Security Monitor Reference

Monitor security events and logins.

Security Monitor (Reference) displays real-time client connections and user sessions during runtime, providing visibility into system access and usage.

The Security Monitor shows:

- Active user sessions
- Client connections
- Process identification
- Network addresses
- Connection duration

Monitor Table Properties
Runtime Access
Connection Types
Monitoring Use Cases
Security Auditing
Best Practices Checklist
Troubleshooting

Monitor Table Properties

The monitor displays results from `@Server.GetAllConnections()`:

Property	Description	Example
UserName	Logged-in user credentials	john.doe, Administrator
ConnectionTime	Session duration since login	00:45:32
IP	Client network address	192.168.1.100
ModuleName	Connected process name	TDisplayClient.exe

Runtime Access

Programmatic Monitoring

csharp

```
// Get all active connections
DataTable connections = @Server.GetAllConnections();

// Process connection data
foreach(DataRow row in connections.Rows)
{
    string user = row["UserName"].ToString();
    string ip = row["IP"].ToString();
    TimeSpan duration = (TimeSpan)row["ConnectionTime"];
    string process = row["ModuleName"].ToString();

    @Info.Trace($"User: {user} from {ip} via {process}");
}
```

Display Integration

csharp

```
// Show connections in DataGrid
DataGrid1.DataSource = @Server.GetAllConnections();
DataGrid1.Refresh();
```

Connection Types

Module	Process	Description
TDisplayClient	Display client	WPF display runtime
TWebClient	Web browser	HTML5 client

TDataAccess	External API	REST/OPC connections
TEngineering	Designer	Engineering connection

Monitoring Use Cases

Active User Count

csharp

```csharp
int activeUsers = @Server.GetAllConnections().Rows.Count;
@Tag.ActiveUserCount = activeUsers;
```

Session Duration Check

csharp

```csharp
var connections = @Server.GetAllConnections();
foreach(DataRow row in connections.Rows)
{
    TimeSpan duration = (TimeSpan)row["ConnectionTime"];
    if (duration.TotalHours > 8)
    {
        // Alert for extended session
        @Alarm.LongSession = true;
    }
}
```

Concurrent User Limit

csharp

```csharp
public void CheckUserLimit()
{
    int maxUsers = 10;
    int current = @Server.GetAllConnections().Rows.Count;

    if (current >= maxUsers)
    {
        @Tag.MaxUsersReached = true;
        // Prevent new logins
    }
}
```

Security Auditing

Log Connections

csharp

```csharp
public void LogConnections()
{
    var connections = @Server.GetAllConnections();

    foreach(DataRow row in connections.Rows)
    {
        @Dataset.Query.AuditLog.Insert(
            DateTime.Now,
            row["UserName"],
            row["IP"],
            row["ModuleName"],
            row["ConnectionTime"]
        );
    }
}
```

Detect Suspicious Activity

csharp

```
// Check for multiple IPs per user
var connections = @Server.GetAllConnections();
var userIPs = new Dictionary<string, List<string>>();

foreach(DataRow row in connections.Rows)
{
    string user = row["UserName"].ToString();
    string ip = row["IP"].ToString();

    if (!userIPs.ContainsKey(user))
        userIPs[user] = new List<string>();

    if (!userIPs[user].Contains(ip))
        userIPs[user].Add(ip);
}

// Alert if user from multiple locations
foreach(var kvp in userIPs)
{
    if (kvp.Value.Count > 1)
    {
        @Alarm.MultipleLocationLogin = true;
    }
}
```

Best Practices Checklist

- [ ] **Regular monitoring** - Check connections periodically
- [ ] **Set user limits** - Enforce concurrent user restrictions
- [ ] **Log sessions** - Maintain audit trail
- [ ] **Monitor duration** - Track extended sessions
- [ ] **Check patterns** - Identify unusual access
- [ ] **Document processes** - Know expected connections
- [ ] **Alert on anomalies** - Detect security issues

Troubleshooting

No connections shown:

- Verify runtime is active
- Check server connectivity
- Confirm monitor permissions
- Review server status

Unknown process:

- Check third-party integrations
- Review custom modules
- Verify client types
- Document all processes

Duplicate users:

- Check AllowShareUser policy
- Review session management
- Verify logout procedures
- Monitor IP addresses

In this section...
