (vedeți avertismentele inițiale).

Coada de trimitere. Cererile către /send și /send-media intră într-o coadă și pleacă în ordinea sosirii, distanțate cu cel puțin SendMinIntervalSec secunde; fiecare interval este prelungit aleatoriu cu până la 30% pentru a evita un ritm fix. Într-o oră nu pleacă mai mult de SendMaxPerHour mesaje. Cererea HTTP își așteaptă rândul și primește răspunsul după efectuarea trimiterii. Dacă rândul ar veni după mai mult de SendMaxWaitSec secunde, bridge-ul răspunde imediat 429 cu câmpul retryAfterSec și header-ul Retry-After, care indică după câte secunde să reîncercați. SendMaxWaitSec trebuie să rămână sub timeout-ul clientului HTTP (în .NET, valoarea implicită pentru HttpClient este de 100 de secunde). Dacă SendMinIntervalSec și SendMaxPerHour sunt ambele 0, la pornire este afișat un avertisment SECURITY.

Verificarea numărului. Cu SendCheckNumber=Y, înainte de a pune mesajul în coadă, bridge-ul verifică dacă destinatarul folosește WhatsApp; dacă nu îl folosește, răspunde 422 (code NOT_ON_WHATSAPP), iar mesajul nu consumă din cota orară. Rezultatul este reținut 24 de ore dacă numărul există și 10 minute dacă nu există. Grupurile (@g.us) și identificatorii @lid nu sunt verificați. Dacă verificarea nu reușește din motive tehnice (de exemplu, un timeout), mesajul este trimis oricum și în jurnal apare un avertisment.

Trimitere fără duplicate (clientMsgId). Clientul poate indica în JSON un identificator unic al mesajului, clientMsgId (de la 1 la 100 de caractere), care trebuie reutilizat doar atunci când repetă aceeași cerere, de exemplu după un timeout. Dacă mesajul cu acel clientMsgId a fost deja trimis cu succes în ultimele SendIdempotencyHours ore, bridge-ul nu îl retrimite și returnează același răspuns cu duplicate=true; dacă prima trimitere este încă în curs, îi așteaptă rezultatul. Dacă același clientMsgId sosește cu un destinatar sau un conținut diferit, răspunsul este 409. Trimiterile nereușite nu sunt reținute, deci cererea poate fi repetată cu același clientMsgId. Identificatorii sunt păstrați în memorie și se resetează la repornirea programului.

/status raportează starea trimiterilor în blocul send: mesaje trimise, trimiteri din ultima oră, cereri în așteptare, secunde până la următoarea trimitere posibilă, cereri refuzate din cauza limitei sau a numărului neînregistrat, duplicate.

6. Configurarea bazei de date

Sunt acceptate SQL Server, MySQL, MariaDB, SQLite prin ODBC și SQLite încorporat.

```
TypeDB=InternalSqlite
#ConnStr=
# senza ConnStr:
# PathWork\Instance_<CodeInstance>\Albatros_Embedded.db
```

Cu InternalSqlite nu este necesar să instalați un driver ODBC pentru baza de date. Dacă fișierul este ocupat de un alt program, bridge-ul efectuează câteva reîncercări rapide și păstrează datele în coadă.

Tabel	Conținut
hh_Albatros_Webhooks	Antetul mesajelor/evenimentelor și RawData integral.
hh_Albatros_WebHooks_Media	Atașamentele multimedia asociate antetului.
hh_Albatros_WebHooks_Locations	Locația (latitudine și longitudine) a mesajelor de tip locație, legată de antet.
hh_Albatros_WebHooks_Secrets	messageSecret al mesajelor. Numărul conectat și Key nu sunt memorate în acest tabel: se obțin printr-un JOIN cu hh_Albatros_Webhooks prin MsgID.
hh_Albatros_Webhooks_log	Jurnal de diagnosticare.
hh_Albatros_SendQueue	Mesaje de trimis scrise de un alt program (trimitere din baza de date). Creat doar cu DbSend=Y; vedeți 15.

Tabelele sunt create automat dacă lipsesc. La pornire, programul creează și indexul IX_hh_Albatros_Webhooks_MsgID pe coloana MsgID a antetului, dacă nu există deja, pentru a accelera căutarea mesajelor originale (necesară pentru editări). Pe un tabel foarte mare, crearea poate dura câteva minute, timp în care mesajele rămân în coadă. Dacă utilizatorul bazei de date nu are permisiunea de a crea indecși, este afișat doar un avertisment și programul continuă normal.

Fiecare mesaj primește un identificator unic (MsgUID) în momentul în care intră în coadă. Dacă o salvare a fost deja finalizată, dar programul s-a întrerupt înainte de a șterge fișierul din coadă, la noua încercare rândul este recunoscut și nu este inserat a doua oară.

7. Parola bazei de date și sesiunea WhatsApp criptate

Parola criptată este legată de elemente hardware ale PC-ului. Pentru a o genera:

```
Albatros-Bridge-odbc.exe --pwd
```

Parola este cerută de două ori, cu introducere ascunsă, atunci când PowerShell este disponibil. La final este produs un rând PwdConnStr=ENC:... care trebuie copiat în fișierul de configurare. O parolă criptată pe un PC nu poate fi folosită pe un alt PC (nici după înlocuirea plăcii de bază, a procesorului sau a discului de sistem): în aceste cazuri trebuie regenerată. Aceeași valoare criptată poate fi folosită pentru DbSendSecret: programul o propune și în rândul DbSendSecret=ENC:... (vedeți 15).

7.1 Criptarea sesiunii WhatsApp

Fișierele sesiunii WhatsApp, din folderul auth_info, sunt întotdeauna criptate cu o cheie legată de elementele hardware ale PC-ului, diferită de cea folosită pentru parola bazei de date. Criptarea este automată, mereu activă și neconfigurabilă: nu necesită nicio intervenție.

• Cine copiază folderul auth_info pe un alt PC nu îl poate folosi pentru a se conecta cu numărul WhatsApp al instanței.

• Actualizare de la o versiune anterioară: la prima pornire, fișierele existente în clar sunt criptate automat, fără o nouă scanare a codului QR.

• Schimbarea PC-ului sau a hardware-ului (placă de bază, procesor, discul C sau o mașină virtuală mutată pe o gazdă cu alt procesor): sesiunea nu mai poate fi decriptată și trebuie reconectată cu un nou cod QR.

• În acest caz, folderul nu este șters: este redenumit în auth_info.undecryptable_<dată și oră> și este creată o nouă sesiune. După ce ați verificat noua conectare, folderul redenumit poate fi șters; pe telefon, în Dispozitive asociate, este bine să eliminați dispozitivul anterior.

• Parametrul EncryptAuth nu este acceptat: dacă este prezent în fișierul de configurare, este ignorat cu un avertisment.

La pornire, citirea elementelor hardware poate dura câteva secunde. Dacă prima verificare nu reușește, programul o repetă înainte de a considera sesiunea imposibil de decriptat.

Criptarea împiedică utilizarea sesiunii pe un alt PC, dar nu înlocuiește protecția PC-ului însuși: cine accesează PC-ul cu permisiunile contului care rulează bridge-ul poate folosi sesiunea. Prin urmare, protejați accesul la server.

8. Pornirea programului

Căutarea configurației: --config; PHONES_BRIDGE_CONFIG; Instance_Config.ini în folderul programului; calea istorică; în final, valorile implicite, cu avertisment.

La pornire sunt afișate întotdeauna bannerul și rezumatul configurației, care include și BindAddress, ApiToken (doar dacă este setat sau nu, niciodată valoarea), MaxMediaMB, parametrii trimiterilor (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec, SendCheckNumber, SendIdempotencyHours), trimiterea din baza de date (DbSend și, dacă este activă, DbSendSecret: doar dacă este criptat sau în clar, niciodată valoarea) și rândul Session files, care confirmă criptarea sesiunii. Rezumatul se încheie cu DebugLog, DebugLogFilter, indicarea prin excludere a fișierelor media acceptate și, dacă condițiile o cer, avertismentul SECURITY.

```
MaxMediaMB            : 100 MB
MaxMemQueue           : 1500
SendMinIntervalSec    : 3s between two sends (+ up to 30% random)
SendMaxPerHour        : 60 sends per hour
SendMaxWaitSec        : 60s maximum wait in the send queue, then HTTP 429
SendCheckNumber       : Y (recipient verified before sending)
```

```
   SendIdempotencyHours  : 24h (repeated clientMsgId is not sent again)
   DbSend                : Y (table hh_Albatros_SendQueue, read every 5s, rows valid 60 minutes)
   DbSendSecret          : set (encrypted, bound to this PC)
   ConsoleLog            : Y (console debug enabled)
   DebugLog              : All (DEBUG, ERROR, INFO, WARNING)
   DebugLogFilter        : NO-VIDEO, NO-OTHER
   Accepted media        : AUDIO, PDF
⚠ SECURITY: the HTTP server is reachable from the network without ApiToken: ...
```

Rezumatul indică și folderul cozii de pornire (Startup queue folder), descris în 10.6.

Manuale de utilizare: la pornire, manualele în italiană și în engleză sunt salvate în folderul programului, cu aceleași nume implicite ca la --man_IT și --man_EN, doar dacă nu există deja: un fișier existent nu este niciodată suprascris. Pentru fiecare fișier creat este afișat un rând informativ; dacă scrierea nu este posibilă, apare un avertisment și programul continuă. Salvarea automată se dezactivează cu parametrul --Skip-Manuals sau cu Skip-Manuals=Y în fișierul de configurare.

Dacă apare o eroare neprevăzută netratată, programul o scrie în jurnal (pe ecran, în Connection.log și în tabelul de jurnal) și se încheie cu codul de ieșire 1 după aproximativ 2 secunde. Pentru o funcționare continuă, se recomandă rularea lui ca serviciu Windows configurat pentru repornire automată în caz de eroare.

9. Prima conectare la WhatsApp

• Dacă nu există credențiale valide, este afișat codul QR.

• Același cod QR este salvat în QrCode\latest.png și este disponibil și prin endpoint-ul /qr (cu token-ul, dacă ApiToken este setat).

• Pe telefon: Setări → Dispozitive asociate → Asociază un dispozitiv.

• După scanare, WhatsApp cere o repornire a conexiunii: bridge-ul se reconectează imediat.

• Când conexiunea este deschisă, fișierul QR este șters.

• Fișierele noii sesiuni sunt create deja criptate (vedeți 7.1).

• În timpul unei reconectări automate, codul QR nu este afișat și nici salvat dacă nu este realmente necesar.

• După o deconectare (logout), sesiunea este eliminată și codul QR este cerut din nou.

10. Funcționare generală

Fiecare mesaj/eveniment este clasificat, eventual asociat unui mesaj anterior, serializat și introdus în fluxul de salvare. Fiecare element trece prin coada persistentă de pe disc, iar fișierul este șters doar după salvarea reușită în baza de date.

10.1 Reîncercări și cozi

• Backoff liniar cu DbRetrySec și plafonul DbRetryMaxSec; orice dată nouă provoacă oricum o încercare imediată.

• Primul acces reușit la baza de date resetează contorul de încercări.

• Fișierele din coadă sunt scrise în mod sigur (fișier temporar și redenumire): o întrerupere bruscă nu lasă fișiere incomplete.

• Erorile de disponibilitate a bazei de date (rețea, server oprit, bază de date ocupată) sunt reîncercate conform DbMaxRetries și PendingMaxSec.

• Un fișier din coadă ilizibil sau o dată refuzată de baza de date (de exemplu, o valoare nevalidă) este mutat imediat în pending_db\failed, fără a bloca elementele următoare.

• Elementele care depășesc limitele sunt mutate în pending_db\failed; cauza este afișată pe ecran și în jurnal.

• După salvarea reușită a unui mesaj, poate fi pornit ExeName (vedeți mai jos în acest paragraf).

Pentru a reîncerca un element ajuns în failed, după ce i-ați corectat cauza, mutați fișierul din pending_db\failed în pending_db și reporniți bridge-ul: fișierele prezente în pending_db sunt încărcate la pornire.

Executabil post-inserare. Dacă ExeName este setat, după fiecare mesaj salvat în baza de date (nu pentru messageSecret), bridge-ul pornește PathBin\ExeName fără să aștepte ca acesta să se încheie, transmițând trei parametri separați:

```
--rec=<ID>          ID della riga in hh_Albatros_Webhooks
--istance=<istanza> codice istanza (CodeInstance) che ha registrato la riga
--Cell=<numero>     numero collegato all'istanza (PhoneConnected), solo cifre
```

Exemplu de linie de comandă primită de programul pornit:

```
GestioneMessaggi.exe --rec=45 --istance=Magazzino --Cell=393479528620
```

Valorile sunt cele înregistrate în rând (coloanele ID, Instance și PhoneConnected). Un cod de instanță care conține spații ajunge oricum ca parametru unic, între ghilimele. Dacă executabilul nu există, este afișat o singură dată un avertisment, iar salvările continuă normal; rezumatul de la pornire afișează, lângă Exe, formatul parametrilor. În versiunile anterioare, singurul parametru era ID-ul înregistrării.

10.2 Mesaje modificate

Cu Save_Secrets=S, messageSecret este capturat înaintea filtrelor de salvare ale antetului, cu excepția mesajelor excluse prin configurație (Save_Sent=N, Save_StatusBroadcast=N, Save_protocolMessage=N); modificările și ștergerile mesajelor excluse de Save_Sent și Save_StatusBroadcast nu sunt salvate. Logica actuală gestionează și editările criptate LID și, când este necesar, recuperează mesajul original căutându-l în baza de date (ignorând rândurile de actualizare a stării, care au același MsgID) sau, dacă nu a fost încă salvat, în coada de pe disc.

10.3 RelatedMsgID

Eveniment	Sursă
Răspuns / citare	contextInfo.stanzaId
Reacție	reactionMessage.key.id
Protocol	protocolMessage.key.id
Modificare	targetMessageKey.id

10.4 Fișiere media și locații

• Descărcarea unui fișier media este încercată de până la 3 ori, cu scurte așteptări între încercări.

• Dacă descărcarea nu reușește sau fișierul media depășește MaxMediaMB, mesajul este salvat oricum, fără atașament (MediaCount=0); în jurnal este înregistrată o eroare, iar elementul returnat de /messages conține câmpul mediaError cu cauza.

• O locație cu coordonate nevalide nu duce la pierderea mesajului: acesta este salvat fără rândul din hh_Albatros_WebHooks_Locations și cu câmpul locationError; coordonatele originale rămân în RawData.

