

ANNEXE - Contexte

TechServices SARL

- **Secteur** : Services informatiques et maintenance
- **Effectif** : 15 employés
- **Locaux** : 350 m² sur 1 étage
- **Activité** : Maintenance informatique, développement web, formation

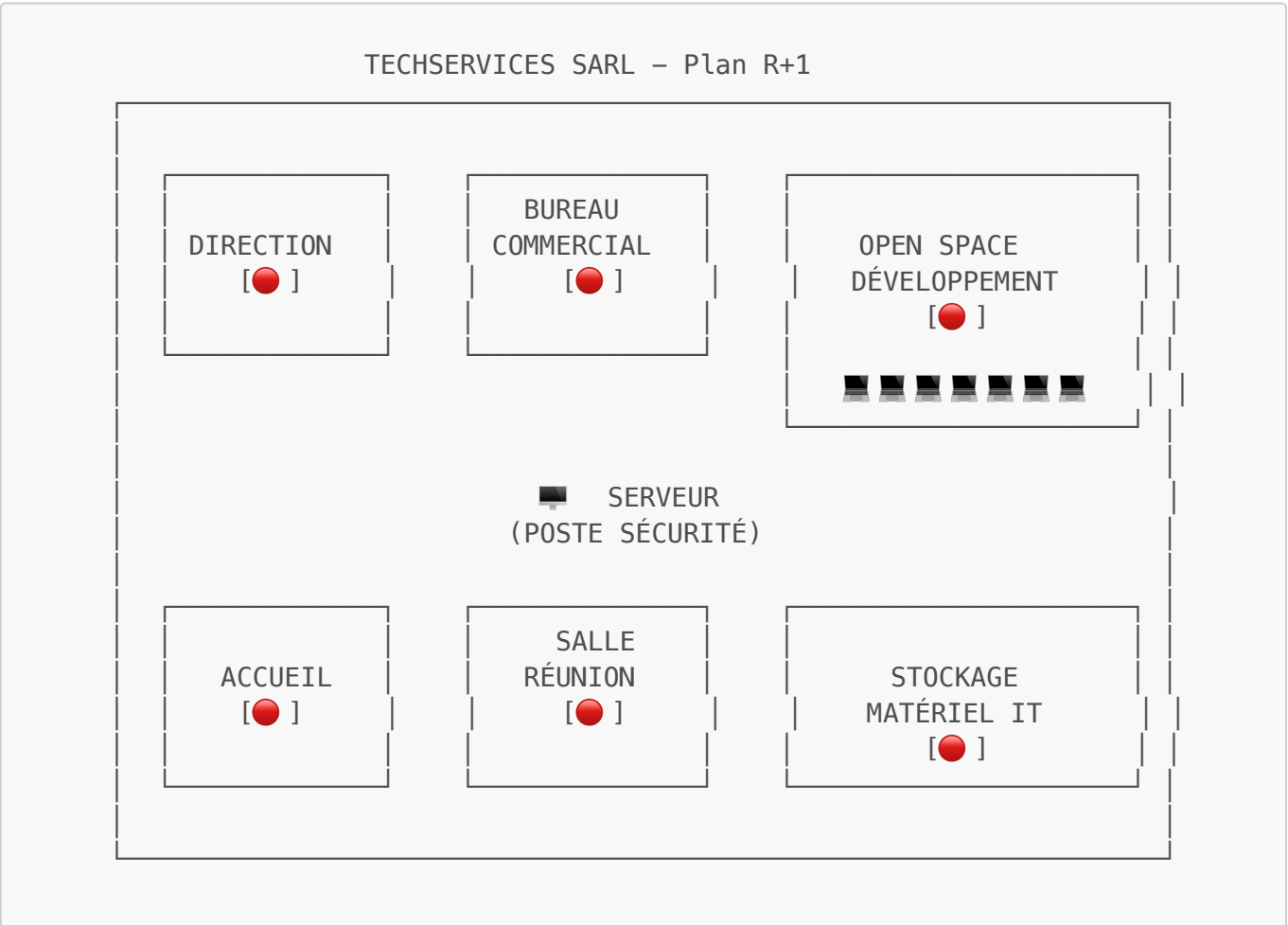
Problématique identifiée




Suite à un incident de sécurité récent (tentative d'intrusion), la direction souhaite équiper les bureaux d'un **système d'alerte d'urgence moderne et économique**.

