dog-agent — Manuale

L'agent unico della SentinelSuite

Dognet Technologies S.r.l.

© Dognet Technologies S.r.l. — P.IVA 04867480164

Table of contents

1. Introduzione 3
1.1 Cos'è dog-agent 3
2. 1. Installazione e file 4
3. 2. Avvio e riga di comando 5
4. 3. Connessione sicura al master (TLS) 6
5. 4. Riferimento della configurazione 7
6. 5. Pairing con i master 9
7. 6. Rilevamento minacce e blocco automatico (FireDog) 10
8. 7. Comandi che l'agent accetta dai master 11
9. 8. Log e diagnostica 12
10. 9. Domande frequenti e problemi comuni 13

© Dognet Technologies S.r.l. — P.IVA 04867480164

1. Introduzione

Manuale di riferimento per **dog-agent 1.3.1**.

1.1 Cos'è dog-agent

Un unico binario Rust che si collega via WebSocket a uno o più master della SentinelSuite: **FireDog** (firewall), **CyberShepherd** (monitoraggio, compliance, hardening) e **SentinelCore** (predisposto, non ancora attivo). Ogni master è un blocco `[[targets]]` nel file di configurazione e gira in modo indipendente: se uno non risponde, gli altri continuano a lavorare.

L'agent gira come **root**: serve per iptables, per leggere `/proc` e i log di sistema e per eseguire hardening e audit richiesti da CyberShepherd.

© Dognet Technologies S.r.l. — P.IVA 04867480164

2. 1. Installazione e file

Pacchetti (vedi il README del repository per come costruirli):

Pacchetto	Quando usarlo
`.deb` musl statico (`make deb-musl`)	consigliato: gira su qualsiasi Debian/Ubuntu
`.deb` glibc (`make deb`)	solo se la glibc del target è ≥ a quella della macchina di build
`.rpm` (`make rpm`)	openSUSE / RHEL / Fedora, sempre musl statico

```
sudo apt install ./dog-agent_<versione>_amd64.deb     # o dpkg -i
sudo nano /etc/dog-agent/agent.conf                    # vedi capitolo 4
sudo systemctl enable --now dog-agent
```

L'installazione **non** sovrascrive un `agent.conf` esistente e **non** riavvia il servizio: dopo un aggiornamento serve `sudo systemctl restart dog-agent` .

Percorso	Contenuto
`/usr/bin/dog-agent`	binario
`/etc/dog-agent/agent.conf`	configurazione (contiene le api_key: permessi **0600** o 0640)
`/etc/dog-agent/agent.conf.example`	esempio commentato
`/etc/sentinelsuite/dog_agent/`	CA dei master per `ca_cert` (convenzione della suite)
`/var/lib/dog-agent/laurel-<target>.json`	punto di lettura del log di Laurel, per non rispedire eventi a ogni riavvio
`/usr/lib/systemd/system/dog-agent.service`	unità systemd (`ProtectSystem=strict`)
`/run/xtables.lock`	creato al boot da `tmpfiles.d` , serve a iptables dentro la sandbox
`/etc/firewall/firedog.conf`	configurazione di firewall-manager: l'agent ci scrive le opzioni PCAP inviate da FireDog

© Dognet Technologies S.r.l. — P.IVA 04867480164

3. 2. Avvio e riga di comando

```
dog-agent [--config <file>] [--log-level <livello>]
dog-agent --version
```

Opzione	Default	Note
`-c`, `--config`	`/etc/dog-agent/agent.conf`	
`-l`, `--log-level`	`log_level` del file	`debug`, `info`, `warn`, `error`; ha la precedenza sul file

All'avvio l'agent controlla la configurazione e **si ferma** se è incompleta (vedi capitolo 9). Se `agent.conf` è leggibile da gruppo o altri scrive un avviso: contiene le api_key in chiaro.

© Dognet Technologies S.r.l. — P.IVA 04867480164

4. 3. Connessione sicura al master (TLS)

Con `wss://` l'agent **verifica il certificato del master**. Dalla 1.3 non esiste più l'accettazione silenziosa dei certificati self-signed: senza una CA configurata l'agent rifiuta di avviarsi su un target `wss://`.

4.0.1 Come funziona

Ogni master (CyberShepherd dalla 0.37, FireDog dalla 1.5.1) ha la **propria CA locale**, creata all'installazione, che firma il suo certificato server. L'agent si fida della CA, non del singolo certificato: il master rinnova il certificato server da solo e l'agent non va toccato.

4.0.2 Configurazione

1. Scarica la CA da ogni master con cui l'agent parla:

```
curl -k -o /tmp/firedog-ca.pem https://<master-firedog>/api/tls/ca.pem
curl -k -o /tmp/cybershepherd-ca.pem https://<master-cybershepherd>/api/tls/ca.pem
```

   `-k` serve solo per questo primo download: subito dopo verifichi tu.