• Contactele și sondajele sunt înregistrate cu tipul contact (text = numele afișat) și poll (text = întrebarea); conținutul complet rămâne în RawData.

• În cazul altor erori în timpul procesării, mesajul nu este considerat „deja văzut”: o nouă livrare din partea WhatsApp este, așadar, procesată normal.

10.5 Reconectare automată

Cauza închiderii	Comportament
Rețea absentă sau conexiune pierdută	Nouă încercare cu așteptare crescătoare: aproximativ 1, 2, 4, 8 ... secunde, până la maximum 60 de secunde. Contorul se resetează când conexiunea se redeschide.
Repornire cerută de WhatsApp (după scanarea codului QR)	Reconectare imediată.
Sesiune înlocuită	O altă instanță folosește aceleași credențiale: este afișată o eroare, iar încercarea următoare are loc după 60 de secunde.
Deconectare (logout) de pe telefon	Credențialele sunt șterse și este cerut un nou cod QR.

În timpul așteptării, /status raportează connected=false, numărul încercării (reconnectAttempt) și secundele rămase (reconnectInSec). Cererile de trimitere primite în această fază primesc răspunsul HTTP 503.

10.6 Mesaje primite înainte ca numărul conectat să fie disponibil

La pornire, dacă există mesaje în așteptare pe server, WhatsApp le poate livra înainte ca numărul conectat să fie disponibil. Pentru a nu le înregistra cu câmpul PhoneConnected gol, bridge-ul le pune în coada de pornire și le înregistrează imediat ce numărul este cunoscut.

• Fiecare eveniment primit în această fază (mesaj, modificare, ștergere, actualizare de stare) este scris într-un fișier din folderul early_queue al instanței, în mod protejat împotriva întreruperilor bruște. Dacă programul este închis, la pornirea următoare fișierele prezente sunt reîncărcate și rămân în așteptarea numărului conectat.

• Imediat ce numărul este disponibil, evenimentele sunt înregistrate unul câte unul, de la cel mai vechi la cel mai nou, conform datei/orei interne WhatsApp. În aceeași secundă, mesajul precede modificarea sa, iar modificarea precede ștergerea; în orice caz, modificările, ștergerile, reacțiile și actualizările de stare sunt înregistrate după mesajul la care se referă, astfel încât messageSecret al mesajului original este deja disponibil când se înregistrează modificarea.

• Și evenimentele care sosesc în timp ce coada este golită sunt puse în coadă: niciun eveniment nu poate fi înregistrat înaintea unuia mai vechi.

• Fișierul unui eveniment este șters doar după înregistrare. Dacă programul se întrerupe după înregistrare, dar înainte de ștergere, la repornire evenimentul este procesat din nou fără a crea rânduri duplicate.

• Un fișier ilizibil sau un eveniment care nu a putut fi procesat este mutat în early_queue\failed; cauza este afișată pe ecran și în jurnal.

• Dacă așteptarea depășește 30 de secunde, este afișat un avertisment cu numărul de evenimente din coadă, repetat la fiecare 60 de secunde. /status raportează numărul de evenimente din coadă în câmpul startupQueue.

• Pentru rândurile înregistrate din coadă, DateReceived corespunde momentului înregistrării; data/ora WhatsApp (DateWA) rămâne cea originală.

• Dacă un eveniment rămâne mult timp în coada de pornire, fișierul media ar putea să nu mai poată fi descărcat, deoarece linkurile WhatsApp expiră: în acest caz, mesajul este salvat fără atașament, așa cum este descris în 10.4.

11. Interfața HTTP (endpoint-uri)

Adresă de bază: http://<adresă>:<Port> — portul implicit 3000. Dacă ApiToken este setat, fiecare cerere trebuie să conțină token-ul (vedeți 5.3). Datele trimise cu POST sunt în format JSON (Content-Type: application/json).

Metodă	Endpoint	Descriere
GET	`/status`	Starea conexiunii, instanța, baza de date, coada, coada de pornire (startupQueue), reconectarea, trimiterile (send), trimiterea din baza de date (sendQueue) și statistici.
GET	`/qr`	Codul QR curent; 404 dacă nu este disponibil.
GET	`/messages`	Returnează și golește coada de mesaje.

Metodă	Endpoint	Descriere
GET	/messages?peek=1	Returnează coada de mesaje fără a o goli; fiecare element are numărul seq.
POST	/messages/ack	Confirmă mesajele procesate: {"upToSeq": N} le elimină pe cele cu seq până la N.
GET	/message-status	Returnează și golește actualizările de stare.
GET	/message-status?peek=1	Ca mai sus, fără golire (cu seq).
POST	/message-status/ack	Confirmă actualizările de stare procesate ({"upToSeq": N}).
GET	/debug-log	Starea jurnalului de diagnosticare.
GET	/db-queue	Coada, reîncercările și parametrii.
POST	/db-retry	Forțează o încercare imediată.
GET	/config	Configurația activă, fără parolă și fără token.
GET	/message-secret/:msgId	MessageSecret al mesajului.
POST	/send	Trimite text (limite de trimitere, verificarea numărului și clientMsgId opțional: vedeți 5.5).
POST	/send-media	Trimite fișiere media prin base64 (aceleași verificări ca /send).
POST	/send-queue/poll	Anticipează citirea tabelului trimiterilor (vedeți 15). Răspuns 202; 409 DB_SEND_DISABLED dacă trimiterea din baza de date nu este activă.

11.1 Citirea mesajelor fără pierderi (peek / ack)

Citirea clasică a /messages golește coada în momentul în care răspunde: dacă clientul se întrerupe înainte de a procesa răspunsul, acele mesaje nu mai sunt returnate (rămân totuși salvate în baza de date). Citirea în două faze evită problema:

• GET /messages?peek=1: returnează mesajele din coadă, fiecare cu un număr progresiv seq, fără a le elimina.

• Clientul procesează mesajele.

• POST /messages/ack cu {"upToSeq": <seq al ultimului mesaj procesat>}: elimină din coadă mesajele confirmate. Dacă clientul cade înainte de confirmare, le regăsește la ciclul următor.

```
GET  /messages?peek=1
{ "key": "Albatros", "messages": [ { "id": "3EB0...", "type": "text", "text": "ciao", ..., "seq": 41 },
                             { "id": "3EB0...", "type": "image", ..., "seq": 42 } ] }

POST /messages/ack        { "upToSeq": 42 }
{ "success": true, "removed": 2, "remaining": 0 }
```

Coada rămâne oricum limitată la MaxMemQueue elemente: dacă clientul nu confirmă mult timp, elementele cele mai vechi sunt eliminate. Aceeași schemă este valabilă pentru /message-status și /message-status/ack. Nu folosiți simultan citirea clasică și citirea în două faze de la clienți diferiți.

11.2 Trimiterea mesajelor

Destinatarul (number) poate fi un număr cu prefix internațional, chiar și cu +, spații, cratime sau paranteze, ori un identificator WhatsApp complet (@s.whatsapp.net, grup @g.us, @lid).

```
POST /send
{ "number": "+39 340 123 4567", "message": "Testo del messaggio",
  "clientMsgId": "ORD-2026-000123" }

POST /send-media
{ "number": "393401234567", "mediaType": "document", "fileName": "fattura.pdf",
  "caption": "Fattura di ottobre", "base64Data": "JVBERi0xLjcK..." }

Risposta: { "success": true, "id": "<id del messaggio inviato>" }
Con clientMsgId: { "success": true, "id": "<id>", "clientMsgId": "ORD-2026-000123",
  "duplicate": false }
```

```
Errore: { "success": false, "code": "SEND_HOURLY_LIMIT",
  "error": "Hourly send limit reached", "retryAfterSec": 1250 }
```

mediaType	Trimitere
image	Imagine, cu eventuală descriere (caption).
audio	Audio (trimis ca audio/mpeg).
orice altă valoare	Document: fileName (implicit file.pdf) și eventual caption.

Câmpul clientMsgId este opțional (vedeți 5.5). Răspunsurile de eroare de la /send și /send-media conțin, pe lângă error, câmpul code, pe care clientul îl poate folosi în locul textului:

code (HTTP)	Semnificație
INVALID_NUMBER (400)	number lipsă sau nevalid.
INVALID_MESSAGE (400)	message lipsă sau gol (/send).
INVALID_MEDIA (400)	base64Data lipsă sau gol (/send-media).
INVALID_CLIENT_MSG_ID (400)	clientMsgId nevalid (nu este text, este gol sau depășește 100 de caractere).
CLIENT_MSG_ID_CONFLICT (409)	clientMsgId deja folosit pentru un mesaj diferit.
NOT_ON_WHATSAPP (422)	Numărul nu folosește WhatsApp (SendCheckNumber=Y). Nu reîncercați.
SEND_HOURLY_LIMIT (429)	S-a atins SendMaxPerHour: reîncercați după retryAfterSec secunde.
SEND_QUEUE_BUSY (429)	Așteptare în coadă peste SendMaxWaitSec: reîncercați după retryAfterSec secunde.
SEND_ERROR (500)	Eroare returnată de WhatsApp în timpul trimiterii.
NOT_CONNECTED (503)	WhatsApp nu este conectat.

11.3 Coduri de răspuns

Cod	Semnificație
200	Cerere executată.
400	Date lipsă sau nevalide (de exemplu, number nevalid, message gol, base64Data absent, upToSeq nenumeric, JSON incorect format).
401	Token lipsă sau greșit (doar cu ApiToken setat).
404	Resursă indisponibilă (de exemplu, codul QR nu este prezent).
409	clientMsgId deja folosit pentru un mesaj diferit (/send, /send-media).
413	Corpul cererii este prea mare (peste 50 MB).
422	Numărul destinatar nu folosește WhatsApp (SendCheckNumber=Y).
429	Limita de trimitere a fost atinsă (SendMaxPerHour) sau așteptarea în coadă depășește SendMaxWaitSec: reîncercați după retryAfterSec secunde (header-ul Retry-After).
500	Eroare internă sau eroare returnată de WhatsApp în timpul trimiterii.
503	WhatsApp nu este conectat sau baza de date nu este disponibilă (/message-secret).

11.4 Exemplu de client C#.NET

```
using System;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;

var http = new HttpClient { BaseAddress = new Uri("http://127.0.0.1:3000/") };
http.DefaultRequestHeaders.Authorization =
    new AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale");

// Lettura senza svuotare la coda
string json = await http.GetStringAsync("messages?peek=1");
// ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

// Conferma dei messaggi elaborati
var ack = new StringContent("{\"upToSeq\":" + ultimoSeq + "}", Encoding.UTF8, "application/json");
HttpResponseMessage r = await http.PostAsync("messages/ack", ack);
```

11.5 Exemplu de client VB.NET

```
Imports System.Net.Http
Imports System.Net.Http.Headers
Imports System.Text

Dim http As New HttpClient()
http.BaseAddress = New Uri("http://127.0.0.1:3000/")
http.DefaultRequestHeaders.Authorization =
    New AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale")

' Lettura senza svuotare la coda
Dim json As String = Await http.GetStringAsync("messages?peek=1")
' ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

' Conferma dei messaggi elaborati
Dim ack As New StringContent("{""upToSeq"":" & ultimoSeq & "}", Encoding.UTF8, "application/json")
Dim r As HttpResponseMessage = Await http.PostAsync("messages/ack", ack)

' Invio con clientMsgId: nuovo per ogni messaggio, riutilizzato solo nei ritentativi
Dim idMsg As String = Guid.NewGuid().ToString()
Dim body As String = "{""number"":""393401234567"",""message"":""Testo"",""clientMsgId"":""" & idMsg & """}"
Dim s As HttpResponseMessage = Await http.PostAsync("send", New StringContent(body, Encoding.UTF8,
"application/json"))
Select Case CInt(s.StatusCode)
    Case 429 ' limite raggiunto: attendere e ritentare con lo stesso idMsg
        Dim attesa As TimeSpan = If(s.Headers.RetryAfter?.Delta, TimeSpan.FromSeconds(60))
    Case 422 ' numero non registrato su WhatsApp: non ritentare
End Select
```

12. Jurnalul conexiunilor și jurnalul pe ecran

Connection.log se află în PathWork\Instances_Log\<număr conectat>\Connection.log. Conține jurnalul de conexiune și o copie a ieșirii text.

Conținut	Vizibilitate
Banner	Întotdeauna
Rezumatul de pornire	Întotdeauna
Avertisment SECURITY	Întotdeauna, când serverul este accesibil din rețea fără ApiToken
Cod QR	Întotdeauna când este realmente necesar
Warning / Error	Întotdeauna
Erori netratate	Întotdeauna, înainte de închiderea programului
INFO / DEBUG	Cu ConsoleLog=Y
Jurnale normale	Data/ora compactă la început

În timpul unei pierderi prelungite a conexiunii la Internet, rândurile identice consecutive "Connection closed. Reconnecting: true" sunt înregistrate o singură dată, atât pe ecran, cât și în jurnal.

13. Diagnosticare și rezolvarea problemelor

Simptom	Acțiune
Fișierul de configurare nu a fost găsit	Folosiți --create_config= sau --config=.
Conexiunea la baza de date a eșuat	Verificați TypeDB, ConnStr și PwdConnStr; datele rămân în coadă.
Parola ENC nu poate fi decriptată	Regenerați-o cu --pwd pe PC-ul care rulează bridge-ul.
SQLite ocupat	Așteptați reîncercarea automată; verificați ce program ține fișierul deschis.
Prima pornire lentă cu o bază de date mare	Crearea indexului pe MsgID este în curs: așteptați; mesajele rămân în coadă.
Clientul nu ajunge la bridge	Dacă clientul este pe un alt PC: BindAddress gol sau 0.0.0.0, regulă de paravan de protecție pentru port, ApiToken. Dacă este pe același PC: folosiți http://127.0.0.1:<Port>.