Cahier des charges :

- Boutons d'alerte dans chaque zone sensible
- Notification immédiate au poste de sécurité
- Système autonome (pas de dépendance Internet)
- Coût maîtrisé (réutilisation matériel existant)
- Extensible (ajout futurs capteurs)

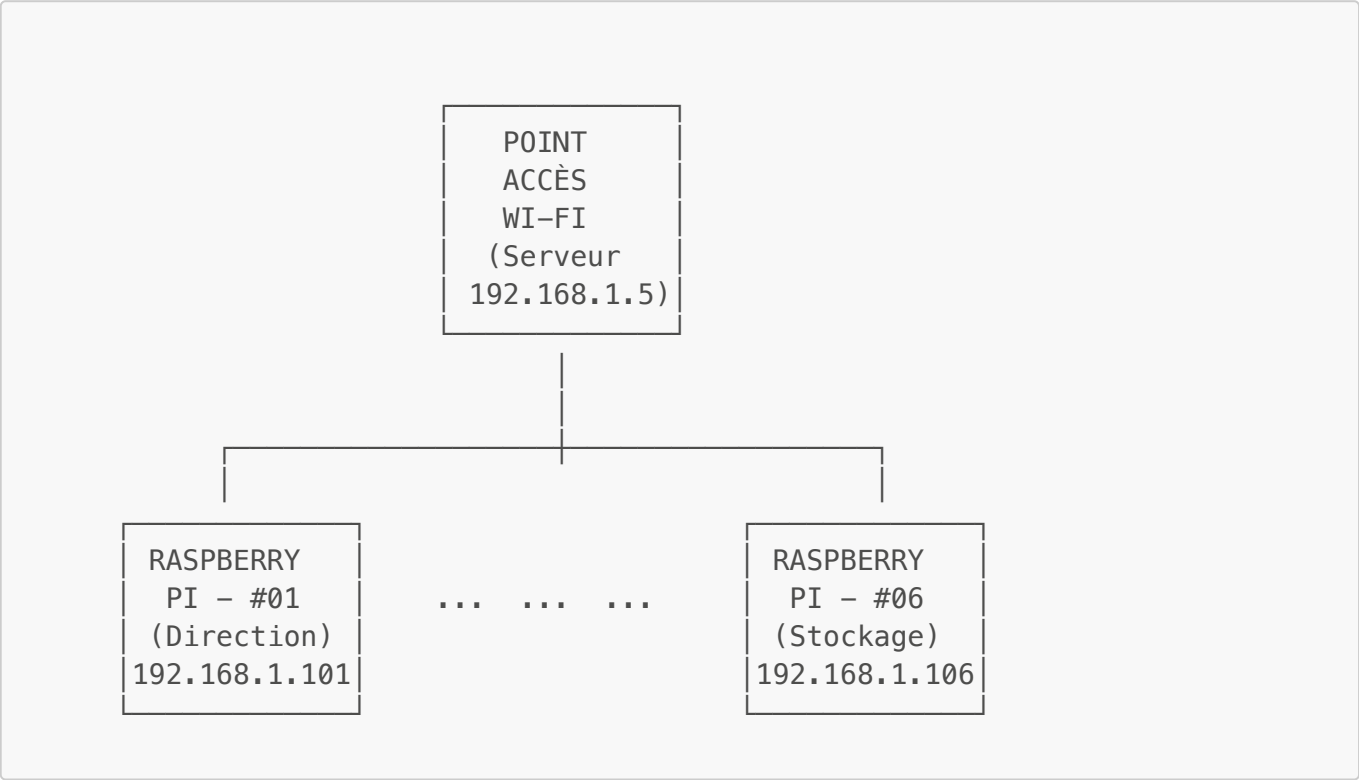
Plan des locaux



- Légende :
-  = Bouton d'alerte d'urgence (Raspberry Pi + bouton + LED)
 -  = Serveur central / Poste de surveillance
 -  = Postes de travail développeurs

