Table of Contents

ePDG Monitoring

Integrated VoWiFi Gateway Monitoring System (ePDG)

1. Review of the decision

The VAS Experts ePDG Monitoring system provides full operational control of the **fast-epdg** component, the VoWiFi (Voice over WiFi) gateway operating according to 3GPP TS 29.273 and TS 24.302. The gateway provides secure transmission of voice and packet traffic through untrusted Wi-Fi channels with IPSec / IKEv2 tunneling and integration with the EPC core through SWu, SWm, SWx, S2b, S6b interfaces.

The solution provides a single monitoring platform for the mobile operator's operational services — from the IPSec SA (L3 security) level to the KPI of VoWiFi subscriber experience.

Key advantages

- **Real-time monitoring** — update metrics every 10-15 seconds, directly display the status of IKE SA / Child SA and GTP tunnels in NOC dashboards without delayed aggregation (hereinafter NOC — Network Operation Center, network management center).
- **Proactive detection of anomalies** — 20+ alarms with automatic escalation in importance. PGW/AAA inaccessibility, increased IKEv2 delays, and an increase in EAP-AKA errors are detected before subscribers notice problems with calls.
- **Open integration interfaces** — Prometheus, SNMP v2c, Alertmanager webhooks, Grafana support. Integration into the existing NMS/OSS infrastructure without vendor binding.
- **Minimum external dependencies at the plugin level** — built-in `/metrics` endpoint in fast-epdg, without Java, without JMX, without external agents.
- **Coverage of the entire SWu → S2b** stack — IKEv2 (SWu), Diameter SWm/SWx/S6b, GTPv2-C (S2b) and GTP-U data plane — all in one place. The 33 metrics cover control plane and data plane.

2. Architecture of the monitoring system

<mermaid> flowchart TB

```
subgraph DataPlane["Data Plane"]
    IPSEC["IPSec ESP<br/>IKEv2 SA / Child SA<br/>Kernel xfrm"]
    GTPU["GTP-U Tunneller<br/>S2b Data<br/>ePDG ↔ PGW"]
end
```

```
subgraph ControlPlane["Control Plane"]
    IKE["IKEv2 SWu<br/>EAP-AKA' auth"]
    DIAM["Diameter Client<br/>SWx/SWm/S6b"]
    GTPC["GTPv2-C S2b<br/>to PGW/SMF"]
```

```
    CTRL["ePDG Controller<br/>Attach/Detach FSM"]
end
```

```
subgraph Collection["Metrics Collection"]
    PROMEXP["fast-epdg<br/>/metrics endpoint<br/>:9817"]
end
```

```
subgraph Storage["Storage"]
    PROM["Prometheus<br/>TSDB<br/>15-day retention"]
end
```

```
subgraph Visualization["Visualization"]
    GRAF["Grafana<br/>4 дашборда, 35+ панелей"]
end
```

```
subgraph Alerting["Alerting"]
    AM["Alertmanager<br/>Routing / Inhibition"]
    EMAIL["Email SMTP"]
    SNMPGW["SNMP Trap Sender<br/>Webhook → Trap gateway"]
    NMS["Внешняя NMS<br/>SNMP v2c UDP/162"]
    WH["Webhooks<br/>Telegram / PagerDuty"]
end
```

```
IKE --> PROMEXP
IPSEC --> PROMEXP
GTPC --> PROMEXP
GTPU --> PROMEXP
DIAM --> PROMEXP
CTRL --> PROMEXP
```

```
PROMEXP --> PROM
PROM --> GRAF
PROM --> AM
```

```
AM --> EMAIL
AM --> SNMPGW
SNMPGW --> NMS
AM --> WH
```

</mermaid>

Four-level monitoring architecture

Level	Component	Technology
Collection	Built-in `/metrics` endpoint fast-epdg	Prometheus text format over HTTP
Storage	Prometheus TSDB	Local storage, 15-day storage by default
Visualization	Grafana + JSON support	Autodownload 4 dashboards
Alerting	Alertmanager + SNMP Trap Sender	PromQL rules → webhook → SNMP v2c trap

