Run case 1 and case 2 with entire samples

In [1]:
```
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC, OneClassSVM
from sklearn.ensemble import IsolationForest
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
```

In [11]:
```
pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\anaa1\anaconda3\lib\site-packages (3.0.10)
Requirement already satisfied: et_xmlfile in c:\users\anaa1\anaconda3\lib\site-packages (from ope
npyxl) (1.1.0)
Note: you may need to restart the kernel to use updated packages.
```

In [12]:
```
def load_chip_data_xlsx(folder_path):
    all_data = []
    labels = []
    for i in range(1, 34):  # Chip1 to Chip33
        file_path = os.path.join(folder_path, f"Chip{i}.xlsx")
        df = pd.read_excel(file_path, header=None)
        # Row 1 and 25 are Trojan-Free (label 0), Rows 2–24 are Trojan-Inserted (label 1)
        tf_rows = df.iloc[[0, 24]].values
        ti_rows = df.iloc[1:24].values
        all_data.extend(tf_rows)
        labels.extend([0] * len(tf_rows))
        all_data.extend(ti_rows)
        labels.extend([1] * len(ti_rows))
    return np.array(all_data), np.array(labels)
```

In [13]:
```
def run_case1(X, y, classifier_type='rf'):
    results = []
    for _ in range(20):
        X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=24, stratify=y)
        if classifier_type == 'rf':
            clf = RandomForestClassifier(n_estimators=100)
        elif classifier_type == 'svm':
            clf = SVC(kernel='rbf')
        clf.fit(X_train, y_train)
        y_pred = clf.predict(X_test)
        acc = accuracy_score(y_test, y_pred)
        results.append(acc)
    return np.mean(results)
```

In [14]:
```
def run_case2(X, y, classifier_type='ocsvm'):
    tf_data = X[y == 0]
    ti_data = X[y == 1]
    results = []
    for _ in range(20):
        X_train, X_test_tf = train_test_split(tf_data, train_size=24)
        X_test = np.vstack([X_test_tf, ti_data])
        y_true = np.array([0] * len(X_test_tf) + [1] * len(ti_data))
        if classifier_type == 'ocsvm':
            clf = OneClassSVM(gamma='auto')
        elif classifier_type == 'iforest':
            clf = IsolationForest()
        clf.fit(X_train)
```

```
        y_pred = clf.predict(X_test)
        y_pred = np.where(y_pred == 1, 0, 1)  # Map 1→TF, -1→TI
        acc = accuracy_score(y_true, y_pred)
        results.append(acc)
    return np.mean(results)
```

In [16]:
```
folder = "ROFreq"
X, y = load_chip_data_xlsx(folder)

print("Case 1 - Random Forest Accuracy:", run_case1(X, y, 'rf'))
print("Case 1 - SVM Accuracy:", run_case1(X, y, 'svm'))

print("Case 2 - One-Class SVM Accuracy:", run_case2(X, y, 'ocsvm'))
print("Case 2 - Isolation Forest Accuracy:", run_case2(X, y, 'iforest'))
```

```
Case 1 - Random Forest Accuracy: 0.913358302122347
Case 1 - SVM Accuracy: 0.919912609238452
Case 2 - One-Class SVM Accuracy: 0.9542446941323346
Case 2 - Isolation Forest Accuracy: 0.9360799001248441
```

6, 12, 24, all remaining samples are used for evaluation

In [17]:
```
def group_by_trojan_type(X, y):
    trojan_groups = {i: [] for i in range(23)}  # Trojan types 0–22
    tf_data = []
    for i in range(0, len(X), 25):  # Each chip has 25 rows
        tf_data.append(X[i])        # Row 0 = TF
        tf_data.append(X[i+24])     # Row 24 = TF
        for j in range(23):         # Rows 1–23 = TI
            trojan_groups[j].append(X[i+1+j])
    return np.array(tf_data), trojan_groups
```