2. **Confronta l'impronta** con quella mostrata nel master (FireDog: Impostazioni > API Keys Agent > "Certificato del master"; CyberShepherd: Impostazioni > Agent pairing e API key). Devono coincidere carattere per carattere:

```
openssl x509 -in /tmp/firedog-ca.pem -noout -fingerprint -sha256
```

3. Metti **tutte** le CA in un solo file (l'agent le carica tutte):

```
sudo install -d -m 0755 /etc/sentinelsuite/dog_agent
cat /tmp/firedog-ca.pem /tmp/cybershepherd-ca.pem | sudo tee /etc/sentinelsuite/dog_agent/ca-bundle.pem >/dev/null
sudo chmod 0644 /etc/sentinelsuite/dog_agent/ca-bundle.pem
```

4. In `agent.conf`, sezione `[agent]`:

```
ca_cert = "/etc/sentinelsuite/dog_agent/ca-bundle.pem"
```

5. `sudo systemctl restart dog-agent`. Nel log deve comparire `TLS: i certificati server sono verificati contro la CA …`.

   `url` deve usare **lo stesso IP o nome** per cui il master ha emesso il certificato (lo vedi nella stessa scheda del master): un IP diverso, anche della stessa macchina, viene rifiutato.

   Una PKI aziendale propria va bene allo stesso modo: in `ca_cert` va la CA che ha firmato il certificato del master.

4.0.3 `allow_insecure_tls` e `ws://`

- `allow_insecure_tls = true` accetta **qualsiasi** certificato senza verificarlo: protegge solo da chi ascolta passivamente, non da chi si mette in mezzo. Solo in una LAN fidata e come misura temporanea.
- `ws://` (senza TLS) resta possibile verso master installati senza HTTPS: api_key e dati viaggiano in chiaro e l'agent lo segnala nel log.

© Dognet Technologies S.r.l. — P.IVA 04867480164

5. 4. Riferimento della configurazione

Formato TOML. Una sezione `[agent]` e uno o più blocchi `[[targets]]`. L'esempio completo e commentato è `config.example.toml`.

5.0.1 `[agent]`

Opzione	Default	Descrizione
`log_level`	`"info"`	`debug`, `info`, `warn`, `error`
`ca_cert`	nessuno	file PEM con le CA dei master; **obbligatorio** se un target usa `wss://` (vedi capitolo 3)
`allow_insecure_tls`	`false`	accetta certificati non verificati; sconsigliato

5.0.2 `[[targets]]` — opzioni comuni

Opzione	Obbligatoria	Default	Descrizione
`name`	sì	—	nome libero, compare nei log
`system_type`	sì	—	`firedog`, `cybershepherd` (accetta anche il vecchio `cybersheppard`), `sentinelcore`
`url`	sì	—	`wss://<master>` consigliato; `https://` e `http://` sono convertiti in `wss://` / `ws://`. Il percorso lo aggiunge l'agent (`/ws/agent/` per FireDog e SentinelCore, `/api/agents/ws` per CyberShepherd)
`api_key`	sì	—	chiave agent generata dal master
`ip`, `hostname`, `mac`	dipende	—	identità della macchina usata dal pairing (capitolo 5)
`environment`	no	`production`	`production`, `testing`, `preproduction`. Solo informativo: non viene inviato ai master e non cambia il comportamento
`reconnect`	no	vedi sotto	riconnessione a due livelli

`reconnect = { fast_backoff_secs = 5, fast_max_attempts = 20, slow_backoff_secs = 1800 }`: dopo una disconnessione l'agent riprova ogni `fast_backoff_secs` secondi per `fast_max_attempts` volte, poi ogni `slow_backoff_secs` (30 minuti). Una sessione rimasta su almeno 15 secondi fa ripartire dal livello veloce.

© Dognet Technologies S.r.l. — P.IVA 04867480164

5.0.3 Opzioni FireDog (`system_type = "firedog"`)

Opzione	Default	Descrizione
`ip` , `hostname` , `mac`	—	**obbligatori**; devono coincidere con il target creato su FireDog
`heartbeat_interval`	`30`	secondi tra un heartbeat (CPU, RAM, disco, rete) e l'altro
`auto_block_threats`	`true`	blocca da solo gli IP con score ≥ `threat_threshold`
`threat_threshold`	`75`	score (0-100) da cui scatta il blocco automatico
`bruteforce_min_attempts`	`10`	login SSH falliti (in 10 minuti) da cui si segnala un brute-force
`bruteforce_score_per_attempt`	`7`	punti per tentativo: score = tentativi × punti, massimo 100
`synflood_min_connections`	`35`	connessioni in stato SYN_RECV dallo stesso IP da cui si segnala un SYN flood
`synflood_score_per_conn`	`2`	punti per connessione: score = connessioni × punti, massimo 100
`integrity_paths`	`[]`	percorsi che il comando di integrità di FireDog può controllare: il master non può chiederne altri