Simptom	Acțiune
Răspuns HTTP 401	Token lipsă sau diferit de ApiToken: verificați header-ul Authorization: Bearer sau X-Api-Key.
Răspuns HTTP 400 la /send	Verificați number (număr cu prefix sau JID) și message (să nu fie gol).
Răspuns HTTP 503 la /send	WhatsApp nu este conectat: verificați /status (connected, reconnectInSec).
Răspuns HTTP 429 la /send	Limita de trimitere a fost atinsă: așteptați retryAfterSec secunde înainte de a reîncerca. Dacă se întâmplă des, reduceți ritmul trimiterilor sau ajustați SendMinIntervalSec, SendMaxPerHour și SendMaxWaitSec (vedeți 5.5).
Răspuns HTTP 422 la /send	Numărul nu folosește WhatsApp: verificați numărul și prefixul internațional. Verificarea se dezactivează cu SendCheckNumber=N.
Răspuns HTTP 409 la /send	Același clientMsgId a fost folosit pentru un mesaj diferit: clientul trebuie să genereze un clientMsgId nou pentru fiecare mesaj și să îl reutilizeze doar la reîncercări.
Avertisment SECURITY la pornire	Setați ApiToken sau BindAddress=127.0.0.1.
Mesaj salvat fără fișier media	Vedeți jurnalul ERROR (descărcare nereușită sau MaxMediaMB depășit) și câmpul mediaError.
Fișiere în pending_db\failed	Citiți cauza în jurnal; după corectare, mutați fișierul în pending_db și reporniți.
Avertismentul "Connected number still not available"	Numărul conectat nu este încă disponibil: mesajele primite rămân în coada de pornire și sunt înregistrate imediat ce numărul este cunoscut (vedeți 10.6). Dacă avertismentul persistă, verificați conexiunea cu /status.
Fișiere în early_queue\failed	Citiți cauza în jurnal; după corectare, mutați fișierul în early_queue și reporniți.
Manuale nedorite în folderul programului	Porniți cu --Skip-Manuals sau setați Skip-Manuals=Y; fișierele deja prezente pot fi șterse.
Eroarea "session replaced"	O altă instanță folosește același folder auth_info: lăsați activă doar una.
Cod QR nou cerut după schimbarea PC-ului sau a hardware-ului	Comportament prevăzut: sesiunea este legată de PC (vedeți 7.1). Scanați noul cod QR; după verificare, ștergeți folderul auth_info.undecryptable_<data> și eliminați vechiul dispozitiv din Dispozitive asociate pe telefon.
Folderul auth_info copiat de pe un alt PC nu funcționează	Comportament prevăzut: fișierele criptate pot fi folosite doar pe PC-ul care le-a creat. Conectați instanța cu un nou cod QR.
Avertismentul "EncryptAuth is ignored" la pornire	Sesiunea este întotdeauna criptată: eliminați rândul EncryptAuth din fișierul de configurare.
Eroare la pornire privind DbSendSecret	Secretul lipsește, este mai scurt de 32 de caractere sau nu poate fi decriptat pe acest PC: trimiterea din baza de date rămâne dezactivată. Setați DbSendSecret sau regenerați-l cu --pwd pe PC-ul bridge-ului.
Rânduri din tabelul trimiterilor în starea K	Programul care scrie folosește un secret diferit de DbSendSecret sau datele au fost modificate după semnare. Cu MySQL, verificați codificarea utf8mb4.
Rânduri din tabelul trimiterilor care rămân în starea N	Verificați DbSend=Y, conexiunea (connected în /status), că PhoneSender din rânduri este numărul conectat și limitele de trimitere.
Rânduri din tabelul trimiterilor în starea X	Rânduri scrise cu mult timp înainte de citire sau ceasuri ale PC-urilor nesincronizate: măriți DbSendMaxAgeMin (maximum 1440) sau sincronizați ceasurile.
Rânduri din tabelul trimiterilor în starea U	Bridge-ul s-a oprit de două ori în timpul trimiterii aceluiași rând: verificați în hh_Albatros_Webhooks dacă mesajul a plecat și, dacă este necesar, reinserați-l cu un nou ClientRef.
Programul se închide cu o eroare	Citiți Connection.log; rulați bridge-ul ca serviciu cu repornire automată.
Consolă fără INFO/DEBUG	Setați ConsoleLog=Y.
Codul QR nu apare	Verificați dacă sesiunea este încă validă.
Editare nedecriptată	Verificați Save_Secrets=S și messageSecret.

Simptom	Acțiune
Jurnalul în tabel lipsește	Setați DebugLog=All sau o listă de tipuri.
Fișiere media nedorite în DebugData	Configurați DebugLogFilter.
Număr WhatsApp blocat (ban)	Blocarea este decisă și aplicată de Meta, de regulă pentru încălcarea regulilor WhatsApp (spam, publicitate, trimiteri în masă): trebuie gestionată direct cu WhatsApp, din aplicația de pe telefon. Vedeți avertismentele inițiale.
Programul nu se mai conectează după o actualizare WhatsApp	Posibilă modificare a protocolului de către WhatsApp (vedeți avertismentele inițiale): verificați dacă este disponibilă o versiune actualizată a programului.
Parola ENC sau sesiunea nu pot fi decriptate, fără schimbări de hardware	Verificați ca PowerShell să fie disponibil pentru contul programului și ca serviciul WMI să fie activ.

14. Bune practici

• Folosiți InternalSqlite pentru instalări de test, când nu este nevoie de o bază de date externă sau când lucrați într-un mediu protejat.

• În producție, cu o bază de date externă, preferați PwdConnStr=ENC:... (cu InternalSqlite parola nu este necesară).

• Setați întotdeauna ApiToken când bridge-ul este accesibil de pe alte PC-uri; folosiți BindAddress=127.0.0.1 când clientul este pe același PC.

• În clienți, preferați citirea în două faze (?peek=1 și /ack) citirii clasice.

• Păstrați Save_Secrets=S dacă aveți nevoie de editări.

• Monitorizați /status și /db-queue și verificați periodic pending_db\failed și early_queue\failed.

• Rulați bridge-ul ca serviciu Windows cu repornire automată în caz de eroare.

• Nu folosiți programul pentru spam, publicitate comercială sau trimiteri în masă: sunt interzise de regulile WhatsApp și pot cauza blocarea numărului.

• Păstrați active limitele de trimitere (SendMinIntervalSec, SendMaxPerHour) și trimiteți un clientMsgId unic pentru fiecare mesaj, reutilizându-l doar când clientul repetă aceeași cerere.

• Pentru trimiterea din baza de date folosiți DbSendSecret criptat (ENC:...), nu scrieți niciodată secretul în baza de date și ștergeți periodic rândurile rămase în stările E, K, X și U.

• Folderul auth_info este criptat și legat de PC: copierea lui pe un alt PC nu folosește la nimic (este oricum nevoie de un nou cod QR). Limitați totuși accesul la el la conturile care rulează bridge-ul; restaurarea dintr-o copie de rezervă funcționează doar pe același PC.

• Folosiți valori CodeInstance distincte pentru mai multe numere WhatsApp; nu folosiți același auth_info pentru două instanțe.

• Activați DebugLog și ConsoleLog în timpul diagnosticării.

• Setați DebugLogFilter pentru a limita fișierele media de diagnosticare și MaxMediaMB în funcție de memoria disponibilă.

• Adaptați PendingMaxSec și DbMaxRetries la contextul operațional.

15. Trimitere din baza de date (tabelul trimiterilor)

Pe lângă endpoint-urile HTTP, bridge-ul poate trimite mesajele pe care un alt program le scrie în tabelul hh_Albatros_SendQueue. Cu DbSend=Y, bridge-ul citește tabelul la fiecare DbSendPollSec secunde și trimite doar

rândurile numărului conectat, ca și cum ar fi sosit prin /send sau /send-media: aceeași coadă, aceleași limite (SendMinIntervalSec, SendMaxPerHour, SendMaxWaitSec) și aceeași verificare a numărului (SendCheckNumber).

Fiecare rând este semnat cu un secret cunoscut doar de bridge și de programul care scrie: cine accesează baza de date fără secret nu poate crea rânduri valide și nici nu le poate modifica conținutul.

15.1 Activare

• DbSend=Y activează funcția. Tabelul este creat automat la pornire; o eventuală eroare de creare este semnalată, dar nu blochează salvarea mesajelor.

• DbSendSecret este secretul semnăturii (minimum 32 de caractere), în clar sau ENC:... generat cu --pwd. Fără un secret valid, funcția rămâne dezactivată, iar pornirea semnalează acest lucru cu o eroare. Același secret trebuie să fie cunoscut de programul care scrie rândurile și nu trebuie scris niciodată în baza de date.

• DbSendPollSec (de la 1 la 300, implicit 5): secunde între două citiri. Tabelul este citit doar când bridge-ul este conectat și numărul conectat este cunoscut.

• DbSendMaxAgeMin (implicit 60): minute de valabilitate ale unui rând; valorile sub 30 devin 30, cele peste 1440 devin 1440.

• POST /send-queue/poll anticipează citirea (răspuns 202). Mai multe cereri apropiate în timp produc o singură citire.

```
DbSend=Y
DbSendSecret=ENC:...
DbSendPollSec=5
DbSendMaxAgeMin=60
```

15.2 Structura tabelului

Coloanele indicate ca semnate intră în semnătură (vedeți 15.4). Cel care scrie inserează rândul cu SendStatus = N; coloanele bridge-ului sunt gestionate de bridge.

Coloană	Scrisă de	Descriere
ID	baza de date	Identificator progresiv; rândurile sunt procesate în ordinea ID.
DateInsert	baza de date	Data inserării (doar informativă).
PhoneSender	cel care scrie	Semnată. Numărul care trebuie să trimită, doar cifre: bridge-ul citește doar rândurile propriului număr.
ClientRef	cel care scrie	Semnată. Referință unică per PhoneSender (index unic), maximum 100 de caractere.
Recipient	cel care scrie	Semnată. Destinatar: număr cu prefix internațional sau JID (grupuri @g.us).
MsgKind	cel care scrie	Semnată. Tipul mesajului (vedeți 15.3).
Title	cel care scrie	Semnată. Numele locului, numele contactului sau întrebarea sondajului.
MsgText	cel care scrie	Semnată. Text, descriere, adresă, vCard sau opțiunile sondajului.
FileName	cel care scrie	Semnată. Numele documentului.
MimeType	cel care scrie	Semnată. Tipul MIME al fișierului media; opțional.
Latitude, Longitude	cel care scrie	Semnate. Coordonate ca text, cu punct zecimal.
MediaData	cel care scrie	Semnată prin SHA-256 al său. Fișier atașat în format binar (nu base64).
CreatedAtUtc	cel care scrie	Semnată. Data creării în milisecunde UTC de la 01/01/1970.
SignVer	cel care scrie	Versiunea formatului semnăturii: 1.
Signature	cel care scrie	Semnătură HMAC-SHA256 în hexazecimal (64 de caractere).
SendStatus	cel care scrie / bridge	Starea rândului (vedeți 15.5); cel care scrie inserează N.
ClaimToken, ClaimedBy, ClaimedAtUtc	bridge	Preluarea rândului.
Attempts	bridge	Încercări suplimentare deja folosite (maximum 1).
SentMsgID	bridge	Identificatorul mesajului, ales înainte de trimitere.

Coloană	Scrisă de	Descriere
ErrorCode, ErrorText, ProcessedAtUtc	bridge	Rezultatul unui rând netrimis.

15.3 Tipuri de mesaje (MsgKind)

MsgKind	Coloane folosite	Note
text	MsgText	Mesaj doar text.
image	MediaData; MsgText (descriere), MimeType	Imagine.
video	MediaData; MsgText (descriere), MimeType	MIME implicit video/mp4.
gif	MediaData (mp4); MsgText (descriere)	Video redat ca GIF animat.
audio	MediaData; MimeType	MIME implicit audio/mpeg.
voice	MediaData; MimeType	Mesaj vocal; MIME implicit audio/ogg; codecs=opus.
document	MediaData, FileName; MsgText (descriere), MimeType	MIME dedus din extensie dacă MimeType este gol.
sticker	MediaData (webp)	Autocolant.
location	Latitude, Longitude; Title (nume), MsgText (adresă)	Latitudine de la -90 la 90, longitudine de la -180 la 180.
contact	MsgText (unul sau mai multe vCard); Title (numele afișat)	Fiecare vCard de la BEGIN:VCARD la END:VCARD.
poll	Title (întrebare), MsgText (opțiuni, câte una pe rând)	De la 2 la 12 opțiuni; o singură alegere.
pollmulti	Title (întrebare), MsgText (opțiuni, câte una pe rând)	Ca poll, cu alegeri multiple.

Fișierul media nu poate depăși MaxMediaMB. Reacțiile, modificările, ștergerile și redirecționările nu sunt gestionate, deoarece necesită un mesaj deja existent.

15.4 Semnătura

Signature este HMAC-SHA256, cu cheia egală cu secretul DbSendSecret în UTF-8, al acestui text UTF-8:

```
ALB-SEND-1
<lunghezza>:<PhoneSender>
<lunghezza>:<ClientRef>
<lunghezza>:<Recipient>
<lunghezza>:<MsgKind>
<lunghezza>:<Title>
<lunghezza>:<MsgText>
<lunghezza>:<FileName>
<lunghezza>:<MimeType>
<lunghezza>:<Latitude>
<lunghezza>:<Longitude>
<lunghezza>:<MediaSha256>
<lunghezza>:<CreatedAtUtc>
```

• Fiecare rând se termină doar cu caracterul LF (sfârșit de linie Unix); <lungime> este numărul de octeți UTF-8 ai valorii.

• O valoare NULL este tratată ca șir gol. MediaSha256 este SHA-256 hexazecimal cu minuscule al MediaData, gol dacă MediaData este NULL sau gol. CreatedAtUtc este scris în cifre.

• Signature este în hexazecimal (majuscule sau minuscule). Comparația se face în timp constant.

• Valorile semnate trebuie să fie identice cu cele scrise în tabel: cu MySQL, conexiunea și tabelul trebuie să folosească utf8mb4, altfel emoji-urile și caracterele speciale se modifică și rândul apare cu semnătură nevalidă.

15.5 Procesare și rezultate

Bridge-ul preia fiecare rând în mod atomic (ClaimToken), fără blocări pe tabel, astfel încât două instanțe conectate la același număr nu îl pot trimite de două ori. Apoi verifică numărul, semnătura și valabilitatea în timp, trimite și înregistrează rezultatul. Rândul este preluat doar dacă coada de trimitere îl poate expedia în SendMaxWaitSec.

SendStatus	Semnificație
N	De trimis (scris de cel care pune mesajul în coadă).
C	În procesare.
(rând șters)	Trimis. Mesajul rămâne în hh_Albatros_Webhooks cu Flag_Base = N_TAB; fișierul media este salvat fără a fi redescărcat.
E	Eroare definitivă: ErrorCode și ErrorText indică motivul.
K	Semnătură nevalidă: rândul nu este trimis, iar jurnalul raportează o eroare SECURITY.
X	Expirat (mai vechi decât DbSendMaxAgeMin).
U	Rezultat incert: rândul nu mai este trimis.