In [18]:
```
def evaluate_case1(tf_data, trojan_groups, sample_size, classifier_type='rf'):
    tf_count = sample_size // 2
    ti_count = sample_size // 2
    results = []
    for _ in range(20):
        tf_train = tf_data[np.random.choice(len(tf_data), tf_count, replace=False)]
        ti_train = []
        ti_eval = []
        for j in range(23):
            samples = np.array(trojan_groups[j])
            selected = samples[np.random.choice(len(samples), ti_count, replace=False)]
            ti_train.extend(selected)
            ti_eval.extend([s for s in samples if s.tolist() not in selected.tolist()])
        X_train = np.vstack([tf_train, ti_train])
        y_train = np.array([0]*tf_count + [1]*len(ti_train))
        X_test = np.vstack([tf_data, ti_eval])
        y_test = np.array([0]*len(tf_data) + [1]*len(ti_eval))
        clf = RandomForestClassifier() if classifier_type == 'rf' else SVC(kernel='rbf')
        clf.fit(X_train, y_train)
        y_pred = clf.predict(X_test)
        acc = accuracy_score(y_test, y_pred)
        results.append(acc)
    return np.mean(results)
```

In [19]:
```
def evaluate_case2(tf_data, trojan_groups, sample_size, classifier_type='ocsvm'):
    results = []
    for _ in range(20):
        tf_train = tf_data[np.random.choice(len(tf_data), sample_size, replace=False)]
        ti_eval = []
        for j in range(23):
            ti_eval.extend(trojan_groups[j])
        X_test = np.vstack([tf_data, ti_eval])
```

```
        y_test = np.array([0]*len(tf_data) + [1]*len(ti_eval))
        clf = OneClassSVM(gamma='auto') if classifier_type == 'ocsvm' else IsolationForest()
        clf.fit(tf_train)
        y_pred = clf.predict(X_test)
        y_pred = np.where(y_pred == 1, 0, 1)
        acc = accuracy_score(y_test, y_pred)
        results.append(acc)
    return np.mean(results)
```

In [20]:
```
def run_all_evaluations(X, y):
    tf_data, trojan_groups = group_by_trojan_type(X, y)
    sample_sizes = [6, 12, 24]
    classifiers_case1 = ['rf', 'svm']
    classifiers_case2 = ['ocsvm', 'iforest']

    print("=== Case 1: Supervised ===")
    for size in sample_sizes:
        for clf in classifiers_case1:
            acc = evaluate_case1(tf_data, trojan_groups, size, classifier_type=clf)
            print(f"Sample Size {size} | Classifier {clf.upper()} | Accuracy: {acc:.2%}")

    print("\n=== Case 2: One-Class ===")
    for size in sample_sizes:
        for clf in classifiers_case2:
            acc = evaluate_case2(tf_data, trojan_groups, size, classifier_type=clf)
            print(f"Sample Size {size} | Classifier {clf.upper()} | Accuracy: {acc:.2%}")
```

In [21]:
```
X, y = load_chip_data_xlsx(folder)
run_all_evaluations(X, y)
```

```
=== Case 1: Supervised ===
Sample Size 6 | Classifier RF | Accuracy: 90.93%
Sample Size 6 | Classifier SVM | Accuracy: 91.23%
Sample Size 12 | Classifier RF | Accuracy: 90.43%
Sample Size 12 | Classifier SVM | Accuracy: 90.29%
Sample Size 24 | Classifier RF | Accuracy: 89.39%
Sample Size 24 | Classifier SVM | Accuracy: 87.74%

=== Case 2: One-Class ===
Sample Size 6 | Classifier OCSVM | Accuracy: 92.30%
Sample Size 6 | Classifier IFOREST | Accuracy: 61.76%
Sample Size 12 | Classifier OCSVM | Accuracy: 92.44%
Sample Size 12 | Classifier IFOREST | Accuracy: 82.91%
Sample Size 24 | Classifier OCSVM | Accuracy: 92.82%
Sample Size 24 | Classifier IFOREST | Accuracy: 86.90%
```

Evaluate per Trojan per case

In [22]:
```
def evaluate_case1_per_trojan(tf_data, trojan_groups, sample_size, classifier_type='rf'):
    tf_count = sample_size // 2
    ti_count = sample_size // 2
    trojan_accuracies = []

    for trojan_id in range(23):
        results = []
        for _ in range(20):
            tf_train = tf_data[np.random.choice(len(tf_data), tf_count, replace=False)]
            ti_train = np.array(trojan_groups[trojan_id])[np.random.choice(len(trojan_groups[troj
            X_train = np.vstack([tf_train, ti_train])
            y_train = np.array([0]*tf_count + [1]*ti_count)

            # Evaluation set: all TF + all TI of this Trojan type not in training
            ti_eval = [x for x in trojan_groups[trojan_id] if x.tolist() not in ti_train.tolist()
            X_test = np.vstack([tf_data, ti_eval])
            y_test = np.array([0]*len(tf_data) + [1]*len(ti_eval))
```

```
            clf = RandomForestClassifier() if classifier_type == 'rf' else SVC(kernel='rbf')
            clf.fit(X_train, y_train)
            y_pred = clf.predict(X_test)
            acc = accuracy_score(y_test, y_pred)
            results.append(acc)
        trojan_accuracies.append(np.mean(results))
    return trojan_accuracies
```