I sei valori del threat-score si possono anche inviare da FireDog (pagina di dettaglio del target): l'agent riscrive queste righe in `agent.conf` , conservando commenti e formattazione, e si riavvia dopo 2 secondi. I valori in Impostazioni > Monitoraggio di FireDog sono solo i **default proposti** e non arrivano agli agent finché non li invii da un target.

Esempio con i default: 10 login falliti = score 70 (segnalato, non bloccato), 11 = 77 (bloccato con soglia 75).

5.0.4 Opzioni CyberShepherd (`system_type = "cybershepherd"`)

Opzione	Default	Descrizione
`ip` , `hostname` , `mac`	—	pairing per identità (consigliato)
`target_id`	—	ID del target sul master; serve solo se mancano `ip` + `hostname` + `mac`
`collection_interval`	`30`	secondi tra una raccolta di metriche e l'altra
`send_interval`	`10`	secondi tra un invio e l'altro
`compression_level`	`3`	compressione Zstd, da 1 (veloce) a 22 (massima)
`max_buffer_size`	`10`	campioni in coda prima di un invio forzato
`collectors`	tutti `true`	`{ system, network, users, files, services, auditd, docker }`
`laurel_log_path`	nessuno	log JSON di Laurel (es. `/var/log/laurel/audit.log`); se impostato l'agent inoltra gli eventi di sicurezza a CyberShepherd

Il punto di lettura di Laurel è salvato in `/var/lib/dog-agent/` (file e posizione): dopo un riavvio dell'agent o del master l'inoltro riprende da lì, senza rispedire eventi già inviati.

© Dognet Technologies S.r.l. — P.IVA 04867480164

6. 5. Pairing con i master

6.0.1 FireDog

1. Su FireDog crea il target con **esattamente** gli stessi IP, hostname e MAC che metterai in `agent.conf` (il master confronta SHA512(ip + hostname + mac)).
2. Genera una chiave in Impostazioni > API Keys Agent e mettila in `api_key`.
3. Configura `ca_cert` (capitolo 3) e `url = "wss://<ip-del-master>"`.
4. Avvia la sessione di pairing dal master, poi `sudo systemctl restart dog-agent`.

   Nel log: `Fase 1 verificata (API key OK)`, `Fase 2 verificata (identity hash OK)`, `Pairing completato`.

6.0.2 CyberShepherd

Con `ip`, `hostname` e `mac` presenti il master riconosce il target dalla sua identità, senza `target_id`. La chiave si genera nella sezione Agent pairing del master.

© Dognet Technologies S.r.l. — P.IVA 04867480164

7. 6. Rilevamento minacce e blocco automatico (FireDog)

Ogni 60 secondi l'agent controlla:

- **Brute-force SSH.** Legge i login falliti degli ultimi **10 minuti** dal journal di systemd (messaggi di `sshd` e `sshd-session`: da OpenSSH 9.8 i login li gestisce `sshd-session`). Se il journal non c'è o è vuoto, usa `/var/log/auth.log` o `/var/log/secure`, filtrati per età. Ogni password sbagliata conta uno; un tentativo verso un utente inesistente conta uno (sshd scrive due righe per lo stesso tentativo, l'agent non le somma).
- **SYN flood.** Conta le connessioni in stato SYN_RECV per IP sorgente in `/proc/net/tcp` e `/proc/net/tcp6`.

Le minacce vengono inviate a FireDog (pagina Minacce) e, se lo score supera `threat_threshold` con `auto_block_threats = true`, l'IP viene bloccato con una regola DROP **in cima** alla catena INPUT, valida anche per SSH e per le connessioni già aperte.

Non vengono mai bloccati automaticamente: loopback, indirizzi non validi o multicast, l'IP del master ricavato da `url`, l'`ip` della macchina stessa. Attenzione: se il master ha più indirizzi, è protetto solo quello di `url`.

I blocchi e le regole applicati dall'agent sono regole iptables in memoria: l'agent non li salva su disco.