• Bridge neconectat sau coadă de trimitere plină: rândul revine în starea N fără a consuma încercări.

• Eroare în timpul trimiterii: o singură încercare suplimentară, cu același identificator al mesajului; apoi starea E.

• Preluare fără rezultat (de exemplu, oprirea bridge-ului): după 15 minute plus SendMaxWaitSec, dacă mesajul apare deja în hh_Albatros_Webhooks (necesită Save_Sent=S), rândul este șters; altfel se face o singură încercare suplimentară; a doua oară, rândul trece în starea U.

• Dacă după trimitere baza de date nu este accesibilă, ștergerea rândului este repetată la citirea următoare: rândul nu este retrimis.

• Rândurile în stările E, K, X și U rămân în tabel: trebuie citite și șterse de programul care le-a scris.

ErrorCode	Stare	Semnificație
INVALID_SIGNATURE	K	Semnătura nu corespunde: secret diferit sau date modificate după semnare.
UNSUPPORTED_SIGN_VER	K	Versiune a semnăturii negestionată.
EXPIRED	X	Rând mai vechi de DbSendMaxAgeMin minute.
CREATED_IN_FUTURE	E	CreatedAtUtc cu peste 5 minute în viitor (ceasuri nesincronizate).
INVALID_CREATED_AT	E	CreatedAtUtc lipsă sau nevalid.
INVALID_RECIPIENT	E	Destinatar lipsă sau nevalid.
INVALID_KIND	E	MsgKind negestionat.
INVALID_MESSAGE	E	Text lipsă pentru tipul text.
MISSING_MEDIA	E	MediaData lipsă pentru un tip care îl necesită.
MISSING_FILE_NAME	E	FileName lipsă pentru tipul document.
MEDIA_TOO_LARGE	E	MediaData depășește MaxMediaMB.
MEDIA_READ_ERROR	E	MediaData nu poate fi citit.
INVALID_LOCATION	E	Coordonate lipsă sau în afara intervalului.
INVALID_CONTACT	E	Niciun vCard în MsgText.
INVALID_POLL	E	Întrebare lipsă sau opțiuni în afara intervalului 2-12.
NOT_ON_WHATSAPP	E	Destinatarul nu folosește WhatsApp.
SEND_ERROR	E	Trimitere nereușită și la încercarea suplimentară.
CLAIM_TIMEOUT	N	Preluare fără rezultat: repus în așteptare pentru încercarea suplimentară.
UNCERTAIN	U	A doua preluare fără rezultat: rândul nu este retrimis.

15.6 Monitorizare

/status raportează în blocul sendQueue: activarea, tabelul, intervalul de citire, valabilitatea rândurilor, citirea în curs, ultima eroare și contoarele rândurilor trimise, cu semnătură nevalidă, expirate, cu eroare, cu rezultat incert, reîncercate și repuse în așteptare.

/config raportează dbSend, dbSendPollSec, dbSendMaxAgeMin și dacă secretul este setat (dbSendSecretSet) și criptat (dbSendSecretEncrypted), niciodată valoarea sa. Rezumatul de la pornire afișează rândurile DbSend și DbSendSecret (vedeți 8).

15.7 Programul Albatros SendQueue

Albatros SendQueue pune mesaje în coadă în tabelul trimiterilor. Este alcătuit dintr-un program cu interfață grafică (Albatros_SendQueue.exe) și din biblioteca AlbatrosSendQueueLib.dll, care conține toată logica: citirea Instance_Config.ini, semnarea, scrierea în tabel, starea rândului și cererile HTTP către bridge.

Biblioteca poate fi folosită și de alte programe .NET (de exemplu, un program de gestiune): parametrii comuni (baza de date, DbSendSecret, numărul expeditorului, adresa bridge-ului) se setează o singură dată, citindu-i din Instance_Config.ini sau transmițându-i direct, fără fișier de configurare, iar pentru fiecare tip de mesaj există o funcție de trimitere dedicată (SendText, SendImage, SendDocument, SendLocation, SendPoll...).

Ca număr al expeditorului se folosește parametrul PhoneSender, pe care bridge-ul îl ignoră. Programul și biblioteca au propriul manual de utilizare.

Anexa A. Exemple de configurare

A.1 Test pe același PC cu SQLite încorporat

```
Key=TEST
TypeDB=InternalSqlite
PathWork=c:\Albatros\Phone\
CodeInstance=Test
Port=3000
BindAddress=127.0.0.1
DebugLog=All
ConsoleLog=Y
```

A.2 Producție cu SQL Server și client pe un alt PC

```
Key=Albatros
TypeDB=SqlServer
ConnStr=Driver={ODBC Driver 17 for SQL Server};Server=SRVSQL;Database=AlbatrosBridge;UID=albatros_bridge;
PwdConnStr=ENC:....
PathWork=c:\Albatros\Phone\
CodeInstance=393401234567
Port=3000
; BindAddress non indicato: ascolto su tutte le interfacce di rete
ApiToken=IlMioTokenLungoECasuale
MaxMediaMB=100
DebugLog=Error, Warning
DebugLogMaxRecords=10000
PendingMaxSec=172800
```

Valorile pentru Server, Database, UID și token sunt doar exemple.

În șirul de conexiune (connection string) nu se indică parola, care se introduce în PwdConnStr (de preferință criptată cu --pwd).

▣ ユーザー マニュアル – Albatros Bridge

バージョン 1.4.129.6121 – ビルド 2026 年 10 月 4 日

⚠ 重要な注意事項

注意事項	詳細
Meta との関係はありません	作者は、Meta Platforms, Inc.、WhatsApp LLC、または Meta グループの他の企業とは一切関係がなく、提携、承認、後援、推奨を受けているものではありません。 WhatsApp は各権利者の商標であり、本プログラムの機能を説明する目的でのみ記載しています。
非公式の手順	WhatsApp への接続は、WhatsApp Business の公式 API ではなく、非公式の手順 (WhatsApp Web と同様のリンク済みデバイスとして、オープンソース ライブラリを使用) によって行われます。 Meta または WhatsApp 製品を運営する企業は、プロトコル、接続方式、暗号化、利用規約をいつでも予告なく変更する可能性があります。その場合、本プログラムは対応が行われるまで、全体または一部が動作しなくなることがあります。
すべての非公式製品に共通する状況	世界には、非公式の WhatsApp ソフトウェアやサービスを提供する企業が数多く存在します。前項で説明した問題はそのすべてに等しく当てはまります。Meta が決定する変更によって動作が中断されないことを保証できる企業はなく、その時期を予測することもできません。
番号に対するリスク	非公式クライアントの使用は WhatsApp の利用規約に適合しない場合があり、リンクした番号が制限またはブロックされる可能性があります。その判断および関連するリスクは、本プログラムの使用者が負うものとします。
禁止されている用途: スパム、広告、一斉送信	本プログラムを使用してスパム、商業広告、または一斉メッセージを送信することは、WhatsApp の規約で明確に禁止されています。したがって、本プログラムの不適切な使用により、リンクした WhatsApp 番号がブロック (BAN) される可能性があります。この場合のブロックは作者には一切起因せず、不適切な使用を理由として Meta が直接適用するものです。
「現状のまま」の使用	本プログラムは「現状のまま」(「as is」) で提供され、動作、特定目的への適合性、サービスの継続性、エラーがないことを含め、明示または黙示を問わず、いかなる種類の保証もありません。
免責事項	適用される法律で認められる最大限の範囲において、作者は、本プログラムの使用または使用不能に起因する直接的または間接的な損害 (データやメッセージの損失、逸失利益、業務の中断、番号のブロックなど) について、プログラムの入手方法 (有償、無償提供、フリー ソフトウェア、その他の方法) を問わず、一切の責任を負いません。

注意事項	詳細
個人データ	保存されたメッセージには個人データが含まれます。本プログラムをインストールして使用する者は、その取り扱い、データベースのセキュリティ、およびデータ保護規則 (GDPR) の遵守について責任を負います。

****
** Albatros Bridge **
** rel. 1.4.129.6121 - 04/10/2026 **
** © Raffaele Romano 2026 **
****

目次

1. 本プログラムについて

2. 要件

3. コマンド ライン パラメーター

4. 主なファイルとフォルダー

5. 構成ファイル Instance_Config.ini

6. データベースの構成

7. データベース パスワードと WhatsApp セッションの暗号化

8. プログラムの起動

9. WhatsApp への初回接続

10. 全般的な動作

11. HTTP インターフェイス (エンドポイント)

12. 接続ログと画面ログ

13. 診断とトラブルシューティング

14. ベスト プラクティス

15. データベースからの送信 (送信テーブル)

付録 A. 構成例

1. ▪ 本プログラムについて

Albatros Bridge は、WhatsApp とデータベースを結ぶブリッジです。WhatsApp Web と同様に QR コードで WhatsApp に接続し、データの受信、分類、永続化、および HTTP による公開を行います。

• 受信: テキスト、画像、動画、音声、ドキュメント、ステッカー、位置情報、ステータス、リアクション、技術メッセージ。

• 編集されたメッセージ: 編集を処理します。必要な messageSecret が利用可能な場合は、LID で暗号化された新しいメッセージも含みます。

• 送信: HTTP エンドポイント経由でテキストおよびメディアのメッセージを送信し、受信したデータをチェックします。

• データベースからの送信: 別のプログラムが署名付きの行を hh_Albatros_SendQueue テーブルに書き込み、ブリッジがそれを送信します (15 を参照)。

• データベース: SQL Server、MySQL、MariaDB、ODBC 経由の SQLite、または組み込みの SQLite。

• 複数インスタンス: 各インスタンスは専用のフォルダーを使用し、それぞれ独自の WhatsApp セッションを持つことができます。

• 永続キュー: データベースが利用できない場合、データはディスク上に保持されます。ディスクへの書き込みは突然の中断から保護されており、保存を繰り返しても重複は作成されません。

• 起動時の時系列順: リンクした番号が利用可能になる前に受信したメッセージは、ディスク上のキューに入れられ、番号が判明した時点で古いものから新しいものへの順に記録されます。

• 信頼性の高いメディア処理: ダウンロードの自動再試行とサイズ制限があります。メディアが利用できない場合でも、メッセージは保存されます。

• HTTP セキュリティ: 待ち受けアドレスの設定 (BindAddress) とトークンによる認証 (ApiToken)。

• 保護された WhatsApp セッション: セッション ファイルは常に暗号化され、プログラムがインストールされた PC に結び付けられます。

• 段階的な待機時間による WhatsApp への自動再接続。

• ログ: テーブルへの診断ログ、接続ログ、およびテキスト出力のコピー。

• 挿入後の実行可能ファイル: オプション。保存が成功した後に実行されます。

2. ✓ 要件

要件	詳細
オペレーティング システム	Windows 64 ビット。
ODBC 経由のデータベース	SQL Server、MySQL、MariaDB、または外部 SQLite 用の 64 ビット ODBC ドライバーがインストールされていること。
組み込み SQLite	InternalSqlite: データベース用の ODBC ドライバーは不要です。
電話	WhatsApp が有効で、初回接続時に QR をスキャンできること。
ネットワーク	WhatsApp サーバーへのインターネット アクセス。HTTP クライアントが別の PC にある場合は、ポート Port に到達可能である必要があり (Windows ファイアウォール)、ApiToken の設定をお勧めします。
HTTP クライアント	JSON データを含む HTTP 要求を送信できる任意のプログラム (例: VB.NET クライアント)。
Windows PowerShell	Windows PowerShell 5.1 (Windows に付属) が存在し、プログラムを実行するアカウントで使用できること。また、WMI サービスが有効で、PowerShell スクリプトの実行をブロックするグループ ポリシーがないこと。データベース パスワードと WhatsApp セッションの暗号化に不可欠です。PowerShell がないと、PC に結び付けられたキーを正しく安定して算出できません。

3. コマンド ライン パラメーター

パラメーター	効果	ブリッジを起動するか
--h / --help	パラメーターのヘルプを表示して終了します。	いいえ
--version	バナーとバージョン番号を表示して終了します。	いいえ
--create_config <file>	すべてのパラメーターがコメント付きで記載されたサンプル構成ファイルを作成します。既存のファイルは上書きしません。	いいえ
--config <file|cartella>	指定したファイル、または指定したフォルダー内の Instance_Config.ini を使用します。	はい

パラメーター	効果	ブリッジを起動するか
--pwd	非表示入力でパスワードを 2 回要求し、PwdConnStr=ENC:... を生成します。同じ暗号化値が DbSendSecret=ENC:... としても提示されます (15 を参照)。	いいえ
--pwd=<password>	指定した値から暗号化された PwdConnStr を直接生成します。	いいえ
--man_IT [=<file|cartella>]	プログラムに含まれるイタリア語のユーザー マニュアル (.docx) を保存します。パスを指定しない場合は現在のフォルダーに、フォルダーを指定した場合はその中に、それ以外の場合は指定した名前で保存します。既定の名前: Manuale_Albatros_Bridge_<versione>.docx。既存のファイルは上書きしません。	いいえ
--man_EN [=<file|cartella>]	--man_IT と同様で、英語のユーザー マニュアル用です。既定の名前: Manual_Albatros_Bridge_<versione>_EN.docx。	いいえ
--Skip-Manuals	起動時にユーザー マニュアルをプログラム フォルダーへ自動保存しません (8 を参照)。構成ファイルの Skip-Manuals=Y と同じです。	はい
パラメーターなし	構成を自動検索する標準の起動。	はい

パラメーターは、スペース区切りの形式、または対応している場合は '=' を使った形式で指定できます。名前の大文字と小文字は区別されません。引用符で囲まれたパスは正規化されます。

```
Albatros-Bridge-odbc.exe --help
Albatros-Bridge-odbc.exe --version
Albatros-Bridge-odbc.exe --config "C:\Albatros\Instance01"
Albatros-Bridge-odbc.exe --config="C:\Albatros\Instance01\Instance_Config.ini"
Albatros-Bridge-odbc.exe --create_config="C:\tmp\MyConfig.ini"
Albatros-Bridge-odbc.exe --pwd
Albatros-Bridge-odbc.exe --pwd=MiaPassword123
Albatros-Bridge-odbc.exe --man_IT
Albatros-Bridge-odbc.exe --man_EN="C:\Doc\"
Albatros-Bridge-odbc.exe --Skip-Manuals
```

環境変数: --config が指定されていない場合、PHONES_BRIDGE_CONFIG で構成ファイルまたはフォルダーを指定できます。

4. 主なファイルとフォルダー

構造は PathWork の下に構成されます。各インスタンスは、プレフィックス Instance_ と CodeInstance の値によって分離されます。

要素	説明
Instance_Config.ini	インスタンスの構成。
Manuale_Albatros_Bridge_<versione>.docx, Manual_Albatros_Bridge_<versione>_EN.docx	イタリア語と英語のユーザー マニュアル。存在しない場合、起動時にプログラム フォルダーへ自動的に保存されます (8 を参照)。
PathWork	作業のルート フォルダー。既定値: c:\Albatros\Phone\。
Instance_<CodeInstance>\	インスタンスのフォルダー。
...\QrCode\latest.png	現在の QR コード。QR が実際に表示されている場合のみ存在します。
...\auth_info\	WhatsApp セッションの資格情報。常に暗号化され、PC に結び付けられます (7.1 を参照)。2 つのインスタンスで同時に使用してはなりません。
...\auth_info.undecryptable_<data>\	この PC で復号できない以前のセッション (7.1 を参照)。削除されずに別の場所へ移動されます。
...\pending_db\	まだ保存されていないデータの永続キュー (メッセージ用の *_M.json ファイル、messageSecret 用の *_S.json ファイル)。
...\pending_db*.json.tmp	キューの書き込み中に使用される一時ファイル。突然の中断後に残っている場合は、起動時に削除されます。
...\pending_db\failed\	再試行回数/時間の上限を超えた要素、または保存できない要素 (読み取れないファイルや無効なデータ)。
...\early_queue\	起動キュー: リンクした番号が利用可能になる前に受信し、記録を待っているメッセージとイベント (10.6 を参照)。
...\early_queue\failed\	読み取れない、または処理できない起動キューの要素 (10.6 を参照)。
...\Albatros_Embedded.db	ConnStr なしで使用する場合の組み込み SQLite データベース。
PathWork\Instances_Log\<numero>\Connection.log	接続ログとテキスト出力のコピー。
%TEMP%\AlbatrosBridgeOdbc\	実行可能ファイルに組み込まれた ODBC 用の一時領域。

5. 構成ファイル Instance_Config.ini

このファイルでは、1 行に 1 組の Chiave=Valore または Chiave:Valore (キー=値) を記述します。# (シャープ) または ; (セミコロン) で始まる行はコメントです。キーの大文字と小文字は区別されません。

パラメーター	既定値 / 例	説明
Key	Albatros	インスタンスのラベル (最大 50 文字)。
TypeDB	InternalSqlite	データベースの種類: MySql、SqlServer、MariaDB、SqLite、InternalSqlite。
ConnStr	空	接続文字列。InternalSqlite の場合は省略できます。
PwdConnStr	空	DB パスワード。平文または ENC:...。組み込み SQLite では無視されます。
PathWork	c:\Albatros\Phone\	ルート作業フォルダー。
CodeInstance	Default	インスタンスのコード (CodeIstance も可)。
PathAuth	自動	WhatsApp 資格情報の代替フォルダー。
PathBin	c:\Albatros\Phone\bin\	挿入後の実行可能ファイルのフォルダー (bin も可)。
ExeName	空	保存が成功した後に、パラメーター --rec、--istance、--Cell を付けて起動される実行可能ファイル (10.1 を参照)。Exe も可。
Save_Status	N	配信/既読の更新を保存します。
Save_StatusBroadcast	N	status@broadcast のステータス/ストーリーを保存します。N の場合、その messageSecret、編集、削除も保存されません。
Save_Sent	S	リンクした番号から送信したメッセージを保存します。N の場合、その messageSecret、およびリンクした番号による編集と削除も保存されません。
Save_protocolMessage	N	プロトコルの技術メッセージを保存します。
Save_Secrets	S	暗号化された編集に必要な messageSecret を保存します。Save_Sent、Save_StatusBroadcast、Save_protocolMessage で除外されたメッセージの messageSecret は保存されません。
ConsoleLog	N	情報/デバッグ メッセージの画面表示を有効にします。バナー、概要、QR、警告、エラーは常に表示されます。
DebugLog	空	INFO/WARNING/ERROR/DEBUG について、hh_Albatros_Webhooks_log への診断ログを有効にします。
DebugLogFilter	空	DebugData に含まれるマルチメディア コンテンツのみをフィルターします。
DebugLogMaxRecords	10000	hh_Albatros_Webhooks_log テーブルに保持するレコードの最大数。空または 0 = 制限なし。

パラメーター	既定値 / 例	説明
DbRetrySec	30	DB に対する線形バックオフの基準値。
DbRetryMaxSec	300	再試行間の待機時間の上限 (最大 86400)。
DbMaxRetries	0	要素ごとの最大試行回数。0 = 無制限。
PendingMaxSec	172800	pending に保持する最大時間。0 = 時間制限なし。
Port	3000	HTTP ポート。
BindAddress	空	HTTP サーバーが待ち受けるアドレス。空 = すべてのネットワーク インターフェイス (以前のバージョンと同じ)、127.0.0.1 = この PC のみ。5.3 を参照してください。
ApiToken	空	すべての HTTP エンドポイントで必要なトークン。空 = 認証なし。5.3 を参照してください。
MaxMediaMB	100	ダウンロードするメディアの最大サイズ (MB)。0 = 制限なし。5.4 を参照してください。
MaxMemQueue	500	メモリ内の HTTP キューの最大要素数。0 にするとキューが無効になります。
SendMinIntervalSec	3	2 回の送信間の最小秒数 (小数も可)。間隔は最大 30% までランダムに延長されます。0 = 待機なし。5.5 を参照してください。
SendMaxPerHour	60	直近 1 時間 (移動ウィンドウ) の最大送信数。0 = 上限なし。5.5 を参照してください。
SendMaxWaitSec	60	送信キュー内での要求の最大待機時間。超えると 429 を応答します (最大 3600)。0 = 待機なし。5.5 を参照してください。
SendCheckNumber	Y	Y = 送信前に、番号が WhatsApp を使用しているかを確認します。使用していない場合は 422 を応答します。5.5 を参照してください。
SendIdempotencyHours	24	clientMsgId の有効時間 (時間単位、最大 720)。0 = clientMsgId を無視します。5.5 を参照してください。
Skip-Manuals	N	Y = 起動時にユーザー マニュアルをプログラム フォルダーへ自動保存しません。パラメーター --Skip-Manuals と同じです。8 を参照してください。
DbSend	N	Y = hh_Albatros_SendQueue テーブルを読み取り、リンクした番号の行を HTTP 送信と同じ制限で送信します。15 を参照してください。
DbSendSecret	空	行の署名に使用するシークレット (最小 32 文字)。平文、または --pwd で生成した ENC:...。有効なシークレットがない場合、DbSend は無効のままです。15 を参照してください。

パラメーター	既定値 / 例	説明
DbSendPollSec	5	テーブルを読み取る間隔の秒数 (1 〜 300)。POST /send-queue/poll で読み取りを前倒しできます。15 を参照してください。
DbSendMaxAgeMin	60	CreatedAtUtc からの行の有効時間 (分)。30 未満の値は 30 に、1440 (24 時間) を超える値は 1440 になります。15 を参照してください。

使用できるブール値: S、Y、SI、SÌ、YES、TRUE、1 (小文字も可)。N や空を含むその他の値は、いいえ (No) と同じです。

5.1 DebugLog と DebugLogFilter

DebugLog は、hh_Albatros_Webhooks_log テーブルへの書き込みに関するものです。DebugLogFilter は、書き込む行を決定するものではありません。DebugData フィールドに含まれるマルチメディア コンテンツにのみ作用します。

設定	結果
DebugLog=All	INFO、WARNING、ERROR、DEBUG。All は他の値より優先されます。
DebugLog=Debug, Error	DEBUG と ERROR のみ。
DebugLog が空 / N / NO / FALSE / 0	テーブルへの診断ログは無効です。
DebugLogFilter=No-Video	DebugData から動画を除去します。
No-Audio	DebugData から音声を除去します。
No-PDF	DebugData から PDF を除去します。
No-Other	DebugData からその他のメディアを除去します。
No-Media	DebugData からすべてのメディアを除外します。

```
DebugLog=All
DebugLogFilter=No-Video,No-Other
ConsoleLog=Y
```

メッセージの DEBUG ログは、データベースへの保存が何度再試行されても、1 回だけ書き込まれます。

5.2 DebugLogMaxRecords

hh_Albatros_Webhooks_log に保持される行数を制限します。ログの書き込みが成功するたびに、最も古い行が別のトランザクションで削除されます。クリーンアップが失敗した場合でも、書き込んだばかりの行は保存されたままとなり、クリーンアップは次回のログ書き込み時に再試行されます。既定値は 10000 です。空、0、または無効な値 = 制限なし。

5.3 HTTP インターフェイスのセキュリティ: BindAddress と ApiToken

HTTP エンドポイントを使用すると、リンクした番号からのメッセージ送信、受信メッセージの読み取り、接続用 QR のダウンロードができます。そのため、エンドポイントにアクセスできる相手を制限することが重要です。

状況	推奨構成
ブリッジと同じ PC 上のクライアント	BindAddress=127.0.0.1。クライアントでは http://127.0.0.1:<Port> を使用します (localhost は使用しないでください。Windows では先に IPv6 アドレス ::1 として解決される場合があります)。ApiToken は任意です。
別の PC または仮想マシン上のクライアント	BindAddress は空 (または 0.0.0.0 かネットワーク アダプターの IP アドレス)、ApiToken は必須とし、クライアントのアドレスにのみポートを開く Windows ファイアウォールの規則を設定します。

ApiToken が設定されている場合、**すべての**エンドポイント (/qr を含む) でトークンが必要です。トークンは次の 2 つの方法のいずれかで送信します。送信しない場合の応答は HTTP 401 です。トークンの値が /config で表示されることはありません。

```
Authorization: Bearer <token>
X-Api-Key: <token>
```

長くランダムなトークンを使用してください。PowerShell で生成するには:

```
$b = New-Object byte[] 32
[System.Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($b)
[Convert]::ToBase64String($b)
```

クライアントの PC (例のアドレス) にのみポート 3000 を開くファイアウォール規則の例:

```
New-NetFirewallRule -DisplayName "Albatros Bridge 3000" -Direction Inbound `
  -Protocol TCP -LocalPort 3000 -RemoteAddress 192.168.1.20 -Action Allow
```

サーバーがネットワークから到達可能 (BindAddress が空またはローカル以外) で ApiToken が設定されていない場合、起動時に概要の最終行として SECURITY 警告が表示されます。

5.4 MaxMediaMB

ダウンロードするメディアのサイズを制限します。チェックは、ダウンロード前に WhatsApp が宣言したサイズに対して行われ、さらに実際のサイズに対しても行われます。上限を超えるメディアはダウンロードされません。メッセージは添付ファイルなしで保存され、ログにエラーが記録されます。既定値は 100 MB、0 = 制限なしです。値が大きすぎると、データベースに保存されるまでメディアが pending に保持されるため、メモリ使用量とディスク上のキュー領域が増加します。

5.5 送信の制限とチェック

これらの制限は、送信が速すぎたり多すぎたりすることで番号がブロックされるリスクを軽減しますが、スパム、広告、一斉送信への本プログラムの使用を正当化するものではありません (冒頭の注意事項を参照)。

送信キュー。/send と /send-media への要求はキューに入り、到着順に、少なくとも SendMinIntervalSec 秒の間隔を空けて送信されます。一定のリズムを避けるため、各間隔は最大 30% までランダムに延長されます。1 時間に送信されるメッセージは SendMaxPerHour 件以下です。HTTP 要求は自分の順番まで待機し、送信完了後に応答を受け取ります。順番が来るまでに SendMaxWaitSec 秒を超える場合、ブリッジは直ちに 429 を応答し、何秒後に再試行すべきかを示す retryAfterSec フィールドと Retry-After ヘッダーを返します。SendMaxWaitSec は HTTP クライアントのタイムアウトより短くしておく必要があります (.NET では HttpClient の既定値は 100 秒)。SendMinIntervalSec と SendMaxPerHour が両方とも 0 の場合、起動時に SECURITY 警告が表示されます。

番号の確認。SendCheckNumber=Y の場合、ブリッジはメッセージをキューに入れる前に、宛先が WhatsApp を使用しているかを確認します。使用していない場合は 422 (code NOT_ON_WHATSAPP) を応答し、そのメッセージは 1 時間あたりの送信枠を消費しません。結果は、番号が存在する場合は 24 時間、存在しない場合は 10 分間記憶されます。グループ (@g.us) と @lid 識別子は確認されません。技術的な理由 (タイムアウトなど) で確認できなかった場合でも、メッセージは送信され、ログに警告が記録されます。

重複のない送信 (clientMsgId)。クライアントは JSON でメッセージの一意の識別子 clientMsgId (1 〜 100 文字) を指定できます。これは、タイムアウト後など、同じ要求を繰り返す場合にのみ再利用します。その clientMsgId のメッセージが直近 SendIdempotencyHours 時間以内に既に正常に送信されている場合、ブリッジは再送信せず、 duplicate=true を付けて同じ応答を返します。最初の送信がまだ処理中の場合は、その結果を待ちます。同じ clientMsgId で宛先または内容が異なる場合、応答は 409 です。失敗した送信は記憶されないため、同じ clientMsgId で要求を繰り返すことができます。識別子はメモリ内に保持され、プログラムの再起動時にリセットされます。

/status は、send ブロックに送信の状態を報告します。送信済みメッセージ数、直近 1 時間の送信数、待機中の要求数、次に送信可能になるまでの秒数、制限または未登録番号により拒否された要求数、重複数です。

6. データベースの構成

SQL Server、MySQL、MariaDB、ODBC 経由の SQLite、および組み込みの SQLite がサポートされています。

```
TypeDB=InternalSqlite
#ConnStr=
# senza ConnStr:
# PathWork\Instance_<CodeInstance>\Albatros_Embedded.db
```

InternalSqlite では、データベース用の ODBC ドライバーをインストールする必要はありません。ファイルが別のプログラムに使用されている場合、ブリッジは短い間隔で何回か再試行し、データをキューに保持します。

テーブル	内容
hh_Albatros_Webhooks	メッセージ/イベントのヘッダーと完全な RawData。
hh_Albatros_WebHooks_Media	ヘッダーに関連付けられたマルチメディア添付ファイル。
hh_Albatros_WebHooks_Locations	位置情報タイプのメッセージの位置 (緯度と経度)。ヘッダーに関連付けられます。
hh_Albatros_WebHooks_Secrets	メッセージの messageSecret。リンクした番号と Key はこのテーブルには保存されません。MsgID を介して hh_Albatros_Webhooks と JOIN することで取得します。
hh_Albatros_Webhooks_log	診断ログ。
hh_Albatros_SendQueue	別のプログラムが書き込んだ送信対象のメッセージ (データベースからの送信)。DbSend=Y の場合にのみ作成されます。15 を参照してください。

テーブルが存在しない場合は自動的に作成されます。また、起動時に、元のメッセージの検索 (編集の処理に必要) を高速化するため、ヘッダーの MsgID 列にインデックス IX_hh_Albatros_Webhooks_MsgID が存在しなければ作成します。非常に大きなテーブルでは作成に数分かかる場合があり、その間メッセージはキューに保持されます。データベース ユーザーにインデックスの作成権限がない場合は、警告が表示されるだけで、プログラムは通常どおり続行します。

各メッセージは、キューに入った時点で一意の識別子 (MsgUID) を受け取ります。保存が既に完了していても、キューファイルを削除する前にプログラムが中断した場合、次の試行時にその行が認識され、2 回目の挿入は行われません。

7. データベース パスワードと WhatsApp セッションの暗号化

暗号化されたパスワードは、PC のハードウェア要素に結び付けられます。生成するには:

```
Albatros-Bridge-odbc.exe --pwd
```

PowerShell が利用可能な場合、パスワードは非表示入力で 2 回要求されます。完了すると、構成ファイルにコピーする PwdConnStr=ENC:... の行が生成されます。ある PC で暗号化したパスワードは、別の PC では使用できません (マザーボード、CPU、システム ディスクを交換した後も同様です)。その場合は再生成する必要があります。同じ暗号化値を DbSendSecret に使用することもできます。プログラムは DbSendSecret=ENC:... の行も提示します (15 を参照)。

7.1 WhatsApp セッションの暗号化

auth_info フォルダー内の WhatsApp セッション ファイルは、常に PC のハードウェア要素に結び付けられたキーで暗号化されます。このキーはデータベース パスワードに使用されるキーとは異なります。暗号化は自動で常に有効であり、構成はできません。操作は一切不要です。

• auth_info フォルダーを別の PC にコピーしても、それを使用してインスタンスの WhatsApp 番号で接続することはできません。

• 以前のバージョンからの更新: 初回起動時に、既存の平文ファイルは自動的に暗号化されます。 QR を再スキャンする必要はありません。

• PC またはハードウェアの変更 (マザーボード、CPU、C ドライブ、または CPU が異なるホストへの仮想マシンの移動): セッションは復号できなくなり、新しい QR で再接続する必要があります。

• この場合、フォルダーは削除されず、auth_info.undecryptable_<data e ora> (日時) に名前が変更され、新しいセッションが作成されます。新しい接続を確認した後、名前を変更したフォルダーは削除してかまいません。電話の [リンク済みデバイス] で、以前のデバイスを削除することをお勧めします。

• パラメーター EncryptAuth はサポートされていません。構成ファイルに存在する場合は、警告を表示して無視されます。

起動時、ハードウェア要素の読み取りに数秒かかる場合があります。最初の確認が失敗した場合、プログラムはセッションを復号不能と判断する前に確認を繰り返します。

暗号化により別の PC でのセッションの使用は防止されますが、PC 自体の保護に代わるものではありません。ブリッジを実行するアカウントの権限で PC にアクセスできる者は、セッションを使用できます。したがって、サーバーへのアクセスを保護してください。

8. プログラムの起動

構成の検索順序: --config、PHONES_BRIDGE_CONFIG、プログラム フォルダー内の Instance_Config.ini、従来のパス、最後に警告付きの既定値。

起動時には常にバナーと構成の概要が表示されます。概要には、BindAddress、ApiToken (設定されているかどうかのみで、値は表示されません)、MaxMediaMB、送信のパラメーター (SendMinIntervalSec、SendMaxPerHour、SendMaxWaitSec、SendCheckNumber、SendIdempotencyHours)、データベースからの送信 (DbSend と、有効な場合は DbSendSecret。暗号化か平文かのみで、値は表示されません)、およびセッションの暗号化を確認する Session files 行も

含まれます。概要の最後には、DebugLog、DebugLogFilter、受け入れるメディアの除外指定、および条件に該当する場合は SECURITY 警告が表示されます。

```
MaxMediaMB       : 100 MB
MaxMemQueue      : 1500
SendMinIntervalSec   : 3s between two sends (+ up to 30% random)
SendMaxPerHour     : 60 sends per hour
SendMaxWaitSec     : 60s maximum wait in the send queue, then HTTP 429
SendCheckNumber    : Y (recipient verified before sending)
SendIdempotencyHours : 24h (repeated clientMsgId is not sent again)
DbSend         : Y (table hh_Albatros_SendQueue, read every 5s, rows valid 60 minutes)
DbSendSecret      : set (encrypted, bound to this PC)
ConsoleLog       : Y (console debug enabled)
DebugLog        : All (DEBUG, ERROR, INFO, WARNING)
DebugLogFilter     : NO-VIDEO, NO-OTHER
Accepted media     : AUDIO, PDF

⚠ SECURITY: the HTTP server is reachable from the network without ApiToken: ...
```

概要には、10.6 で説明する起動キューのフォルダー (Startup queue folder) も表示されます。

ユーザー マニュアル: 起動時に、イタリア語と英語のマニュアルが --man_IT および --man_EN と同じ既定の名前でプログラム フォルダーに保存されます。ただし、まだ存在しない場合に限ります。既存のファイルが上書きされることはありません。作成されたファイルごとに情報行が表示されます。書き込みができない場合は警告が表示され、プログラムは続行します。自動保存は、パラメーター --Skip-Manuals、または構成ファイルの Skip-Manuals=Y で無効にできます。

処理されない予期しないエラーが発生した場合、プログラムはそれをログ (画面、Connection.log、ログ テーブル) に書き込み、約 2 秒後に終了コード 1 で終了します。継続的に運用するには、エラー時に自動的に再起動するよう構成した Windows サービスとして実行することをお勧めします。

9. WhatsApp への初回接続

• 有効な資格情報が存在しない場合、QR が表示されます。

• 同じ QR が QrCode\latest.png に保存され、エンドポイント /qr からも取得できます (ApiToken が設定されている場合はトークンが必要)。

• 電話で: [設定] → [リンク済みデバイス] → [デバイスをリンク]。

• スキャン後、WhatsApp は接続の再起動を要求します。ブリッジは直ちに再接続します。

• 接続が開くと、QR ファイルは削除されます。

• 新しいセッションのファイルは、最初から暗号化された状態で作成されます (7.1 を参照)。

• 自動再接続中は、本当に必要な場合を除き、QR は表示も保存もされません。

• ログアウト後、セッションは削除され、再び QR が要求されます。

10. 全般的な動作

各メッセージ/イベントは分類され、必要に応じて以前のメッセージに関連付けられ、シリアル化されて保存フローに投入されます。各要素はディスク上の永続キューを経由し、ファイルはデータベースへの保存が成功した後にのみ削除されます。

10.1 再試行とキュー

• DbRetrySec と上限 DbRetryMaxSec による線形バックオフ。新しいデータが届くたびに、いずれにしても即時の試行が行われます。

• データベースへの最初のアクセスが成功すると、試行カウンターがリセットされます。

• キュー ファイルは安全な方法 (一時ファイルと名前変更) で書き込まれます。突然中断しても不完全なファイルは残りません。

• データベースの可用性に関するエラー (ネットワーク、サーバーの停止、データベースのビジー) は、DbMaxRetries と PendingMaxSec に従って再試行されます。

• 読み取れないキュー ファイルや、データベースに拒否されたデータ (無効な値など) は、後続の要素をブロックしないよう、直ちに pending_db\failed に移動されます。

• 上限を超えた要素は pending_db\failed に移動されます。原因は画面とログに表示されます。

• メッセージの保存が成功した後に、ExeName を起動できます (この項の後半を参照)。

failed に移動した要素を再試行するには、原因を修正した後、ファイルを pending_db\failed から pending_db に移動してブリッジを再起動します。pending_db 内のファイルは起動時に読み込まれます。

挿入後の実行可能ファイル。 ExeName が設定されている場合、データベースにメッセージが保存されるたびに (messageSecret の場合を除く)、ブリッジは PathBin\ExeName を終了を待たずに起動し、次の 3 つの個別のパラメーターを渡します。

```
--rec=<ID>      ID della riga in hh_Albatros_Webhooks
--istance=<istanza> codice istanza (CodeInstance) che ha registrato la riga
--Cell=<numero>   numero collegato all'istanza (PhoneConnected), solo cifre
```

起動されたプログラムが受け取るコマンド ラインの例:

```
GestioneMessaggi.exe --rec=45 --istance=Magazzino --Cell=393479528620
```

値は行に記録されたもの (ID、Instance、PhoneConnected 列) です。スペースを含むインスタンス コードも、引用符で囲まれた単一のパラメーターとして渡されます。実行可能ファイルが存在しない場合は警告が 1 回だけ表示され、保存は通常どおり続行します。起動時の概要には、Exe の横にパラメーターの形式が表示されます。以前のバージョンでは、パラメーターはレコードの ID のみでした。

10.2 編集されたメッセージ

Save_Secrets=S の場合、messageSecret はヘッダーの保存フィルターより前に取得されます。ただし、構成で除外されたメッセージ (Save_Sent=N、Save_StatusBroadcast=N、Save_protocolMessage=N) は除きます。Save_Sent と Save_StatusBroadcast で除外されたメッセージの編集と削除は保存されません。現在のロジックでは LID で暗号化された編集も処理し、必要に応じて、元のメッセージをデータベースから検索して取得します (同じ MsgID を持つステータス更新の行は無視します)。まだ保存されていない場合は、ディスク上のキューから取得します。

10.3 RelatedMsgID

イベント	取得元
返信 / 引用	contextInfo.stanzaId
リアクション	reactionMessage.key.id
プロトコル	protocolMessage.key.id
編集	targetMessageKey.id

10.4 メディアと位置情報

• メディアのダウンロードは、試行の間に短い待機時間を置いて最大 3 回試行されます。

• ダウンロードが失敗した場合、またはメディアが MaxMediaMB を超える場合でも、メッセージは添付ファイルなし (MediaCount=0) で保存されます。ログにエラーが記録され、/messages が返す要素には原因を示す mediaError フィールドが含まれます。

• 座標が無効な位置情報でもメッセージは失われません。hh_Albatros_WebHooks_Locations の行なしで、locationError フィールド付きで保存されます。元の座標は RawData に残ります。

• 連絡先とアンケートは、タイプ contact (テキスト = 表示名) と poll (テキスト = 質問) として記録されます。完全な内容は RawData に残ります。

• 処理中にその他のエラーが発生した場合、そのメッセージは「処理済み」とは見なされません。そのため、WhatsApp から再配信された場合は通常どおり処理されます。

10.5 自動再接続

切断の原因	動作
ネットワークがない、または接続が失われた	待機時間を増やしながら再試行します: 約 1、2、4、8 ... 秒、最大 60 秒まで。接続が再び開くとカウンターはリセットされます。
WhatsApp による再起動の要求 (QR のスキャン後)	直ちに再接続します。
セッションが置き換えられた	別のインスタンスが同じ資格情報を使用しています。エラーが表示され、次の試行は 60 秒後に行われます。
電話からのログアウト	資格情報が削除され、新しい QR が要求されます。

待機中、/status は connected=false、試行回数 (reconnectAttempt)、残り秒数 (reconnectInSec) を報告します。この間に受信した送信要求には HTTP 503 が応答されます。

10.6 リンクした番号が利用可能になる前に受信したメッセージ

起動時、サーバーに待機中のメッセージがある場合、WhatsApp はリンクした番号が利用可能になる前にそれらを配信することがあります。PhoneConnected フィールドが空のまま記録されるのを避けるため、ブリッジはそれらを起動キューに入れ、番号が判明した時点で記録します。

• この段階で受信した各イベント (メッセージ、編集、削除、ステータス更新) は、突然の中断から保護された方法で、インスタンスの early_queue フォルダー内のファイルに書き込まれます。プログラムが終了した場合、次回の起動時に既存のファイルが再読み込みされ、リンクした番号を待機します。

• 番号が利用可能になると、イベントは WhatsApp 内部の日時に従って、古いものから新しいものへ 1 件ずつ記録されます。同じ秒内では、メッセージがその編集より先に、編集が削除より先になります。いずれの場合も、編集、削除、リアクション、ステータス更新は、参照先のメッセージの後に記録されます。そのため、編集を記録する時点で元のメッセージの messageSecret は既に利用可能です。

• キューを空にしている最中に到着したイベントもキューに入れられます。古いイベントより先に記録されるイベントはありません。

• イベントのファイルは、記録後にのみ削除されます。記録後、削除前にプログラムが中断した場合、再起動時にイベントは再度処理されますが、重複した行は作成されません。

• 読み取れないファイル、または処理できなかったイベントは early_queue\failed に移動されます。原因は画面とログに表示されます。

• 待機が 30 秒を超えると、キュー内のイベント数を示す警告が表示され、60 秒ごとに繰り返されます。/status は、キュー内のイベント数を startupQueue フィールドで報告します。

• キューから記録された行では、DateReceived は記録した時点を示します。WhatsApp の日時 (DateWA) は元の値のままです。

• イベントが長時間起動キューに残っていると、WhatsApp のリンクの有効期限が切れるため、メディアをダウンロードできなくなる場合があります。この場合、10.4 で説明したとおり、メッセージは添付ファイルなしで保存されます。

11. HTTP インターフェイス (エンドポイント)

ベース アドレス: http://<indirizzo>:<Port> — 既定のポートは 3000 です。ApiToken が設定されている場合、すべての要求にトークンを含める必要があります (5.3 を参照)。POST で送信するデータは JSON 形式 (Content-Type: application/json) です。

メソッド	エンドポイント	説明
GET	`/status`	接続、インスタンス、DB、キュー、起動キュー (startupQueue)、再接続、送信 (send)、データベースからの送信 (sendQueue) の状態と統計情報。
GET	`/qr`	現在の QR。利用できない場合は 404。
GET	`/messages`	メッセージ キューを返して空にします。
GET	`/messages?peek=1`	メッセージ キューを空にせずに返します。各要素には seq 番号が付きます。
POST	`/messages/ack`	処理済みのメッセージを確認します: {"upToSeq": N} で seq が N 以下のものを削除します。
GET	`/message-status`	ステータスの更新を返して空にします。
GET	`/message-status?peek=1`	上記と同じですが、空にしません (seq 付き)。
POST	`/message-status/ack`	処理済みのステータス更新を確認します ({"upToSeq": N})。
GET	`/debug-log`	診断ログの状態。
GET	`/db-queue`	キュー、再試行、パラメーター。
POST	`/db-retry`	即時の試行を強制します。
GET	`/config`	パスワードとトークンを除いた現在の構成。
GET	`/message-secret/:msgId`	メッセージの MessageSecret。
POST	`/send`	テキストを送信します (送信の制限、番号の確認、任意の clientMsgId: 5.5 を参照)。
POST	`/send-media`	base64 でメディアを送信します (/send と同じチェック)。

メソッド	エンドポイント	説明
POST	/send-queue/poll	送信テーブルの読み取りを前倒しします (15 を参照)。応答は 202。データベースからの送信が無効な場合は 409 DB_SEND_DISABLED。

11.1 メッセージを失わない読み取り (peek / ack)

/messages の従来の読み取りでは、応答した時点でキューが空になります。クライアントが応答を処理する前に中断すると、それらのメッセージは再び返されることはありません (ただし、データベースには保存されたままです)。2 段階の読み取りにより、この問題を回避できます。

• GET /messages?peek=1: キュー内のメッセージを削除せずに返します。各メッセージには連番 seq が付きます。

• クライアントがメッセージを処理します。

• {"upToSeq": <seq dell'ultimo messaggio elaborato>} (最後に処理したメッセージの seq) を付けて POST /messages/ack を送信: 確認されたメッセージをキューから削除します。確認前にクライアントが停止した場合は、次回の読み取りでそれらを再び取得できます。

```
GET /messages?peek=1
{ "key": "Albatros", "messages": [ { "id": "3EB0...", "type": "text", "text": "ciao", ..., "seq": 41 },
                { "id": "3EB0...", "type": "image", ..., "seq": 42 } ] }

POST /messages/ack     { "upToSeq": 42 }
{ "success": true, "removed": 2, "remaining": 0 }
```

ただし、キューは MaxMemQueue 個の要素に制限されます。クライアントが長時間確認しない場合、最も古い要素は破棄されます。/message-status と /message-status/ack にも同じ方式が適用されます。従来の読み取りと 2 段階の読み取りを、異なるクライアントから同時に使用しないでください。

11.2 メッセージの送信

宛先 (number) には、国際プレフィックス付きの番号 (+、スペース、ハイフン、かっこを含んでもかまいません)、または完全な WhatsApp 識別子 (@s.whatsapp.net、グループ @g.us、@lid) を指定できます。

```
POST /send
{ "number": "+39 340 123 4567", "message": "Testo del messaggio",
 "clientMsgId": "ORD-2026-000123" }

POST /send-media
{ "number": "393401234567", "mediaType": "document", "fileName": "fattura.pdf",
 "caption": "Fattura di ottobre", "base64Data": "JVBERi0xLjcK..." }

Risposta: { "success": true, "id": "<id del messaggio inviato>" }
Con clientMsgId: { "success": true, "id": "<id>", "clientMsgId": "ORD-2026-000123",
```

```
 "duplicate": false }
Errore: { "success": false, "code": "SEND_HOURLY_LIMIT",
 "error": "Hourly send limit reached", "retryAfterSec": 1250 }
```

mediaType	送信内容
image	画像。キャプションを付けることもできます。
audio	音声 (audio/mpeg として送信)。
その他の値	ドキュメント: fileName (既定値 file.pdf) と、必要に応じてキャプション。

clientMsgId フィールドは任意です (5.5 を参照)。/send と /send-media のエラー応答には、error に加えて code フィールドが含まれます。クライアントはテキストの代わりにこれを使用できます。

code (HTTP)	意味
INVALID_NUMBER (400)	number がない、または無効です。
INVALID_MESSAGE (400)	message がない、または空です (/send)。
INVALID_MEDIA (400)	base64Data がない、または空です (/send-media)。
INVALID_CLIENT_MSG_ID (400)	clientMsgId が無効です (テキストでない、空、または 100 文字を超える)。
CLIENT_MSG_ID_CONFLICT (409)	clientMsgId が別のメッセージで既に使用されています。
NOT_ON_WHATSAPP (422)	番号が WhatsApp を使用していません (SendCheckNumber=Y)。再試行しないでください。
SEND_HOURLY_LIMIT (429)	SendMaxPerHour に達しました。retryAfterSec 秒後に再試行してください。
SEND_QUEUE_BUSY (429)	キューでの待機が SendMaxWaitSec を超えます。retryAfterSec 秒後に再試行してください。
SEND_ERROR (500)	送信中に WhatsApp からエラーが返されました。
NOT_CONNECTED (503)	WhatsApp に接続されていません。

11.3 応答コード

コード	意味
200	要求が実行されました。
400	データがない、または無効です (例: 無効な number、空の message、base64Data がない、upToSeq が数値でない、JSON の形式が正しくない)。
401	トークンがない、または正しくありません (ApiToken が設定されている場合のみ)。
404	リソースを利用できません (例: QR が存在しない)。
409	clientMsgId が別のメッセージで既に使用されています (/send、/send-media)。
413	要求本文が大きすぎます (50 MB 超)。

コード	意味
422	宛先の番号が WhatsApp を使用していません (SendCheckNumber=Y)。
429	送信制限 (SendMaxPerHour) に達したか、キューでの待機が SendMaxWaitSec を超えます。 retryAfterSec 秒後に再試行してください (Retry-After ヘッダー)。
500	内部エラー、または送信中に WhatsApp から返されたエラー。
503	WhatsApp に接続されていない、またはデータベースを利用できません (/message-secret)。

11.4 C#.NET クライアントの例

```
using System;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;

var http = new HttpClient { BaseAddress = new Uri("http://127.0.0.1:3000/") };
http.DefaultRequestHeaders.Authorization =
  new AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale");

// Lettura senza svuotare la coda
string json = await http.GetStringAsync("messages?peek=1");
// ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

// Conferma dei messaggi elaborati
var ack = new StringContent("{\"upToSeq\":" + ultimoSeq + "}", Encoding.UTF8, "application/json");
HttpResponseMessage r = await http.PostAsync("messages/ack", ack);
```

11.5 VB.NET クライアントの例

```
Imports System.Net.Http
Imports System.Net.Http.Headers
Imports System.Text

Dim http As New HttpClient()
http.BaseAddress = New Uri("http://127.0.0.1:3000/")
http.DefaultRequestHeaders.Authorization =
  New AuthenticationHeaderValue("Bearer", "IlMioTokenLungoECasuale")

' Lettura senza svuotare la coda
Dim json As String = Await http.GetStringAsync("messages?peek=1")
' ... elaborazione; ultimoSeq = seq dell'ultimo messaggio elaborato ...

' Conferma dei messaggi elaborati
Dim ack As New StringContent("{""upToSeq"":" & ultimoSeq & "}", Encoding.UTF8, "application/json")
Dim r As HttpResponseMessage = Await http.PostAsync("messages/ack", ack)

' Invio con clientMsgId: nuovo per ogni messaggio, riutilizzato solo nei ritentativi
Dim idMsg As String = Guid.NewGuid().ToString()
Dim body As String = "{""number"":""393401234567"",""message"":""Testo"",""clientMsgId"":""" & idMsg & """}"
Dim s As HttpResponseMessage = Await http.PostAsync("send", New StringContent(body, Encoding.UTF8, "application/json"))
Select Case CInt(s.StatusCode)
  Case 429 ' limite raggiunto: attendere e ritentare con lo stesso idMsg
    Dim attesa As TimeSpan = If(s.Headers.RetryAfter?.Delta, TimeSpan.FromSeconds(60))
  Case 422 ' numero non registrato su WhatsApp: non ritentare
End Select
```

12. 接続ログと画面ログ

Connection.log は PathWork\Instances_Log\<numero connesso>\Connection.log (接続している番号のフォルダー) にあります。接続ログとテキスト出力のコピーが含まれます。

内容	表示
バナー	常時
起動時の概要	常時
SECURITY 警告	常時 (ApiToken なしでサーバーがネットワークから到達可能な場合)
QR	実際に必要な場合は常時
Warning / Error	常時
処理されないエラー	常時 (プログラムの終了前)
INFO / DEBUG	ConsoleLog=Y の場合
通常のログ	先頭に短縮形式の日時

インターネット接続が長時間失われている間、連続する同一の行 "Connection closed. Reconnecting: true" は、画面とログの両方で 1 回だけ記録されます。

13. 診断とトラブルシューティング

症状	対処
構成ファイルが見つからない	--create_config= または --config= を使用します。
DB 接続に失敗した	TypeDB、ConnStr、PwdConnStr を確認します。データはキューに保持されます。
ENC パスワードを復号できない	ブリッジを実行する PC で --pwd を使用して再生成します。
SQLite がビジー	自動再試行を待ちます。どのプログラムがファイルを開いているかを確認します。
大きなデータベースで初回起動が遅い	MsgID のインデックスを作成中です。待機してください。メッセージはキューに保持されます。
クライアントがブリッジに到達できない	クライアントが別の PC にある場合: BindAddress を空または 0.0.0.0 にし、ポートのファイアウォール規則と ApiToken を設定します。同じ PC にある場合: http://127.0.0.1:<Port> を使用します。

症状	対処
HTTP 401 応答	トークンがない、または ApiToken と異なります。Authorization: Bearer ヘッダーまたは X-Api-Key ヘッダーを確認します。
/send で HTTP 400 応答	number (プレフィックス付きの番号または JID) と message (空でないこと) を確認します。
/send で HTTP 503 応答	WhatsApp に接続されていません。/status (connected、reconnectInSec) を確認します。
/send で HTTP 429 応答	送信制限に達しました。retryAfterSec 秒待ってから再試行します。頻繁に発生する場合は、送信のペースを落とすか、SendMinIntervalSec、SendMaxPerHour、SendMaxWaitSec を調整します (5.5 を参照)。
/send で HTTP 422 応答	番号が WhatsApp を使用していません。番号と国際プレフィックスを確認します。この確認は SendCheckNumber=N で無効にできます。
/send で HTTP 409 応答	同じ clientMsgId が別のメッセージに使用されました。クライアントはメッセージごとに新しい clientMsgId を生成し、再試行時にのみ再利用する必要があります。
起動時の SECURITY 警告	ApiToken または BindAddress=127.0.0.1 を設定します。
メッセージがメディアなしで保存された	ERROR ログ (ダウンロードの失敗または MaxMediaMB の超過) と mediaError フィールドを確認します。
pending_db\failed 内のファイル	ログで原因を確認します。修正後、ファイルを pending_db に移動して再起動します。
警告 "Connected number still not available"	リンクした番号がまだ利用できません。受信したメッセージは起動キューに保持され、番号が判明した時点で記録されます (10.6 を参照)。警告が続く場合は、/status で接続を確認します。
early_queue\failed 内のファイル	ログで原因を確認します。修正後、ファイルを early_queue に移動して再起動します。
プログラム フォルダーに不要なマニュアルがある	--Skip-Manuals で起動するか、Skip-Manuals=Y を設定します。既存のファイルは削除してかまいません。
エラー "session replaced"	別のインスタンスが同じ auth_info フォルダーを使用しています。1 つだけを有効にしてください。

症状	対処
PC またはハードウェアの変更後に新しい QR が要求される	想定どおりの動作です。セッションは PC に結び付けられています (7.1 を参照)。新しい QR をスキャンし、確認後に auth_info.undecryptable_<data> フォルダーを削除して、電話の [リンク済みデバイス] から古いデバイスを削除します。
別の PC からコピーした auth_info フォルダーが機能しない	想定どおりの動作です。暗号化されたファイルは、それを作成した PC でのみ使用できます。新しい QR でインスタンスを接続してください。
起動時の警告 "EncryptAuth is ignored"	セッションは常に暗号化されます。構成ファイルから EncryptAuth の行を削除してください。
起動時の DbSendSecret に関するエラー	シークレットがない、32 文字より短い、またはこの PC で復号できません。データベースからの送信は無効のままです。DbSendSecret を設定するか、ブリッジの PC で --pwd を使用して再生成します。
送信テーブルの行が状態 K になる	書き込み側のプログラムが DbSendSecret とは異なるシークレットを使用しているか、署名後にデータが変更されています。MySQL では utf8mb4 エンコードを確認します。
送信テーブルの行が状態 N のまま	DbSend=Y、接続 (/status の connected)、行の PhoneSender がリンクした番号であること、および送信制限を確認します。
送信テーブルの行が状態 X になる	行が読み取りよりかなり前に書き込まれたか、PC の時計がずれています。DbSendMaxAgeMin を増やす (最大 1440) か、時計を同期します。
送信テーブルの行が状態 U になる	同じ行の送信中にブリッジが 2 回停止しました。hh_Albatros_Webhooks でメッセージが送信されたかを確認し、必要に応じて新しい ClientRef で再挿入します。
プログラムがエラーで終了する	Connection.log を確認します。ブリッジを自動再起動付きのサービスとして実行します。
コンソールに INFO/DEBUG が表示されない	ConsoleLog=Y を設定します。
QR が表示されない	セッションがまだ有効かどうかを確認します。
編集が復号されない	Save_Secrets=S と messageSecret を確認します。
テーブルのログがない	DebugLog=All または種類の一覧を設定します。
DebugData に不要なメディアがある	DebugLogFilter を構成します。

症状	対処
WhatsApp 番号がブロック (BAN) された	ブロックは Meta が決定して適用するもので、通常は WhatsApp の規約違反 (スパム、広告、一斉送信) が原因です。電話のアプリから WhatsApp と直接対応する必要があります。冒頭の注意事項を参照してください。
WhatsApp の更新後にプログラムが接続できなくなった	WhatsApp によるプロトコル変更の可能性があります (冒頭の注意事項を参照)。プログラムの更新版が利用可能かどうかを確認してください。
ハードウェアを変更していないのに ENC パスワードまたはセッションを復号できない	プログラムのアカウントで PowerShell が利用可能であること、および WMI サービスが有効であることを確認します。

14. ベスト プラクティス

• 外部 DB が不要な場合や保護された環境で運用する場合のテスト インストールには、InternalSqlite を使用します。

• 外部データベースを使用する運用環境では、PwdConnStr=ENC:... を推奨します (InternalSqlite ではパスワードは不要です)。

• ブリッジが他の PC から到達可能な場合は、必ず ApiToken を設定します。クライアントが同じ PC にある場合は BindAddress=127.0.0.1 を使用します。