In [23]:
```
def evaluate_case2_per_trojan(tf_data, trojan_groups, sample_size, classifier_type='ocsvm'):
    trojan_accuracies = []

    for trojan_id in range(23):
        results = []
        for _ in range(20):
            tf_train = tf_data[np.random.choice(len(tf_data), sample_size, replace=False)]
            ti_eval = trojan_groups[trojan_id]
            X_test = np.vstack([tf_data, ti_eval])
            y_test = np.array([0]*len(tf_data) + [1]*len(ti_eval))

            clf = OneClassSVM(gamma='auto') if classifier_type == 'ocsvm' else IsolationForest()
            clf.fit(tf_train)
            y_pred = clf.predict(X_test)
            y_pred = np.where(y_pred == 1, 0, 1)
            acc = accuracy_score(y_test, y_pred)
            results.append(acc)
        trojan_accuracies.append(np.mean(results))
    return trojan_accuracies
```

In [24]:
```
def run_per_trojan_evaluation(X, y):
    tf_data, trojan_groups = group_by_trojan_type(X, y)
    sample_sizes = [6, 12, 24]

    for size in sample_sizes:
        print(f"\n=== Sample Size: {size} ===")
        for clf in ['rf', 'svm']:
            accs = evaluate_case1_per_trojan(tf_data, trojan_groups, size, classifier_type=clf)
            print(f"Case 1 | {clf.upper()} | Avg Accuracy per Trojan Type:")
            for i, acc in enumerate(accs):
                print(f"  Trojan {i+1}: {acc:.2%}")

        for clf in ['ocsvm', 'iforest']:
            accs = evaluate_case2_per_trojan(tf_data, trojan_groups, size, classifier_type=clf)
            print(f"Case 2 | {clf.upper()} | Avg Accuracy per Trojan Type:")
            for i, acc in enumerate(accs):
                print(f"  Trojan {i+1}: {acc:.2%}")
```