© Dognet Technologies S.r.l. — P.IVA 04867480164

8. 7. Comandi che l'agent accetta dai master

FireDog

Comando	Effetto
`add_rule`	aggiunge una regola iptables/ip6tables; su un target con firewall-manager va prima dello scarto finale della catena
`remove_rule`	rimuove una regola
`sync_rules`	rigenera lo stato completo con `firewall-manager --export-json` e lo rimanda al master (non cancella regole)
`block_ip` / `unblock_ip`	blocca o sblocca un IP (INPUT, OUTPUT o entrambi), in cima alla catena
`update_config`	riscrive il threat-score in `agent.conf` e riavvia l'agent
`update_pcap_config`	scrive le opzioni PCAP in `/etc/firewall/firedog.conf` (nessun riavvio)
`check_integrity`	controlla l'integrità dei file; il master può solo restringere l'elenco, mai uscire da `integrity_paths`

CyberShepherd

Comando	Effetto
`execute_hardening`	applica (o simula) un template di hardening
`security_audit_scan`	audit di sicurezza in sola lettura
`apply_remediation`	applica una singola remediation curata per un controllo
`ping`, `get_version`	diagnostica

© Dognet Technologies S.r.l. — P.IVA 04867480164

9. 8. Log e diagnostica

```
sudo journalctl -u dog-agent -f                  # in tempo reale
sudo journalctl -u dog-agent --since "-1 hour"
sudo dog-agent --config /etc/dog-agent/agent.conf --log-level debug   # a mano, servizio fermo
```

Messaggi utili:

Messaggio	Significato
`TLS: i certificati server sono verificati contro la CA ...`	`ca_cert` caricato
`Pairing completato`	collegato a FireDog
`Retry tra 1800s`	il master non risponde da tempo: livello lento (capitolo 4)
`endpoint in chiaro (ws://)`	target senza TLS
`Il file di config ... ha permessi ...`	`agent.conf` leggibile da altri utenti

© Dognet Technologies S.r.l. — P.IVA 04867480164

10. 9. Domande frequenti e problemi comuni

L'agent non parte: `Error: Target '...': endpoint wss:// senza [agent] ca_cert configurato`. Dalla 1.3 serve la CA del master: vedi capitolo 3.

`TLS: CA ... non caricabile (...): No such file or directory`. Il percorso in `ca_cert` è sbagliato o il file non esiste.

`invalid peer certificate: Other(OtherError(CaUsedAsEndEntity))`. Il master usa ancora il vecchio certificato self-signed, che l'agent non accetta come certificato server. Aggiorna il master (FireDog 1.5.1, CyberShepherd 0.37), che crea la CA locale, poi metti in `ca_cert` la nuova CA. Su FireDog installato da clone git la migrazione è manuale (vedi `INSTALL.md` di FireDog).

`invalid peer certificate: UnknownIssuer`. Il certificato del master non è firmato da nessuna delle CA in `ca_cert`: manca la CA di quel master, oppure il master ha creato una CA nuova (`renew --force-ca`). Riscaricala e verifica l'impronta.

`invalid peer certificate: NotValidForName`. `url` usa un IP o un nome diverso da quello del certificato del master. Usa lo stesso indirizzo, oppure, se il master ha cambiato IP, rifai il suo certificato (`sudo firedog-tls ensure --name <ip>` / `sudo cybershepherd-tls ensure --name <ip>`).

Il pairing con FireDog fallisce alla fase 2 (identity hash). IP, hostname o MAC in `agent.conf` non coincidono con il target creato su FireDog (attenzione a maiuscole e separatori del MAC), oppure non è attiva una sessione di pairing sul master.

Dopo aver cambiato `agent.conf` non succede niente. L'agent legge il file solo all'avvio: `sudo systemctl restart dog-agent`.

L'agent è fermo su "Retry tra 1800s". Ha esaurito i tentativi veloci. Quando il master torna raggiungibile si ricollega al prossimo tentativo; per non aspettare, riavvia l'agent.

Il brute-force non viene rilevato. Controlla che i login falliti compaiano in `journalctl SYSLOG_IDENTIFIER=sshd-session` (o `sshd`) oppure in `/var/log/auth.log` / `/var/log/secure`, e ricorda che contano solo gli ultimi 10 minuti e servono almeno `bruteforce_min_attempts` tentativi.

Ho inviato il threat-score da FireDog ma l'agent usa ancora i vecchi valori. Controlla in `agent.conf` che le righe siano cambiate e nel log che l'agent si sia riavviato. Ricorda che i valori in Impostazioni > Monitoraggio sono solo i default proposti: si inviano dalla pagina del singolo target.

Il servizio non parte con `226/NAMESPACE`. Manca `/run/xtables.lock`: `sudo systemd-tmpfiles --create dog-agent.conf`. Se esiste una copia manuale dell'unità in `/etc/systemd/system/`, quella ha la precedenza sul pacchetto: rimuovila e fai `systemctl daemon-reload`.

© Dognet Technologies S.r.l. — P.IVA 04867480164