

Smart-Mail-Flow API

Diese Seite beschreibt die maschinenlesbare Ausgabe (API) von Smart-Mail-Flow ab **Version 2.0.0** (aktuell 2.3.0). Sie richtet sich an Entwickler und an Programme oder KI-Systeme, die Testergebnisse automatisch auswerten. Was sich gegenüber älteren Versionen geändert hat, steht auf der Seite [Smart-Mail-Flow API – Änderungen in Version 2.0.0](#).

1. Ablauf eines Tests

1. **Test-ID wählen.** Die Test-ID (Kennung) ist der Teil der Empfängeradresse vor dem @. Sie bestimmt, unter welcher Adresse das Ergebnis abrufbar ist. Die Website erzeugt beim Klick auf „Test“ eine zufällige ID (z. B. `test-tmyn0ls7ona8`). Eigene IDs sind erlaubt.
2. **Mail senden** an eine dieser Adressen:
 - `<id>@smartmailflow.email` – Zustellung über IPv4 oder IPv6, wie der sendende Server es wählt
 - `<id>@ip4.smartmailflow.email` – nur über IPv4 erreichbar (prüft den IPv4-Versand)
 - `<id>@ip6.smartmailflow.email` – nur über IPv6 erreichbar (prüft den IPv6-Versand)
3. **Ergebnis abrufen:**
 - für Menschen: `https://smartmailflow.de/test?token=<id>`
 - für Programme (JSON): `https://smartmailflow.de/api?token=<id>`

Regeln für die Test-ID

- mindestens **8 Zeichen**, erlaubt sind nur `A-Z`, `a-z`, `0-9`, `_` und `-`
- Andere IDs werden schon beim Einliefern abgewiesen:
 - `550 5.1.1 id too short (minimum 8 characters)`
 - `550 5.1.1 invalid id (allowed: A-Z a-z 0-9 _ -)`
- **Wer die ID kennt, kann das Ergebnis sehen.** Für eigene Tests deshalb einen zufälligen Anteil verwenden (z. B. `meinefirma-` plus 12 Zufallszeichen) und keine erratbaren IDs wie `test1234`.
- **Jede neue Mail an dieselbe ID überschreibt das Ergebnis.** Die API liefert immer die zuletzt eingegangene Mail. Für wiederkehrende Prüfungen (Monitoring) pro Durchlauf eine neue ID verwenden.

2. Abruf

```
GET https://smartmailflow.de/api?token=<id>
```

HTTP	Antwort	Bedeutung
------	---------	-----------

200	<code>{"error": "", "data": {...}}</code>	Ergebnis vorhanden
400	<code>{"error": "No token"}</code>	ID fehlt oder ist ungültig (unter 8 Zeichen, unerlaubte Zeichen)
404	<code>{"error": "Token not found"}</code>	Zu dieser ID ist (noch) keine Mail eingegangen
429	<code>{"error": "Rate limiter"}</code>	Mehr als 300 Abrufe pro Stunde von derselben IP
500	<code>{"error": "internal error", "error_id": "..."} </code>	Interner Fehler, <code>error_id</code> bitte bei Rückfragen angeben

Warten auf das Ergebnis: Eine Mail ist normalerweise wenige Sekunden nach dem Eingang ausgewertet. Die Prüfung selbst dauert 1-5 Sekunden. Bitte während des Wartens höchstens alle 10-15 Sekunden abfragen und nach einigen Minuten aufhören.

3. Aufbau der Antwort

`data` enthält den gespeicherten Datensatz. Die Rohmail wird nie ausgegeben (`raw` ist immer "Removed").

Feld	Inhalt
<code>id</code>	laufende Nummer des Datensatzes
<code>mail_id</code>	die Test-ID
<code>queue_id</code>	Kennung der Mail beim Empfang (für Rückfragen)
<code>smtp_mail_from</code>	Absenderadresse aus dem SMTP-Umschlag (MAIL FROM, auch Envelope-From oder Return-Path genannt). Leer bei Null-Sender, z. B. bei Unzustellbarkeitsmeldungen
<code>smtp_rcpt_to</code>	Empfängeradresse aus dem SMTP-Umschlag (RCPT TO)
<code>header_from</code>	<code>From:</code> -Kopfzeile der Mail, wie sie der Empfänger sieht (eine Zeile, dekodiert)
<code>header_to</code>	<code>To:</code> -Kopfzeile (eine Zeile, dekodiert)
<code>subject</code>	Betreff (dekodiert)
<code>date_hdr</code>	<code>Date:</code> -Kopfzeile
<code>spf_status</code>	SPF-Ergebnis (wie <code>debug.spf.result</code>)
<code>dkim_status</code>	DKIM-Gesamtergebnis (wie <code>debug.dkim.result</code>)
<code>created</code>	Eingang der Mail (UTC)
<code>deleted_at</code>	gesetzt, wenn der Test gelöscht wurde. Inhalte sind dann entfernt (<code>REMOVED</code> , <code>REPLACED_...</code>)