3. Components and indicators

Monitoring coverage

<mermaid> flowchart LR

```
  EXP["fast-epdg<br/>/metrics :9817"]
```

```
  EXP --> CFG["Config<br/>2 metrics"]
  EXP --> NET["Network<br/>1 metric"]
  EXP --> PROTO["Protocols L5-L7<br/>15 metrics"]
  EXP --> SVC["Service KPI<br/>4 metrics"]
  EXP --> SESS["Session State<br/>4 metrics"]
  EXP --> APP["Application<br/>3 metrics"]
  EXP --> SYS["System<br/>4 metrics"]
```

```
  PROTO --> IKEV2["IKEv2<br/>SWu — 3"]
  PROTO --> GTPC["GTPv2-C<br/>S2b — 4"]
  PROTO --> GTPU["GTP-U<br/>S2b data — 3"]
  PROTO --> DIA["Diameter<br/>SWm/SWx/S6b — 5"]
```

</mermaid>

Quantitative review by category

Category	Number of metrics	Survey interval	Key indicators
Config	2	10 sec	Configuration status, reload counter
Network	1	10 sec	Node connection status (PGW/AAA/HSS)
IKEv2 (SWu)	3	10 sec	Reports by type (IKE_SA_INIT, IKE_AUTH, CREATE_CHILD_SA), delay diagram, errors
GTPv2-C (S2b)	4	10 sec	Messages (Create/Modify/Delete Session), delays, errors, relays
GTP-U data plane	3	10 sec	Packets/bytes, tunneling errors
Diameter (SWm/SWx/S6b)	5	10 sec	Command code messages (DER/DEA, MAR/MAA, AAR/AAA), delays, errors, watchdog, connection status
Service KPI	4	10 sec	Percentage of successful attempts, duration histogram, service availability, uptime
Session State	4	10 sec	IKE SA, Child SA, GTP sessions, all users
Application	3	10 sec	Number of streams, memory, log messages by levels
System	4	10 sec	CPU recycling, memory, memory disposal, open FD
Total	**33 metrics**		

Naming principles

All metrics have the prefix `epdg_` and are organized in a hierarchy:

```
epdg_
├── config_*           # Configuration
├── network_*          # Network layer
├── ikev2_*            # SWu (IKEv2/IPSec)
├── gtp_*              # S2b control-plane GTPv2-C
├── gtpu_*             # S2b data-plane GTP-U
├── diameter_*         # SWm/SWx/S6b
├── service_*          # Service KPIs (attach, availability, uptime)
├── session_*          # Session Status (IKE SA, Child SA, GTP, subscribers)
├── app_*              # App Metrics (memory, threads, logs)
└── system_*           # System metrics (CPU, disk, network)
```

4. List of metrics

All metrics are exported through a single `/metrics` endpoint in Prometheus text format. The name follows the rules of Prometheus: `epdg_<group>_<name>[_unit]`, the Counter type has the suffix `_total`, Histogram is the suffix `_seconds`/`_bytes`.

4.1 Config (2)

Name	Type	Appointment
`epdg_config_status`	Gauge	Component configuration status (0=error, 1=ok)
`epdg_config_reload_total`	Counter	Configuration download counter (success/failure)

4.2 Network (1)

Name	Type	Appointment
`epdg_network_connection_status`	Gauge	TCP/UDP connection status to a node (0=down, 1=up) — applies to PGW (S2b), AAA (SWm), HSS (SWx)

4.3 IKEv2 SWu (3)

Name	Type	Appointment
`epdg_ikev2_messages_total`	Counter	IKEv2 Message Counter (IKE_SA_INIT / IKE_AUTH / CREATE_CHILD_SA / INFORMATIONAL)
`epdg_ikev2_request_duration_seconds`	Histogram	IKEv2 response time
`epdg_ikev2_errors_total`	Counter	IKEv2 errors (NO_PROPOSAL_CHOSEN, AUTHENTICATION_FAILED, INVALID_SYNTAX, etc.)