Topologie réseau

Architecture réseau actuelle



Spécifications techniques

Zone	Dispositif	IP	Rôle
Direction	RPi-Direction	192.168.1.101	Alerte bureau direction
Commercial	RPi-Commercial	192.168.1.102	Alerte bureau commercial
Open Space	RPi-DevTeam	192.168.1.103	Alerte espace développement
Accueil	RPi-Accueil	192.168.1.104	Alerte accueil
Réunion	RPi-Meeting	192.168.1.105	Alerte salle réunion
Stockage	RPi-Storage	192.168.1.106	Alerte zone matériel
Surveillance	Serveur-Central	192.168.1.5	Réception toutes alertes

Spécifications techniques détaillées

Matériel par point d'alerte

- **Raspberry Pi 3B+** (1GB RAM minimum)
- **Bouton d'urgence** (ou Touche Entrée du clavier)
- **LED d'état** (rouge/vert) + Résistances
- **Alimentation** 5V/3A sécurisée

Installation réseau

- **Connexion** : Wi-Fi WPA3
- **Protocole** : HTTP/HTTPS (port 5000)
- **Sécurité** : Réseau isolé VLAN (optionnel)

Serveur de surveillance

```
Configuration serveur central :  
- Services : Flask + Interface web  
- Monitoring : Supervision état des dispositifs
```

Protocole de communication

Format message d'alerte

```
{  
  "appareil_id": "RPi-Direction", // Doit correspondre à APPAREIL_ID dans  
  config.py  
  "priorité": "Niveau 2",  
  "message": "Alerte d'urgence déclenchée !",  
  "date_heure": "2025-09-22T14:30:25",  
  "source": "RPi-01", // Doit correspondre à APPAREIL_ID ou un  
  identifiant unique  
  "nom": "RPi-Direction", // Doit correspondre à APPAREIL_nom dans  
  config.py  
  "ip_source": "192.168.1.101", // IP du Raspberry Pi émetteur  
  "type": "intrusion", // Doit correspondre à TYPEALERTE_APPAREIL dans  
  config.py  
  "system_info": {"cpu_usage": 45, "memory_usage": 60}, // Résultat de  
  obtenir_info_systeme()  
  "temperature": 31.5, // Résultat de obtenir_temperature()  
  "localisation": "Direction" // Doit correspondre à LOCALISATION dans  
  config.py  
}
```

Codes de réponse serveur

Code	Statut	Signification	Action LED
200	OK	Alerte reçue et traitée	2 clignotements verts

Code	Statut	Signification	Action LED
400	ERREUR	Format incorrect	3 clignotements rouges
503	INDISPONIBLE	Serveur surchargé	4 clignotements rouges
Timeout	RÉSEAU	Pas de connexion	6 clignotements rouges

Niveaux de logs - Guide d'utilisation

Niveau	Utilisation	Exemple
DEBUG	Informations techniques détaillées	Valeurs de variables, état des composants
INFO	Événements normaux importants	Démarrage, envoi d'alerte réussie
WARNING	Problèmes mineurs, situation anormale	Erreur serveur non bloquante
ERROR	Erreurs importantes mais le programme continue	Connexion impossible
CRITICAL	Erreurs graves, arrêt du programme	Panne matérielle critique

Réception automatique de l'alerte

1. **Serveur** : Enregistrement horodaté de l'alerte
2. **Interface** : Affichage temps réel sur écran de surveillance

Ressources techniques

Documentation officielle

- [Raspberry Pi GPIO Programming](#)
- [Flask Web Framework](#)
- [Python Requests Library](#)

Outils de diagnostic réseau

```
# Vérifier l'IP du Pi
hostname -I

# Test connectivité au serveur
ping 192.168.1.5

# Vérification ports ouverts
nmap -p 5000 192.168.1.5
netstat -tlnp | grep 5000

# Logs système
journalctl -u alerte-service -f
```

```
tail -f logs/RPi-XX_application.log # Suivre les logs en temps réel

# Test charge serveur
curl -X POST http://192.168.1.5:5000/test

curl -X POST http://192.168.1.5:5000/alerte -H "Content-Type: application/json" -d '{"message":"Test","source":"RPi-XX"}' # Test manuel d'envoi d'alerte
```

Commandes utiles Raspberry Pi

```
# Processus Python en cours
ps aux | grep python

# Espace disque
df -h

# Température CPU
vcgencmd measure_temp

# Version OS
cat /etc/os-release
```