Feld	Inhalt
raw	immer "Removed"
debug	alle Prüfergebnisse, siehe Abschnitt 4

4. Prüfergebnisse (debug)

Allgemeine Statuswerte, soweit nicht anders angegeben:

- pass = in Ordnung
- warn = Auffälligkeit, kein Fehler
- fail = Fehler
- temperror = vorübergehender DNS-Fehler, Ergebnis unbekannt
- permerror = dauerhaft fehlerhafte Konfiguration
- none / not_present = nicht konfiguriert
- not_tested = nicht prüfbar

4.1 Allgemein

Feld	Inhalt
software_version	Version von Smart-Mail-Flow, die die Mail geprüft hat (z. B. 2.2.0; 2.1.0 ergänzt smime, 2.2.0 dns_analysis, domain_security.summary und arc.aar_differences, sonst wie 2.0.0). Fehlt das Feld oder beginnt es mit 1., hat der Datensatz das ältere Format
checked_at	Zeitpunkt der Prüfung (UTC, ISO 8601)
duration_ms	Dauer der Prüfung in Millisekunden
queue_id	wie oben

4.2 smtp – Verbindung zum Smart-Mail-Flow-Server

Feld	Inhalt
client_ip, ip_version	IP-Adresse des einliefernden Servers und IP-Version (4 oder 6)
helo	Name, mit dem sich der Server gemeldet hat (HELO/EHLO)
mail_from, rcpt_to	SMTP-Umschlag (wie smtp_mail_from / smtp_rcpt_to)
tls.protocol, tls.cipher	Verschlüsselung dieser Verbindung, null = unverschlüsselt

Feld	Inhalt
tls.status	pass bei TLS 1.2 oder 1.3, warn bei älterem TLS, fail unverschlüsselt

Grenze: Bewertet wird nur die letzte Übertragung zu Smart-Mail-Flow, nicht frühere Stationen der Mail.

4.3 header – Kopfzeilen der Mail

- from[], to[]: je Adresse address, display_name (Anzeigename oder null) und display_name_is_address.
 - display_name_is_address = true heißt: Der Anzeigename ist nur die Adresse selbst, oft in Anführungszeichen ("'a@b.de'" <a@b.de>). Outlook macht das häufig. Das ist kosmetisch und kein Fehler.
- subject, date, message_id: Betreff, Datum und Message-ID der Mail.
- return_path: Return-Path: -Kopfzeile.

4.4 spf – Sender Policy Framework (RFC 7208)

Feld	Inhalt
result	pass, fail, softfail, neutral, none, permerror, temperror
identity, domain	geprüfte Identität: mailfrom = Domain aus smtp.mail_from, helo = HELO-Name (nur bei Null-Sender)
record	ausgewerteter SPF-Eintrag der Domain (bei mehreren Einträgen eine Liste, dann permerror)
mechanism	der Teil des SPF-Eintrags, der das Ergebnis bestimmt hat (z. B. ip4:192.0.2.0/24, include:spf.protection.outlook.com, -all)
dns_lookups, dns_lookup_limit	DNS-auslösende Bestandteile (include, a, mx, ptr, exists, redirect), die bei dieser Auswertung bis zum entscheidenden Eintrag gezählt wurden, und die Grenze laut RFC 7208 (10). Spätere Bestandteile zählen nicht mit, und es ist nicht die Zahl der DNS-Pakete. Für den ganzen Eintrag siehe dns_analysis.spf.dns_lookups_total
explanation	Erklärungstext

4.5 dkim – DomainKeys Identified Mail (RFC 6376, 8463)

- `result`: Gesamtergebnis. `pass`, wenn mindestens eine Signatur gültig ist, sonst das „beste“ Ergebnis der Signaturen (`fail` > `temperror` > `permerror`). `none` = nicht signiert.
- `signatures[]`: **jede Signatur einzeln.**

Feld	Inhalt
<code>domain</code> , <code>selector</code>	<code>d=</code> und <code>s=</code> der Signatur
<code>signing_algorithm</code> , <code>canonicalization</code>	z. B. <code>rsa-sha256</code> , <code>relaxed/relaxed</code>
<code>signed_headers</code>	signierte Kopfzeilen
<code>body_length_limit</code>	<code>l=</code> -Tag (nur ein Teil des Inhalts signiert), sonst <code>null</code>
<code>dns_name</code> , <code>dns_txt</code>	abgefragter DNS-Name und gefundener Schlüssel-Eintrag
<code>key_algorithm</code> , <code>key_bits</code>	<code>rsa</code> oder <code>ed25519</code> und Schlüssellänge
<code>key_strength</code>	<code>ok</code> (RSA ab 1024 Bit, ed25519), <code>weak</code> (RSA unter 1024 Bit). Bis Version 2.2.0 wurden RSA 1024–2047 Bit als <code>warning</code> bewertet
<code>recommendation</code>	ab 2.3.0: Empfehlung bei RSA unter 2048 Bit („Ein Schlüssel mit 2048 Bit ist empfohlen, sofern der Anbieter das unterstützt.“), sonst <code>null</code> . Keine Warnung: Viele Anbieter stellen nur 1024 Bit aus
<code>key_status</code>	<code>ok</code> , <code>revoked</code> (Schlüssel zurückgezogen), <code>invalid</code> , <code>unsupported</code>
<code>testing</code>	Schlüssel mit <code>t=y</code> (Testmodus)
<code>body_hash_ok</code>	<code>false</code> = Inhalt nach dem Signieren verändert, <code>null</code> = nicht prüfbar
<code>aligned_with_from</code>	Signaturdomain passt zur From-Domain (relaxed bzw. laut DMARC <code>adkim</code>)
<code>result</code> , <code>reason</code>	Ergebnis dieser Signatur. Ein fehlender Schlüssel ergibt <code>permerror</code>

4.6 `dmARC` – DMARC (RFC 7489)

Feld	Inhalt
<code>result</code>	<code>pass</code> , <code>fail</code> , <code>none</code> (kein Eintrag), <code>permerror</code> (z. B. mehrere From-Adressen), <code>temperror</code>
<code>from_domain</code> , <code>org_domain</code>	Domain aus <code>header_from</code> und ihre Organisationsdomain
<code>record</code> , <code>record_source</code>	gefundener DMARC-Eintrag und die Domain, bei der er stand
<code>published_policy</code> , <code>subdomain_policy</code>	veröffentlichte Werte <code>p=</code> und <code>sp=</code>
<code>effective_policy</code>	für diese From-Domain geltende Richtlinie (bei Subdomains <code>sp=</code> , sonst <code>p=</code>)

Feld	Inhalt
<code>pct</code> , <code>adkim</code> , <code>aspf</code>	Anteil sowie Ausrichtungsmodus (<code>r</code> = relaxed, <code>s</code> = strict)
<code>disposition</code>	was ein Empfänger nach DMARC mit dieser Mail tun soll. <code>none</code> bei <code>pass</code> oder bei <code>p=none</code> , sonst <code>quarantine</code> bzw. <code>reject</code> . Bei <code>pct</code> < 100 mit Anteil, z. B. <code>reject (50%)</code>
<code>spf_aligned</code> , <code>dkim_aligned</code>	ob SPF bzw. DKIM bestanden haben und zur From-Domain passen

`effective_policy = reject` bei `result = pass` ist richtig: Die Richtlinie ist veröffentlicht, wird aber nicht angewendet (`disposition = none`). Ein Empfänger kann aus anderen Gründen trotzdem anders entscheiden.

4.7 `arc` – Authenticated Received Chain (RFC 8617)

ARC hilft bei weitergeleiteten Mails (Verteiler, Weiterleitungen), bei denen SPF oder DKIM unterwegs brechen. Jeder Weiterleiter hält fest, was er gesehen hat, und versiegelt das.

Feld	Inhalt
<code>status</code>	<code>not_present</code> (keine ARC-Kopfzeilen, bei direkter Zustellung normal), <code>pass</code> (Kette vollständig und kryptografisch gültig), <code>fail</code> , <code>permerror</code> (Kopfzeilen fehlerhaft), <code>tempererror</code>
<code>chain_validation</code>	Ergebnis der Kettenprüfung (<code>none</code> , <code>pass</code> , <code>fail</code>)
<code>sealer_trust</code>	immer <code>unknown</code> . Ob einem Weiterleiter vertraut wird, entscheidet jeder Empfänger selbst
<code>instances[]</code>	je Station <code>i</code> , <code>seal_domain</code> / <code>seal_selector</code> , <code>ams_domain</code> / <code>ams_selector</code> , <code>ams_verified</code> (Nachrichtensignatur gültig), <code>as_verified</code> (Siegel gültig), <code>cv</code> (gemeldeter Kettenstatus) und <code>aar</code> (vom Weiterleiter festgehaltene Ergebnisse <code>spf</code> / <code>dkim</code> / <code>dmARC</code> samt <code>authserv_id</code>)

| `aar_differences`, `aar_note` | ab 2.2.0: wo eine frühere Station (`station`) etwas anderes festgehalten hat (`station_result`) als Smart-Mail-Flow (`own_result`). **Reine Information, kein Fehler:** Smarthosts und Mail-Gateways prüfen die Einlieferung vom eigenen Mailserver des Absenders (oft ohne DKIM, mit dynamischer IP) und signieren die Mail erst danach |

Das ARC-Ergebnis verändert nie das eigene DMARC-Ergebnis. Auch eine gültige Kette macht aus `dmARC.result = fail` kein `pass`.

4.8 `return_path_dns` – Kann die Absenderadresse Rückmeldungen empfangen?

Prüft die Domain aus `smtp.mail_from` (Unzustellbarkeitsmeldungen gehen dorthin).

- `status`: `pass`, `warn`, `fail`, `temperror`, `not_tested` (Null-Sender).
- `reasons[]`, mögliche Gründe:
 - `nxdomain`: Domain existiert nicht
 - `null_mx`: Domain nimmt ausdrücklich keine Mails an
 - `no_mx_implicit_a`: kein MX, nur A/AAAA vorhanden (`warn`)
 - `no_mx_no_a`: weder MX noch A/AAAA
 - `mx_targets_unresolvable`: MX-Ziele lösen nicht auf
 - `some_mx_targets_unresolvable`: einige MX-Ziele lösen nicht auf
 - `mx_is_cname:<host>`: MX-Ziel ist ein CNAME (`warn`)
 - `cname_loop`, `cname_chain_too_long`: Problem mit der CNAME-Kette
- `cname_chain`, `final_name`, `mx[]` (`pref`, `host`, `is_cname`, `a`, `aaaa`), `implicit_mx`: die gefundenen DNS-Daten.

Diese Prüfung ist unabhängig von SPF: Ein Server kann per SPF erlaubt sein, auch wenn die Domain keine Mails empfangen kann.

4.9 `fcrdns` und `helo` – Identität des sendenden Servers

- `fcrdns` (Forward-Confirmed Reverse DNS): PTR der IP (`ptr`) und ob ein PTR-Name wieder auf dieselbe IP zeigt (`confirmed`). `fail` heißt: kein PTR oder kein Rückverweis. Viele Empfänger lehnen solche Mails ab.
- `helo`, aufgeteilt in Teilprüfungen:
 - `syntax`: `fqdn`, `address_literal`, `invalid` oder `missing`
 - `resolves`: Name löst auf
 - `matches_client_ip`: zeigt auf die einliefernde IP
 - `status`: `pass` (alles passt), `warn` (Name gültig, zeigt aber auf andere IPs oder ist ein Adressliteral), `fail` (fehlt, ungültig, nicht auflösbar)
 - `helo.status = warn` ist bei großen Anbietern (z. B. Microsoft 365) üblich und **kein Fehler** der Konfiguration.

4.10 `ip_protocol` – IPv4/IPv6

- `requested`: `ipv4` bzw. `ipv6`, wenn an `ip4.` / `ip6.` gesendet wurde, sonst `auto`.
- `used`: tatsächlich genutzte Version.
- `status`:
 - `pass`: wie angefordert bzw. `auto`
 - `not_tested`: angeforderte Version konnte nicht genutzt werden. Praktisch nur bei Umwegen über Zwischenserver möglich, denn die `ip4.` / `ip6.`-Adressen sind nur über die jeweilige Version erreichbar.

4.11 dnsbl – Blacklists

Eigene DNS-Abfrage bei jeder Liste:

- Spamhaus ZEN (über den Spamhaus Data Query Service, IPv4 und IPv6)
- SpamCop
- SpamRATS
- Mailspike
- PSBL
- UCEPROTECT Level 1

Feld	Inhalt
<code>listed_count</code> , <code>error_count</code> , <code>no_response_count</code>	Zusammenfassung
<code>lists[].status</code>	<code>listed</code> (gelistet), <code>not_listed</code> (nur bei ausdrücklicher Negativantwort der Liste), <code>no_response</code> (Liste hat nicht geantwortet), <code>error</code> (Fehlerantwort, z. B. Abfrage abgelehnt), <code>not_supported</code> (Liste prüft keine IPv6-Adressen)
<code>codes</code> , <code>meaning</code> , <code>evidence</code>	Antwortcodes der Liste, ihre Bedeutung und ggf. Begründungstext
<code>query</code>	abgefragter DNS-Name. <code>[key]</code> steht für einen nicht veröffentlichten Zugangsschlüssel und ist kein Fehler im Abfragenamen
<code>rcode</code> , <code>resolver</code> , <code>duration_ms</code>	DNS-Antwortcode, abfragender Resolver (<code>system</code> oder <code>list_nameservers</code>) und Dauer

`blacklistmaster` enthält zusätzlich das Ergebnis des externen Dienstes `blacklistmaster.com` (über 100 Listen, eigenes Format). Bei internen Selbsttests steht dort `"status": "skipped"`.

4.12 domain_security – Konfiguration der Absenderdomain

Geprüft wird die Domain aus `header_from`. Das sind **Konfigurationsprüfungen der Domain**, kein Nachweis über diese Mail. MTA-STS und DANE schützen zum Beispiel Mails, die *an* diese Domain gehen.

Feld	Inhalt
<code>mx</code>	MX-Server der Domain
<code>dnssec.status</code>	<code>signed</code> (Antwort DNSSEC-validiert), <code>unsigned</code> , <code>bogus</code> (Signatur fehlerhaft oder DNS-Ausfall), <code>temperror</code>

Feld	Inhalt
mta_sts	pass = TXT _mta-sts und Policy per HTTPS gültig, mode=enforce, alle MX abgedeckt. warn = mode=testing / none. fail = Policy nicht abrufbar, ungültig oder MX nicht abgedeckt (mx_not_covered)
tls_rpt	TXT _smtp._tls mit rua=: pass, fail, not_present
dane	TLSA-Einträge _25._tcp.<mx> je MX. pass = für alle MX vorhanden und DNSSEC-validiert, warn = nur für einige MX, fail = TLSA ohne gültiges DNSSEC (unwirksam)
bimi	TXT default._bimi mit Logo-URL (logo_url, vmc_url): pass, fail, not_present
summary	ab 2.2.0: bewertet werden nur dnssec, mta_sts und tls_rpt (rated, rated_ok von rated_total, status). DANE und BIMi stehen unter informational und führen nie zu Warnung oder Fehler – kleine Firmen haben sie in der Regel nicht

Wichtig: domain_security und dns_analysis beschreiben die Konfiguration der Domain für Mails, die **an** sie gehen. Sie beweisen nicht, dass die geprüfte Übertragung MTA-STS oder DANE genutzt hat. Die tatsächlich ausgehandelte Verschlüsselung steht in smtp.tls.

4.13 smime – S/MIME-Signatur (RFC 8551, ab Version 2.1.0)

Wird eine Mail mit S/MIME signiert (z. B. von einem Verschlüsselungs-Gateway), prüft Smart-Mail-Flow die Signatur und das Zertifikat des Unterzeichners. Fehlt das Feld, wurde der Datensatz vor 2.1.0 geprüft.

Feld	Inhalt
status	not_present (nicht signiert, für die meisten Mails normal), pass (gültig, vertrauenswürdig, passt zur From-Adresse), warn (Signatur gültig, aber Einschränkung, siehe reasons), fail (Inhalt nach dem Signieren verändert), encrypted (S/MIME-verschlüsselt, ohne Schlüssel nicht prüfbar), permerror (Signatur beschädigt oder unvollständig)
type	detached (multipart/signed, Inhalt lesbar), opaque (Inhalt in der Signatur eingebettet), encrypted
nested	Signatur steckt nicht auf oberster Ebene, z. B. weil ein Gateway danach einen Hinweis angehängt hat
micalg	angegebenes Hash-Verfahren (z. B. sha-256)
signature_valid	true = Inhalt unverändert

Feld	Inhalt
chain_trusted	Zertifikat führt zu einer vertrauenswürdigen Zertifizierungsstelle (Zweck E-Mail-Signatur). null = nicht geprüft. Grundlage sind die weit verbreiteten Mozilla-Wurzelzertifikate; einzelne reine S/MIME-Zertifizierungsstellen können fehlen
signer	Zertifikat des Unterzeichners: subject, common_name, emails, issuer, issuer_organization, serial, not_before, not_after, expired, not_yet_valid, days_left, self_signed, key_algorithm, key_bits, signature_hash, email_protection (für E-Mail-Schutz ausgestellt, null = ohne Einschränkung), matches_from (Zertifikat enthält die From-Adresse)
reasons	Gründe für warn / fail, z. B. „Zertifikatskette nicht vertrauenswürdig“, „selbstsigniertes Zertifikat“, „Zertifikat abgelaufen“, „Zertifikat passt nicht zur From-Adresse“, „Zertifikat läuft in N Tagen ab“ (nur Hinweis)

Hinweis für S/MIME-Gateways: Ist im Gateway die Verschlüsselung erzwungen, lehnt es Mails an Smart-Mail-Flow selbst ab („E-Mail could not be encrypted“), weil für Testadressen kein Empfängerzertifikat existiert. Für Tests die Verschlüsselung auf „wenn möglich“ stellen oder eine Ausnahme für smartmailflow.email, ip4.smartmailflow.email und ip6.smartmailflow.email anlegen. Signieren funktioniert ohne Empfängerzertifikat.

4.14 dns_analysis – DNS-Analyse der Absenderdomain (ab 2.2.0)

Geprüft wird die Domain aus header_from, unabhängig davon, welcher Server die Mail geschickt hat.

Feld	Inhalt
mx	MX-Server (records[]: pref, host, a, aaaa) mit STARTTLS-Prüfung je Server: tls.protocol, tls.verified, tls.certificate (issuer_organization, type = DV/OV/EV, not_after, days_left, key_algorithm, key_bits, hostname_match). status: pass (alle mit gültigem Zertifikat), warn (ungültiges Zertifikat oder nicht erreichbar), fail (ohne STARTTLS), none (kein MX). Geprüft werden höchstens 5 MX
spf	record, tree (jeder Bestandteil mit meaning und action, bei a / mx die aufgelösten ips / hosts, bei include / redirect der eingebundene Eintrag als child – beliebig tief), dns_lookups_total (über den ganzen Eintrag, Grenze 10), void_lookups (Grenze 2), includes, authorized_networks (alle erlaubten Adressen/Netze), problems, status (fail bei Überschreitung, Schleife, ungültigen Bestandteilen oder +all; warn ab 8 Abfragen)

Feld	Inhalt
dkim[]	je verwendetem Selektor: Schlüssel (key_algorithm, key_bits, key_strength), key_tags mit Erklärung, signed_at / expires_at aus der Signatur. DKIM-Schlüssel sind kein Zertifikat: kein Aussteller, kein Ablaufdatum
dmARC	record, tags[] (jedes Tag mit meaning), notes (z. B. p=none, fehlendes rua), status (warn bei p=none oder pct < 100)

4.15 mta_checks – Gegenprobe

Ergebnisse der Standardfilter des Mailservers (SPF, DKIM, DMARC) zum Vergleich mit der eigenen Prüfung. In mismatch steht, wo sie abweichen. Erwartet ist eine Liste ohne Einträge. Eine bekannte Ausnahme: Fehlt ein DKIM-Schlüssel, meldet der Filter fail statt permerror.

5. Hinweise zur Bewertung

- Alle Ergebnisse gelten für **diese eine Mail** und den Zeitpunkt checked_at. Andere Server oder Versandwege derselben Domain können anders konfiguriert sein.
- Für die Zustellbarkeit am wichtigsten:
 - dmarc.result = pass
 - dkim.result = pass mit key_strength = ok
 - spf.result = pass
 - fcrdns.status = pass
 - return_path_dns.status = pass
 - keine Einträge in dnsbl
- warn ist ein Hinweis, kein Fehler, besonders bei helo und domain_security.
- Fehlende Felder in alten Datensätzen (software_version fehlt oder 1.x) bedeuten nicht „nicht bestanden“, sondern „damals nicht geprüft“.

Revision #4

Created 15 October 2026 21:20:56 by smartmailflowki

Updated 16 October 2026 07:54:44 by smartmailflowki