4.4 GTPv2-C S2b (4)

Name	Type	Appointment
epdg_gtp_messages_total	Counter	GTPv2-C (Create/Modify/Delete Session, Echo)
epdg_gtp_request_duration_seconds	Histogram	Waiting time request → reply
epdg_gtp_errors_total	Counter	GTP-C error by Cause Code
epdg_gtp_retransmissions_total	Counter	Redirecting GTP-C requests

4.5 GTP-U data plane (3)

Name	Type	Appointment
epdg_gtpu_packets_total	Counter	Packages via GTP-U tunnel (uplink/downlink)
epdg_gtpu_bytes_total	Counter	Bytes through GTP-U tunnel
epdg_gtpu_errors_total	Counter	Tunneling errors (TEID mismatch, decap fail)

4.6 Diameter SWm/SWx/S6b (5)

Name	Type	Appointment
epdg_diameter_messages_total	Counter	DER/DEA (SWm), MAR/MAA (SWx), AAR/AAA (S6b), STR/STA
epdg_diameter_request_duration_seconds	Histogram	Waiting time request → reply by Diameter
epdg_diameter_errors_total	Counter	Errors by Experimental-Result-Code
epdg_diameter_watchdog_status	Gauge	DWR/DWA watchdog status to node (0=timeout, 1=ok)
epdg_diameter_connection_status	Gauge	Diameter connection status to node (0=disconnected, 1=connected)

4.7 Service KPI (4)

Name	Type	Appointment
epdg_service_attach_total	Counter	Attempts to connect (success/failure) via APN
epdg_service_attach_duration_seconds	Histogram	Duration of connection (IKE_SA_INIT → session ready)
epdg_service_availability	Gauge	Accessibility flag (0=down, 1=up)
epdg_service_uptime_seconds	Gauge	Service availability time

4.8 Session State (4)

Name	Type	Appointment
epdg_session_ike_sa_total	Gauge	Active IKE SA
epdg_session_child_sa_total	Gauge	Active Child SA (IPSec tunnels)
epdg_session_gtp_sessions_total	Gauge	Active GTP-C sessions on S2b
epdg_session_subscribers_total	Gauge	Unique subscribers (UE connected)

4.9 Application (3)

Name	Type	Appointment
epdg_app_threads_total	Gauge	Total number of work streams

Name	Type	Appointment
epdg_app_memory_bytes	Gauge	Process memory by type
epdg_app_log_messages_total	Counter	Log messages by level (debug/info/warn/error/fatal)

4.10 System (4)

Name	Type	Appointment
epdg_system_cpu_usage_percent	Gauge	Download CPU
epdg_system_memory_bytes	Gauge	System memory
epdg_system_disk_bytes	Gauge	Disk space
epdg_system_open_fds	Gauge	Open file descriptions

Types of metrics (reminder)

Type	Appointment
Counter	Monotonically growing counter (messages, errors, reboots)
Gauge	Current value (active sessions, memory, status)
Histogram	Distribution of values with automatic slices over intervals (duration, lifetime)

5. Integration interfaces

<mermaid> flowchart LR

```
  CORE["VAS Experts<br/>ePDG Monitoring"]
```

```
  CORE --> P["Prometheus<br/>CNCF / OpenMetrics"]
  CORE --> S["SNMP v2c<br/>EPDG-MIB"]
  CORE --> G["Grafana<br/>JSON Provisioning"]
  CORE --> W["Webhooks<br/>ChatOps"]
  CORE --> AM["Alertmanager<br/>Routing"]
```

```
  P --> P1["Cloud-native NMS<br/>Thanos / Cortex / Mimir"]
  S --> S1["Legacy NMS<br/>HP OpenView, NetAct<br/>IBM Tivoli"]
  G --> G1["NOC Wall Displays<br/>Drill-down Analytics"]
  W --> W1["Telegram / Slack<br/>PagerDuty / OpsGenie"]
  AM --> AM1["Smart routing<br/>Severity-based"]
```

</mermaid>

5.1 Prometheus (CNCF Standard)

The native `/metrics` endpoint on port **9817** is built into fast-epdg. The format is standard text format Prometheus v0.0.4 (compatible with OpenMetrics). Aggregation is supported with the central Prometheus operator; remote_write team support for long-term storage in Thanos, Cortex, Grafana Mimir.