• クライアントでは、従来の読み取りよりも 2 段階の読み取り (?peek=1 と /ack) を推奨します。

• 編集が必要な場合は Save_Secrets=S を維持します。

• /status と /db-queue を監視し、pending_db\failed と early_queue\failed を定期的に確認します。

• ブリッジは、エラー時に自動再起動する Windows サービスとして実行します。

• スパム、商業広告、一斉送信に本プログラムを使用しないでください。これらは WhatsApp の規約で禁止されており、番号のブロックにつながる可能性があります。

• 送信制限 (SendMinIntervalSec、SendMaxPerHour) を有効にしたままにし、メッセージごとに一意の clientMsgId を送信して、クライアントが同じ要求を繰り返す場合にのみ再利用します。

• データベースからの送信には暗号化された DbSendSecret (ENC:...) を使用し、シークレットをデータベースに書き込まないでください。また、状態 E、K、X、U のまま残った行を定期的に削除します。

• auth_info フォルダーは暗号化され PC に結び付けられているため、別の PC にコピーしても役に立ちません (いずれにしても新しい QR が必要です)。それでも、アクセスはブリッジを実行するアカウントに制限してください。バックアップからの復元は同じ PC でのみ機能します。

• 複数の WhatsApp 番号には別々の CodeInstance を使用します。同じ auth_info を 2 つのインスタンスで使用しないでください。

• 診断中は DebugLog と ConsoleLog を有効にします。

• DebugLogFilter を設定して診断用のメディアを制限し、使用可能なメモリに応じて MaxMediaMB を設定します。

• PendingMaxSec と DbMaxRetries を運用環境に合わせて調整します。

15. データベースからの送信 (送信テーブル)

HTTP エンドポイントに加えて、ブリッジは別のプログラムが hh_Albatros_SendQueue テーブルに書き込んだメッセージを送信できます。DbSend=Y の場合、ブリッジは DbSendPollSec 秒ごとにテーブルを読み取り、リンクした番号の行のみを、/send または /send-media から届いたものと同様に送信します。キュー、制限 (SendMinIntervalSec、SendMaxPerHour、SendMaxWaitSec)、番号の確認 (SendCheckNumber) も同じです。

各行は、ブリッジと書き込み側のプログラムだけが知っているシークレットで署名されます。シークレットを持たずにデータベースにアクセスしても、有効な行を作成したり内容を変更したりすることはできません。

15.1 有効化

• DbSend=Y でこの機能が有効になります。テーブルは起動時に自動的に作成されます。作成時にエラーが発生した場合は報告されますが、メッセージの保存は妨げられません。

• DbSendSecret は署名用のシークレット (最小 32 文字) で、平文、または --pwd で生成した ENC:... です。有効なシークレットがない場合、この機能は無効のままとなり、起動時にエラーとして報告されます。同じシークレットを、行を書き込むプログラムも知っている必要があり、データベースには決して書き込まないでください。

• DbSendPollSec (1 〜 300、既定値 5): 読み取り間隔の秒数。テーブルは、ブリッジが接続され、リンクした番号が判明している場合にのみ読み取られます。

• DbSendMaxAgeMin (既定値 60): 行の有効時間 (分)。30 未満の値は 30 に、1440 を超える値は 1440 になります。

• POST /send-queue/poll で読み取りを前倒しします (応答 202)。短時間に複数の要求があっても、読み取りは 1 回だけです。

```
DbSend=Y
DbSendSecret=ENC:...
DbSendPollSec=5
DbSendMaxAgeMin=60
```

15.2 テーブルの構造

署名対象と記載された列は署名に含まれます (15.4 を参照)。書き込み側は SendStatus = N で行を挿入します。ブリッジの列はブリッジが管理します。

列	書き込み元	説明
ID	データベース	連番の識別子。行は ID の順に処理されます。
DateInsert	データベース	挿入日時 (情報のみ)。
PhoneSender	書き込み側	署名対象。送信元の番号 (数字のみ)。ブリッジは自分の番号の行のみを読み取ります。
ClientRef	書き込み側	署名対象。PhoneSender ごとの一意の参照 (一意インデックス)。最大 100 文字。
Recipient	書き込み側	署名対象。宛先: 国際プレフィックス付きの番号、または JID (グループは @g.us)。
MsgKind	書き込み側	署名対象。メッセージの種類 (15.3 を参照)。
Title	書き込み側	署名対象。場所の名前、連絡先の名前、またはアンケートの質問。
MsgText	書き込み側	署名対象。テキスト、キャプション、住所、vCard、またはアンケートの選択肢。
FileName	書き込み側	署名対象。ドキュメントの名前。
MimeType	書き込み側	署名対象。メディアの MIME タイプ。任意。
Latitude, Longitude	書き込み側	署名対象。小数点にピリオドを使用したテキスト形式の座標。
MediaData	書き込み側	SHA-256 を介して署名対象。バイナリ形式の添付ファイル (base64 ではありません)。
CreatedAtUtc	書き込み側	署名対象。作成日時 (1970/01/01 からの UTC ミリ秒)。
SignVer	書き込み側	署名形式のバージョン: 1。
Signature	書き込み側	16 進数の HMAC-SHA256 署名 (64 文字)。
SendStatus	書き込み側 / ブリッジ	行の状態 (15.5 を参照)。書き込み側は N を挿入します。
ClaimToken, ClaimedBy, ClaimedAtUtc	ブリッジ	行の取得 (処理の引き受け)。

列	書き込み元	説明
Attempts	ブリッジ	既に使用した追加の試行回数 (最大 1)。
SentMsgID	ブリッジ	メッセージの識別子。送信前に決定されます。
ErrorCode, ErrorText, ProcessedAtUtc	ブリッジ	送信されなかった行の結果。

15.3 メッセージの種類 (MsgKind)

MsgKind	使用する列	備考
text	MsgText	テキストのみのメッセージ。
image	MediaData; MsgText (キャプション), MimeType	画像。
video	MediaData; MsgText (キャプション), MimeType	既定の MIME は video/mp4。
gif	MediaData (mp4); MsgText (キャプション)	アニメーション GIF として再生される動画。
audio	MediaData; MimeType	既定の MIME は audio/mpeg。
voice	MediaData; MimeType	ボイス メッセージ。既定の MIME は audio/ogg; codecs=opus。
document	MediaData, FileName; MsgText (キャプション), MimeType	MimeType が空の場合、MIME は拡張子から決定されます。
sticker	MediaData (webp)	ステッカー。
location	Latitude, Longitude; Title (名前), MsgText (住所)	緯度は -90 ～ 90、経度は -180 ～ 180。
contact	MsgText (1 つ以上の vCard); Title (表示名)	各 vCard は BEGIN:VCARD から END:VCARD まで。
poll	Title (質問), MsgText (選択肢、1 行に 1 つ)	選択肢は 2 ～ 12 個。単一選択。
pollmulti	Title (質問), MsgText (選択肢、1 行に 1 つ)	poll と同様で、複数選択。

メディアは MaxMediaMB を超えることはできません。リアクション、編集、削除、転送は既存のメッセージが必要なため、処理されません。

15.4 署名

Signature は、UTF-8 の DbSendSecret シークレットをキーとして、次の UTF-8 テキストから算出した HMAC-SHA256 です。

```
ALB-SEND-1
<lunghezza>:<PhoneSender>
<lunghezza>:<ClientRef>
<lunghezza>:<Recipient>
<lunghezza>:<MsgKind>
<lunghezza>:<Title>
<lunghezza>:<MsgText>
<lunghezza>:<FileName>
```

```
<lunghezza>:<MimeType>
<lunghezza>:<Latitude>
<lunghezza>:<Longitude>
<lunghezza>:<MediaSha256>
<lunghezza>:<CreatedAtUtc>
```

• 各行は LF 文字 (Unix の改行) のみで終わります。<lunghezza> (長さ) は値の UTF-8 バイト数です。

• NULL 値は空文字列として扱われます。MediaSha256 は MediaData の SHA-256 を小文字の 16 進数で表したもので、MediaData が NULL または空の場合は空です。CreatedAtUtc は数字で記述します。

• Signature は 16 進数 (大文字または小文字) です。比較は一定時間で行われます。

• 署名する値は、テーブルに書き込む値と同一である必要があります。MySQL では接続とテーブルで utf8mb4 を使用する必要があります。そうしないと、絵文字や特殊文字が変化し、行の署名が無効になります。

15.5 処理と結果

ブリッジは各行をアトミックに取得し (ClaimToken)、テーブルをロックしないため、同じ番号に接続された 2 つのインスタンスが同じ行を 2 回送信することはありません。その後、番号、署名、時間的な有効性を確認し、送信して結果を記録します。行は、送信キューが SendMaxWaitSec 以内に送信できる場合にのみ取得されます。

SendStatus	意味
N	送信待ち (メッセージをキューに入れる側が書き込みます)。
C	処理中。
(行は削除済み)	送信済み。メッセージは Flag_Base = N_TAB で hh_Albatros_Webhooks に残ります。メディアは再ダウンロードせずに保存されます。
E	確定エラー: ErrorCode と ErrorText に理由が示されます。
K	署名が無効: 行は送信されず、ログに SECURITY エラーが記録されます。
X	期限切れ (DbSendMaxAgeMin より古い)。
U	結果が不明: 行は再送信されません。

• ブリッジが接続されていない、または送信キューがいっぱい: 行は試行回数を消費せずに状態 N に戻ります。

• 送信中のエラー: 同じメッセージ識別子で追加の試行を 1 回だけ行い、その後状態 E になります。

• 結果のない取得 (ブリッジの停止など): 15 分と SendMaxWaitSec が経過した後、メッセージが既に hh_Albatros_Webhooks に存在する場合 (Save_Sent=S が必要) は行が削除されます。存在しない場合は追加の試行を 1 回だけ行い、2 回目には行が状態 U になります。

• 送信後にデータベースに到達できない場合、行の削除は次回の読み取り時に再度行われます。行が再送信されることはありません。

• 状態 E、K、X、U の行はテーブルに残ります。それらを書き込んだプログラムが読み取って削除する必要があります。

ErrorCode	状態	意味
INVALID_SIGNATURE	K	署名が一致しません: シークレットが異なるか、署名後にデータが変更されました。
UNSUPPORTED_SIGN_VER	K	処理できない署名のバージョンです。
EXPIRED	X	DbSendMaxAgeMin 分より古い行です。
CREATED_IN_FUTURE	E	CreatedAtUtc が 5 分以上未来です (時計がずれています)。
INVALID_CREATED_AT	E	CreatedAtUtc がない、または無効です。
INVALID_RECIPIENT	E	宛先がない、または無効です。
INVALID_KIND	E	処理できない MsgKind です。
INVALID_MESSAGE	E	種類 text のテキストがありません。
MISSING_MEDIA	E	MediaData が必要な種類で MediaData がありません。
MISSING_FILE_NAME	E	種類 document の FileName がありません。
MEDIA_TOO_LARGE	E	MediaData が MaxMediaMB を超えています。
MEDIA_READ_ERROR	E	MediaData を読み取れません。
INVALID_LOCATION	E	座標がない、または範囲外です。
INVALID_CONTACT	E	MsgText に vCard がありません。
INVALID_POLL	E	質問がない、または選択肢が 2 〜 12 の範囲外です。
NOT_ON_WHATSAPP	E	宛先が WhatsApp を使用していません。
SEND_ERROR	E	追加の試行でも送信に失敗しました。
CLAIM_TIMEOUT	N	結果のない取得: 追加の試行のために待機状態に戻されました。
UNCERTAIN	U	2 回目の取得も結果なし: 行は再送信されません。

15.6 監視

/status は sendQueue ブロックで次の情報を報告します: 有効化の状態、テーブル、読み取り間隔、行の有効期間、読み取り中かどうか、最後のエラー、および送信済み、署名無効、期限切れ、エラー、結果不明、再試行、待機状態に戻された行のカウンター。

/config は dbSend、dbSendPollSec、dbSendMaxAgeMin、およびシークレットが設定されているか (dbSendSecretSet)、暗号化されているか (dbSendSecretEncrypted) を報告しますが、その値は表示しません。起動時の概要には DbSend と DbSendSecret の行が表示されます (8 を参照)。

15.7 Albatros SendQueue プログラム

Albatros SendQueue は、送信テーブルにメッセージをキューイングします。グラフィカル インターフェイスを備えたプログラム (Albatros_SendQueue.exe) と、すべてのロジック (Instance_Config.ini の読み取り、署名、テーブルへの書き込み、行の状態、ブリッジへの HTTP 要求) を含むライブラリ AlbatrosSendQueueLib.dll で構成されます。

このライブラリは他の .NET プログラム (業務管理ソフトウェアなど) からも使用できます。共通のパラメーター (データベース、DbSendSecret、送信元番号、ブリッジのアドレス) は、Instance_Config.ini から読み取るか、構成ファイルなしで直接渡すことで、一度だけ設定します。メッセージの種類ごとに専用の送信関数 (SendText、SendImage、SendDocument、SendLocation、SendPoll...) があります。

送信元番号には、ブリッジが無視するパラメーター PhoneSender を使用します。プログラムとライブラリには、それぞれ専用のユーザー マニュアルがあります。

付録 A. 構成例

A.1 組み込み SQLite を使用した同じ PC でのテスト

```
Key=TEST
TypeDB=InternalSqlite
PathWork=c:\Albatros\Phone\
CodeInstance=Test
Port=3000
BindAddress=127.0.0.1
DebugLog=All
ConsoleLog=Y
```

A.2 SQL Server を使用し、クライアントが別の PC にある運用環境

```
Key=Albatros
TypeDB=SqlServer
ConnStr=Driver={ODBC Driver 17 for SQL Server};Server=SRVSQL;Database=AlbatrosBridge;UID=albatros_bridge;
PwdConnStr=ENC:....
PathWork=c:\Albatros\Phone\
CodeInstance=393401234567
Port=3000
; BindAddress non indicato: ascolto su tutte le interfacce di rete
ApiToken=IlMioTokenLungoECasuale
MaxMediaMB=100
DebugLog=Error, Warning
DebugLogMaxRecords=10000
PendingMaxSec=172800
```

Server、Database、UID、トークンの値は例にすぎません。

接続文字列にはパスワードを記述しないでください。パスワードは PwdConnStr に入力します (できれば --pwd で暗号化してください)。