In [25]:
```
X, y = load_chip_data_xlsx(folder)
run_per_trojan_evaluation(X, y)
```

```
=== Sample Size: 6 ===
Case 1 | RF | Avg Accuracy per Trojan Type:
  Trojan 1: 63.54%
  Trojan 2: 74.09%
  Trojan 3: 80.79%
  Trojan 4: 85.25%
  Trojan 5: 78.24%
  Trojan 6: 80.30%
  Trojan 7: 84.93%
  Trojan 8: 81.42%
  Trojan 9: 85.50%
  Trojan 10: 72.30%
  Trojan 11: 74.84%
  Trojan 12: 75.64%
  Trojan 13: 80.51%
  Trojan 14: 78.92%
  Trojan 15: 75.55%
  Trojan 16: 88.96%
  Trojan 17: 75.69%
  Trojan 18: 79.75%
  Trojan 19: 79.12%
  Trojan 20: 87.03%
  Trojan 21: 84.16%
  Trojan 22: 77.45%
  Trojan 23: 63.20%
Case 1 | SVM | Avg Accuracy per Trojan Type:
  Trojan 1: 63.87%
  Trojan 2: 69.35%
  Trojan 3: 80.96%
  Trojan 4: 80.96%
  Trojan 5: 78.44%
  Trojan 6: 79.79%
  Trojan 7: 75.03%
  Trojan 8: 75.80%
  Trojan 9: 81.21%
  Trojan 10: 64.53%
  Trojan 11: 66.04%
  Trojan 12: 70.87%
  Trojan 13: 72.10%
  Trojan 14: 73.09%
  Trojan 15: 72.72%
  Trojan 16: 77.74%
  Trojan 17: 63.94%
  Trojan 18: 73.37%
  Trojan 19: 77.99%
  Trojan 20: 83.55%
  Trojan 21: 68.58%
  Trojan 22: 66.20%
  Trojan 23: 64.30%
Case 2 | OCSVM | Avg Accuracy per Trojan Type:
  Trojan 1: 36.01%
  Trojan 2: 36.52%
  Trojan 3: 36.67%
  Trojan 4: 36.97%
  Trojan 5: 36.57%
  Trojan 6: 36.31%
  Trojan 7: 36.31%
  Trojan 8: 36.46%
  Trojan 9: 36.52%
  Trojan 10: 36.77%
  Trojan 11: 36.92%
  Trojan 12: 36.52%
  Trojan 13: 37.02%
  Trojan 14: 36.36%
  Trojan 15: 36.72%
  Trojan 16: 36.26%
  Trojan 17: 36.52%
```

```
  Trojan 18: 36.72%
  Trojan 19: 36.16%
  Trojan 20: 36.97%
  Trojan 21: 36.57%
  Trojan 22: 36.92%
  Trojan 23: 36.16%
Case 2 | IFOREST | Avg Accuracy per Trojan Type:
  Trojan 1: 45.40%
  Trojan 2: 44.90%
  Trojan 3: 53.74%
  Trojan 4: 53.89%
  Trojan 5: 54.65%
  Trojan 6: 58.18%
  Trojan 7: 51.87%
  Trojan 8: 52.93%
  Trojan 9: 50.20%
  Trojan 10: 53.64%
  Trojan 11: 50.51%
  Trojan 12: 50.30%
  Trojan 13: 50.45%
  Trojan 14: 46.92%
  Trojan 15: 53.33%
  Trojan 16: 52.63%
  Trojan 17: 51.46%
  Trojan 18: 46.41%
  Trojan 19: 53.08%
  Trojan 20: 59.44%
  Trojan 21: 55.00%
  Trojan 22: 53.23%
  Trojan 23: 53.03%

=== Sample Size: 12 ===
Case 1 | RF | Avg Accuracy per Trojan Type:
  Trojan 1: 65.17%
  Trojan 2: 80.95%
  Trojan 3: 85.59%
  Trojan 4: 91.81%
  Trojan 5: 92.40%
  Trojan 6: 93.11%
  Trojan 7: 91.26%
  Trojan 8: 89.23%
  Trojan 9: 92.82%
  Trojan 10: 80.64%
  Trojan 11: 81.63%
  Trojan 12: 82.59%
  Trojan 13: 83.29%
  Trojan 14: 84.17%
  Trojan 15: 86.62%
  Trojan 16: 87.52%
  Trojan 17: 81.62%
  Trojan 18: 85.76%
  Trojan 19: 93.05%
  Trojan 20: 93.47%
  Trojan 21: 90.30%
  Trojan 22: 86.47%
  Trojan 23: 64.88%
Case 1 | SVM | Avg Accuracy per Trojan Type:
  Trojan 1: 63.76%
  Trojan 2: 75.49%
  Trojan 3: 85.31%
  Trojan 4: 86.36%
  Trojan 5: 93.31%
  Trojan 6: 92.33%
  Trojan 7: 85.46%
  Trojan 8: 72.86%
  Trojan 9: 87.10%
  Trojan 10: 63.60%
```

```
  Trojan 11: 68.84%
  Trojan 12: 70.77%
  Trojan 13: 78.82%
  Trojan 14: 81.59%
  Trojan 15: 84.75%
  Trojan 16: 87.70%
  Trojan 17: 66.02%
  Trojan 18: 82.86%
  Trojan 19: 92.39%
  Trojan 20: 91.96%
  Trojan 21: 88.96%