5.2 SNMP v2c — EPDG-MIB

47 OID covers the Prometheus metric + **14 trap notifications** (with raise/clear pairs according to RFC 3877 ALARM-MIB). Compatible with HP OpenView, IBM Tivoli NetCool, Nokia NetAct, Huawei U2000.

<mermaid> flowchart TB

```
  IANA["IANA PEN<br/>enterprises<br/>.1.3.6.1.4.1"]
  VAS["VAS Experts<br/>.1.3.6.1.4.1.43823<br/>(vas.expert)"]
  EPDG["EPDG-MIB<br/>.43823.1"]
  EPC["EPC Monitoring<br/>.43823.100"]
```

```
  IANA --> VAS
  VAS --> EPDG
  VAS --> EPC
```

```
  EPDG --> OBJ["epdgObjects<br/>.43823.1.1"]
  EPDG --> NOTIF["epdgNotifications<br/>.43823.1.2<br/>14 trap types"]
  EPDG --> CONF["epdgConformance<br/>.43823.1.3"]
```

```
  OBJ --> SERVICE["service .1.1.1<br/>4 OID"]
  OBJ --> IKE["ikev2 .1.1.2<br/>6 OID"]
  OBJ --> GTP["gtp .1.1.3<br/>8 OID"]
  OBJ --> DIAM["diameter .1.1.4<br/>7 OID"]
  OBJ --> SESS["sessions .1.1.5<br/>8 OID"]
  OBJ --> SYS["system .1.1.6<br/>8 OID"]
  OBJ --> NET["network .1.1.7<br/>6 OID"]
```

```
  NOTIF --> TRAPAGR["7 raise / 7 clear<br/>pairs"]
```

</mermaid>

Examples of SNMP requests:

```
# The entire ePDG tree
snmpwalk -v2c -c public <host>.1.3.6.1.4.1.43823.1

# Service availability (Gauge 0..1)
snmpget -v2c -c public <host> .1.3.6.1.4.1.43823.1.1.0
```

5.3 Grafana

4 JSON dashboard support (35+ panels total):

- **ePDG Overview** — availability, KPI connections, sessions, state of interfaces
- **IKEv2 Details** — Messages, Performance, Errors, IKE SA Lifecycle
- **GTP Details** — GTPv2-C + GTP-U data on PGW nodes
- **Diameter Details** — Application messages, delays, watchdog

Automatic installation through an API that supports Grafana. Adaptive design for Network Control Center (NOC) status monitors with auto-update every 15 seconds.

5.4 Alertmanager Webhooks

Webhook interface for integration with any notification system: Telegram Bot, Slack, PagerDuty Events API v2, OpsGenie, Microsoft Teams. A separate **SNMP Trap Sender** service converts Alertmanager webhooks to SNMP v2c traps with Enterprise OID.

6. The alarm system

Alarm categories

Criticism	Alarma	Description	Reaction
Critical	ePDG_Service_Down, ePDG_High_Attach_Failure_Rate, ePDG_PGW_Unreachable, ePDG_AAA_Unreachable, ePDG_Diameter_Watchdog_Timeout	Component is unavailable, widespread connection failures, nodes are unavailable	Immediate escalation: Email + SNMP Trap + Webhook. Repeat every hour
Warning	ePDG_High_IKEv2_Latency, ePDG_High_GTP_Latency, ePDG_High_IKEv2_Error_Rate, ePDG_High_GTP_Error_Rate, ePDG_High_Memory_Usage, ePDG_High_CPU_Usage, ePDG_Low_Disk_Space, ePDG_High_Error_Log_Rate	Performance degradation, resource anomalies	Email. Resend every 4 hours. Suppressed if a “Critical” status is present on the same component

Complete list of alarms (20+ rules)

<mermaid> flowchart LR

```
  AL["ePDG Alert Rules<br/>20+"]
```

```
  AL --> CR["Critical<br/>5 rules"]
  AL --> WR["Warning<br/>8 rules"]
  AL --> INFO["Recording<br/>34 rules"]
```

```
  CR --> C1["Service_Down<br/>availability == 0"]
  CR --> C2["Attach_Failure_Rate<br/>> 10%"]
  CR --> C3["PGW_Unreachable<br/>connection_status{s2b} == 0"]
  CR --> C4["AAA_Unreachable<br/>connection_status{swm} == 0"]
  CR --> C5["Diameter_Watchdog_Timeout<br/>watchdog_status == 0"]
```

```
  WR --> W1["High_IKEv2_Latency<br/>p95 > 1.0 s"]
  WR --> W2["High_GTP_Latency<br/>p95 > 0.5 s"]
  WR --> W3["High_IKEv2_Error_Rate<br/>> 5%"]
  WR --> W4["High_GTP_Error_Rate<br/>> 5%"]
```

```
  WR --> W5["High_Memory_Usage<br/>> 80%"]
  WR --> W6["High_CPU_Usage<br/>> 80%"]
  WR --> W7["Low_Disk_Space<br/>< 10%"]
  WR --> W8["High_Error_Log_Rate<br/>> 10/s"]
```

```
  INFO --> I1["attach_success_rate<br/>preaggregated"]
  INFO --> I2["p95_p99_latency<br/>preaggregated"]
  INFO --> I3["throughput<br/>preaggregated"]
```

</mermaid>

Alarm treatment process

<mermaid> sequenceDiagram

```
  participant M as Метрика (Prometheus)
  participant R as Alert Rule (PromQL)
  participant AM as Alertmanager
  participant E as Email (SMTP)
  participant SG as SNMP Trap Gateway
  participant NMS as Внешняя NMS
  participant W as Webhook (ChatOps)
```

```
  M->>R: The value exceeds the threshold
  R->>R: Waiting (for: 1-10 мин)
  R->>AM: Alert FIRING
  AM->>AM: Group by [alertname, component]
  AM->>AM: Inhibition check (critical overrides warning)
```

```
  alt severity = critical
      AM->>E: Email [CRITICAL]
      AM->>SG: Webhook → SNMP Trap
      SG->>NMS: SNMP v2c Trap (OID .1.3.6.1.4.1.43823.1.2.X)
      AM->>W: Webhook (Telegram / PagerDuty)
  else severity = warning
      AM->>E: Email [WARNING]
  end
```

```
  Note over M,R: The metric is returning to normal
  R->>AM: Alert RESOLVED
  R->>SG: clear-trap (paired notification)
  AM->>E: Email [RESOLVED]
```

</mermaid>

Features

- **Inhibition**: Critical alarms automatically suppress Warning for the same component

- **Grouping**: Alarms are grouped into `alertname` + `component` with a 30-second window
- **Dead time / Hysteresis**: 1 to 10 minutes `for` prevents false positives
- **Trap pairing**: raise/clear simultaneous events for compliance with RFC 3877 ALARM-MIB

7. Visualization and operational dashboards

Composition of dashboards

Dashboard	Panel	Purpose
ePDG Overview	10	Service availability, connection success rate, number of active sessions, SWu/SWm/S2b status, interface bandwidth
IKEv2 Details	10	Mes per second by type, histogram of request duration, delay in the 95th percentile, error by type, IKE SA life cycle
GTP Details	8	GTPv2-C PGW messages, retransmissions, cause code errors, GTP-U (uplink/downlink) carriers
Diameter Details	7	Number of application messages (SWm/SWx/S6b), duration of requests, state of watchdog timer, distribution of result codes, chronology of connection states

Design for Network Management Center (NOC)

<mermaid> flowchart TB

```
  NOC["NOC Dashboard Layer"]
```

```
  NOC --> OVER["ePDG Overview<br/>KPI Summary"]
  NOC --> IKE["IKEv2 Details<br/>Drill-down"]
  NOC --> GTP["GTP Details<br/>Drill-down"]
  NOC --> DIA["Diameter Details<br/>Drill-down"]
```

```
  OVER -->|Click attach KPI| IKE
  OVER -->|Click session count| GTP
  OVER -->|Click peer status| DIA
```

</mermaid>

- **Auto Update**: 15-second update period
- **Adaptive color scheme**: green → yellow → red by threshold values
- **Drill-down**: from Overview to Detail to Component
- **Time-range selector**: 5 minutes to 30 days of history
- **JSON provisioning**: dashboards are automatically deployed

8. Integration into a single EPC Monitoring stack

ePDG monitoring is fully integrated into overall packet core monitoring:

<mermaid> flowchart TB

```
  subgraph Common["Unified Monitoring Stack"]
      PROM["Prometheus"]
      GRAF["Grafana"]
      AM["Alertmanager"]
  end
```

```
  subgraph Sources["Sources of EPC metrics"]
      DPI["FastDPI<br/>:9110"]
      SMF["SMF /metrics<br/>:9090"]
      PCEF["fast-pcef /metrics<br/>:9090"]
      PCRF["FastPCRF"]
      EPDG["fast-epdg<br/>:9817"]
  end
```

```
  DPI --> PROM
  SMF --> PROM
  PCEF --> PROM
  PCRF --> PROM
  EPDG --> PROM
```

```
  PROM --> GRAF
  PROM --> AM
```

</mermaid>

The NOC operator sees **all EPC components** (DPI, SMF, PCEF, FastPCRF, ePDG) in a single Grafana interface, with a single alarm system and notification routing through one Alertmanager.

9. Coverage of metrics by OSI levels

<mermaid> graph LR

```
  L1["L1 Physical<br/>NIC counters via system"]
  L2["L2 Data Link<br/>MAC, VLAN"]
  L3["L3 Network<br/>IP, IPSec ESP, GTP-U"]
  L4["L4 Transport<br/>TCP/UDP/SCTP"]
  L5["L5 Session<br/>GTPv2-C, IKEv2"]
  L6["L6 Presentation<br/>IKEv2/IPSec encryption, EAP-AKA'"]
  L7["L7 Application<br/>Diameter, service bearer ops"]
  Operations["Operations<br/>KPI, SLA, Capacity"]
  CX["CX Level<br/>Subscriber Experience"]
```

```
  L1 --> L2 --> L3 --> L4 --> L5 --> L6 --> L7 --> Operations --> CX
```

```
  style L1 fill:#e74c3c,color:#fff
  style L2 fill:#e67e22,color:#fff
  style L3 fill:#f39c12,color:#fff
  style L4 fill:#2ecc71,color:#fff
```

```
  style L5 fill:#1abc9c,color:#fff
  style L6 fill:#3498db,color:#fff
  style L7 fill:#9b59b6,color:#fff
  style Operations fill:#34495e,color:#fff
  style CX fill:#2c3e50,color:#fff
```

</mermaid>

Detailing metrics by level

OSI model:

Level	Metrics	Examples
L1/L2 Physical / Data Link	-	Covered by a separate node_exporter or equivalent at the OS level (not included in the ePDG metrics list)
L3 Network / IPSec tunnels	3	`epdg_gtpu_packets_total`, `epdg_gtpu_bytes_total`, `epdg_gtpu_errors_total` — GTP-U data plane
L4 Transport	1	`epdg_network_connection_status` — TCP connections to nodes (PGW/AAA/HSS)
L5 Session	3	`epdg_session_ike_sa_total`, `epdg_session_child_sa_total`, `epdg_session_gtp_sessions_total`
L6 Presentation/Security	3	`epdg_ikev2_messages_total`, `epdg_ikev2_request_duration_seconds`, `epdg_ikev2_errors_total` — IKEv2/IPSec encryption and EAP-AKA authentication
L7 Application	9	`epdg_diameter_*` (SWm/SWx/S6b, 5 metrics), `epdg_gtp_*` (GTPv2-C, 4 metrics)

Operator level:

Level	Metrics	Examples
Operations	11	`epdg_service_availability`, `epdg_service_uptime_seconds`, `epdg_app_*` (3), `epdg_system_*` (4), `epdg_config_*` (2)
Customer Experience	3	`epdg_service_attach_duration_seconds` p95, `epdg_service_attach_total` (success rate), `epdg_ikev2_request_duration_seconds` p99

Level 9: Quality of VoWiFi service perception

QoE indicator	Source metrics	Interpretation
VoWiFi connection time	`epdg_service_attach_duration_seconds` p95	> 3 seconds — subscriber notices delay when switching to WiFi
Continuity of service	`epdg_session_ike_sa_total` delta	Mass discharge > 50 IKE SA = accessibility issue
Authentication success	`ePDG_High_Attach_Failure_Rate` alert rate	> 5% = HSS/AAA node problem
Delayed appointment bearer	`epdg_gtp_request_duration_seconds{msg=create-session}` p99	> 500 ms — delayed availability of voice channel

QoE indicator	Source metrics	Interpretation
GTP-U tunnel	`epdg_gtpu_errors_total` rate / `epdg_gtpu_packets_total`	> 0.1% = degradation of voice quality
IKEv2-reliability	`epdg_ikev2_errors_total` by type	NO_PROPOSAL_CHOSEN / AUTHENTICATION_FAILED — problems with certs / UE

10. Standards and compatibility

Standard	Area	Application
3GPP TS 29.273	SWx/S6b/SWm	Methodology for accounting for Diameter messages and resulting codes
3GPP TS 24.302	SWu (IKEv2)	Definition of IKEv2 message types and error codes
3GPP TS 33.402	3GPP security for non-3GPP access	EAP-AKA'/IKEv2 security parameters
3GPP TS 23.402	Non-3GPP access architecture	Interface Structure (SWu/SWm/SWx/S6b/S2b)
3GPP TS 32.421	Performance Measurement	Collection methodology KPI
3GPP TS 32.409	Performance measurement charging	Counter structure
IETF RFC 7296	IKEv2	Message types, error notifications, state SA
IETF RFC 6733	Diameter	Command codes, Result-Codes
IETF RFC 4187	EAP-AKA	Authentication via SIM
IETF RFC 3877	ALARM MIB	Enterprise MIB structure for alarms
IETF RFC 3418	SNMPv2 MIB	SNMP v2c compatibility
Prometheus Exposition Format	Metrics (v0.0.4)	Export metric format
OpenMetrics	CNCF Standard	Prospective compatibility

11. The deployment model

<mermaid> flowchart TB

```
  subgraph Host1["ePDG Server"]
      EPDG["fast-epdg<br/>(VoWiFi gateway)"]
      PLUGIN["/metrics endpoint<br/>:9817"]
      EPDG -.-> PLUGIN
  end
```

```
  subgraph Host2["Monitoring server"]
      PROM["Prometheus"]
      GRAF["Grafana"]
      AM["Alertmanager"]
      SNMPTRAP["SNMP Trap Sender<br/>(webhook gateway)"]
      PROM --> GRAF
      PROM --> AM
      AM --> SNMPTRAP
```

```
  end
```

```
  subgraph Host3["External systems"]
      NMS["Операторская NMS<br/>(HP OpenView /<br/>NetAct / Tivoli)"]
      CHAT["ChatOps<br/>(Telegram / PagerDuty)"]
  end
```

```
  PLUGIN -->|HTTP :9817/metrics| PROM
  SNMPTRAP -->|UDP 162| NMS
  AM -->|Webhook| CHAT
```

</mermaid>

Deployment characteristics

Parameter	Value
Metrics footprint	Integrated (~2 MB memory overhead)
External dependencies	The self-contained `fast-epdg` package (rpm)
Management	`fast-epdg.service` systemd
Configuration	The `monitoring` section in `fast-epdg.conf`
Update	Updating the configuration without interrupting operations
OS	Linux (RHEL/CentOS 8+, Ubuntu 22.04+)
Port	9817 TCP (listening on 0.0.0.0, configurable)
Deployment time	< 5 minutes (enable the plugin in the config file + restart)

Accommodation options

- **On-premise** — the plugin runs in the fast-epdg address space, zero resource consumption
- **Co-located Prometheus** — Prometheus collects metrics from an application running on the same host
- **Centralized** — a single Prometheus collects from all ePDG nodes

12. Metric exporter configuration

The `monitoring` section in `fast-epdg.conf`:

```
monitoring {
    enabled = yes
    listen_port = 9817
    listen_address = 0.0.0.0
    update_interval = 10
    metrics {
        ikev2 = yes
        gtp = yes
        diameter = yes
        service = yes
        session = yes
```

```
        app = yes
        system = yes
    }
}
```

Each group of metrics can be independently turned on/off without recompilation.