Code to analyze the hacking risk of cloud server

and display it on dashboard

Cloud Risk Dashboard Code

- **Python-based solution that analyzes the hacking risk of a cloud server and displays it on a web-based dashboard using Flask and Dash.**
- **The program evaluates risk based on multiple factors such as network activity, login attempts, and anomaly detection.**
- **This code sets up a Flask-based web application with a Dash dashboard that visualizes hacking risk factors for a cloud server.**
- **The data updates in real-time every 5 seconds** with simulated risk analysis.

```
import dash
from dash import dcc, html
import plotly.graph_objs as go
from flask import Flask
import random
import threading
import time

# Flask server
server = Flask(__name__)

# Dash app
app = dash.Dash(__name__, server=server, routes_pathname_prefix='/dashboard/')

# Simulated risk assessment function
def analyze_hacking_risk():
    return {
        "CPU Usage": random.randint(10, 90),
        "Unauthorized Access Attempts": random.randint(0, 10),
        "Anomalous Traffic": random.randint(0, 100),
        "Malicious Requests": random.randint(0, 50)
    }

# Periodically update risk data
risk_data = []

def update_data():
    global risk_data
    while True:
        risk_data.append(analyze_hacking_risk())
        if len(risk_data) > 10:
            risk_data.pop(0)
        time.sleep(5)
```

```
threading.Thread(target=update_data, daemon=True).start()

# Dash layout
app.layout = html.Div([
    html.H1("Cloud Server Hacking Risk Dashboard"),
    dcc.Interval(id='interval-update', interval=5000, n_intervals=0),
    dcc.Graph(id='risk-graph')
])

@app.callback(
    dash.dependencies.Output('risk-graph', 'figure'),
    [dash.dependencies.Input('interval-update', 'n_intervals')]
)
def update_graph(n):
    global risk_data
    if not risk_data:
        return go.Figure()

    latest_data = risk_data[-1]
    categories = list(latest_data.keys())
    values = list(latest_data.values())

    fig = go.Figure([go.Bar(x=categories, y=values)])
    fig.update_layout(title="Hacking Risk Factors", xaxis_title="Factors", yaxis_title="Severity")
    return fig

if __name__ == '__main__':
    app.run(debug=True)
```