  Trojan 22: 70.81%
  Trojan 23: 65.01%
Case 2 | OCSVM | Avg Accuracy per Trojan Type:
  Trojan 1: 37.83%
  Trojan 2: 39.44%
  Trojan 3: 39.49%
  Trojan 4: 40.20%
  Trojan 5: 40.05%
  Trojan 6: 40.10%
  Trojan 7: 39.39%
  Trojan 8: 39.60%
  Trojan 9: 39.90%
  Trojan 10: 40.20%
  Trojan 11: 40.56%
  Trojan 12: 39.39%
  Trojan 13: 39.49%
  Trojan 14: 40.35%
  Trojan 15: 39.65%
  Trojan 16: 39.95%
  Trojan 17: 39.65%
  Trojan 18: 39.24%
  Trojan 19: 40.05%
  Trojan 20: 40.00%
  Trojan 21: 40.56%
  Trojan 22: 39.39%
  Trojan 23: 38.43%
Case 2 | IFOREST | Avg Accuracy per Trojan Type:
  Trojan 1: 43.69%
  Trojan 2: 56.57%
  Trojan 3: 56.52%
  Trojan 4: 58.38%
  Trojan 5: 59.85%
  Trojan 6: 61.06%
  Trojan 7: 59.49%
  Trojan 8: 62.12%
  Trojan 9: 57.07%
  Trojan 10: 50.96%
  Trojan 11: 52.68%
  Trojan 12: 54.65%
  Trojan 13: 55.30%
  Trojan 14: 58.99%
  Trojan 15: 57.78%
  Trojan 16: 58.23%
  Trojan 17: 55.00%
  Trojan 18: 59.75%
  Trojan 19: 59.44%
  Trojan 20: 64.24%
  Trojan 21: 61.46%
  Trojan 22: 61.11%
  Trojan 23: 47.42%

=== Sample Size: 24 ===
Case 1 | RF | Avg Accuracy per Trojan Type:
  Trojan 1: 64.51%
  Trojan 2: 85.54%
  Trojan 3: 91.16%
```

```
  Trojan 4: 95.15%
  Trojan 5: 95.33%
  Trojan 6: 93.23%
  Trojan 7: 91.68%
  Trojan 8: 90.72%
  Trojan 9: 95.21%
  Trojan 10: 85.66%
  Trojan 11: 88.44%
  Trojan 12: 86.62%
  Trojan 13: 86.13%
  Trojan 14: 86.92%
  Trojan 15: 89.76%
  Trojan 16: 92.71%
  Trojan 17: 86.72%
  Trojan 18: 89.55%
  Trojan 19: 95.21%
  Trojan 20: 94.93%
  Trojan 21: 90.41%
  Trojan 22: 88.95%
  Trojan 23: 68.97%
Case 1 | SVM | Avg Accuracy per Trojan Type:
  Trojan 1: 61.00%
  Trojan 2: 81.66%
  Trojan 3: 90.51%
  Trojan 4: 95.14%
  Trojan 5: 93.88%
  Trojan 6: 92.27%
  Trojan 7: 91.72%
  Trojan 8: 84.84%
  Trojan 9: 92.85%
  Trojan 10: 62.95%
  Trojan 11: 65.72%
  Trojan 12: 69.44%
  Trojan 13: 81.70%
  Trojan 14: 80.69%
  Trojan 15: 88.63%
  Trojan 16: 90.43%
  Trojan 17: 67.00%
  Trojan 18: 87.71%
  Trojan 19: 93.68%
  Trojan 20: 92.78%
  Trojan 21: 88.79%
  Trojan 22: 64.63%
  Trojan 23: 62.86%
Case 2 | OCSVM | Avg Accuracy per Trojan Type:
  Trojan 1: 43.64%
  Trojan 2: 48.18%
  Trojan 3: 47.17%
  Trojan 4: 47.88%
  Trojan 5: 48.13%
  Trojan 6: 48.59%
  Trojan 7: 46.77%
  Trojan 8: 46.82%
  Trojan 9: 47.68%
  Trojan 10: 46.57%
  Trojan 11: 46.82%
  Trojan 12: 46.92%
  Trojan 13: 48.54%
  Trojan 14: 48.64%
  Trojan 15: 46.82%
  Trojan 16: 47.02%
  Trojan 17: 47.83%
  Trojan 18: 47.22%
  Trojan 19: 47.12%
  Trojan 20: 47.68%
  Trojan 21: 47.63%
  Trojan 22: 45.56%
```

```
  Trojan 23: 43.18%
Case 2 | IFOREST | Avg Accuracy per Trojan Type:
  Trojan 1: 47.83%
  Trojan 2: 62.53%
  Trojan 3: 68.23%
  Trojan 4: 70.05%
  Trojan 5: 70.71%
  Trojan 6: 70.71%
  Trojan 7: 69.39%
  Trojan 8: 71.11%
  Trojan 9: 69.49%
  Trojan 10: 62.93%
  Trojan 11: 67.78%
  Trojan 12: 67.32%
  Trojan 13: 66.11%
  Trojan 14: 65.76%
  Trojan 15: 67.07%
  Trojan 16: 67.83%
  Trojan 17: 66.97%
  Trojan 18: 67.12%
  Trojan 19: 70.25%
  Trojan 20: 70.20%
  Trojan 21: 66.77%
  Trojan 22: 67.93%
  Trojan 23: 51.46%
```

In [ ]: