Phase 3 Design: FL for Advanced Behavioral Analysis & Dispute Resolution in ChainFLIP (TFF)

This document outlines the design for Phase 3 of the Federated Learning (FL) integration into your ChainFLIP project. Phase 3 focuses on leveraging TFF for advanced behavioral analysis, particularly incorporating insights from the `DisputeResolution.sol` contract and detecting more subtle or complex malicious patterns.

This builds upon the setups from Phase 1 (Sybil Detection) and Phase 2 (Batch Processing Monitoring).

1. Goals of Phase 3

Phase 3 aims to achieve a deeper understanding of node behaviors and system integrity by:

- **Detecting Arbitrator Bias:** Identifying arbitrators in `DisputeResolution.sol` who consistently favor certain parties or outcomes without clear justification from on-chain evidence.
- **Predicting High-Risk Disputes:** Identifying disputes that are likely to be contentious, fraudulent, or require significant intervention based on the characteristics of involved parties and the dispute itself.
- **Identifying Nodes Prone to Disputes:** Flagging nodes (Manufacturers, Transporters, etc.) that are frequently involved in disputes, either as initiators or subjects.
- **Advanced Anomaly Detection:** Detecting subtle changes in a node’s behavior over time that might indicate a future compromise or shift towards malicious activity (e.g., using time-series analysis).
- **Correlating Dispute Outcomes with Prior Behavior:** Understanding if prior FL-flagged behaviors (from Phase 1 or 2) correlate with involvement or outcomes in disputes.

2. Data Sources for Phase 3 FL Clients

FL clients will need to access and process data from:

- **`DisputeResolution.sol` (Primary Source):**
  - Events: `DisputeOpened`, `EvidenceSubmitted` (if CIDs of evidence are logged), `ArbitratorProposed`, `ArbitratorVoted`, `DisputeResolved` (outcome, involved parties, arbitrators).
  - State: Details of active and past disputes, arbitrator lists, voting records.
- **`NodeManagement.sol`:** Node reputations, roles, verification status (to correlate with dispute involvement).
- **`BatchProcessing.sol` & `Marketplace.sol` / `NFTCore.sol`:** Historical data about products, transactions, and batch processing involving nodes that later become part of disputes.
- **FL Model Outputs from Phase 1 & 2:** Historical risk scores for nodes can be used as features.

3. Feature Engineering for Phase 3

This phase requires more sophisticated feature engineering, potentially involving time-series data and relational data.

For Arbitrators:

1. `arbitrator_agreement_rate_with_outcome (0-1)`: How often an arbitrator's vote aligns with the final resolved outcome of disputes they arbitrate.
2. `arbitrator_vote_consistency_with_peers (0-1)`: How often an arbitrator votes similarly to other arbitrators on the same dispute panel.
3. `arbitrator_bias_score_party_type (float)`: A more complex feature. E.g., does an arbitrator disproportionately favor Manufacturers over Transporters, or vice-versa, compared to the average? (Requires careful definition and potentially statistical analysis of past votes).
4. `arbitrator_disputes_participated_count (int)`.
5. `arbitrator_avg_time_to_vote (float)`: Average time taken to vote after being assigned (if timestamps are available).

For General Nodes (involved in disputes):

1. `node_dispute_initiation_rate (float)`: How often this node initiates disputes relative to its activity level.
2. `node_dispute_subject_rate (float)`: How often this node is the subject of a dispute relative to its activity level.

3. `node_dispute_win_rate (0-1)` : Percentage of disputes this node is involved in where the outcome is favorable to them.
4. `node_avg_dispute_value (float)` : Average value or severity of disputes involving this node (if quantifiable).
5. `node_prior_fl_risk_score_avg (float)` : Average risk score from Phase 1/2 models leading up to a dispute.
6. **Time-Series Features (for a specific node):**
    - Rolling average of transaction frequency over last N days/weeks.
    - Sudden spikes/dips in activity levels.
    - Changes in interaction patterns (e.g., suddenly interacting with many new, low-reputation nodes).

For Disputes Themselves:

1. `dispute_complexity_score (int/float)` : Based on number of parties, amount of evidence (if trackable), value transacted.
2. `dispute_arbitrator_panel_avg_rep (float)` : Average reputation of arbitrators assigned.
3. `dispute_parties_avg_rep_difference (float)` : Absolute difference in average reputation between disputing parties.

Labels for Training:

- **Arbitrator Bias:** Label arbitrators as "biased" (1) or "unbiased" (0) based on manual review of a subset of their voting history or statistical deviations from norms.
- **High-Risk Dispute:** Label disputes as "high-risk" (1) or "normal-risk" (0) based on outcomes (e.g., disputes that were overturned, took excessively long, or involved proven fraud).
- **Dispute-Prone Node:** Label nodes based on their historical dispute involvement frequency and outcomes.
- **Anomalous Behavior (Time-Series):** Use unsupervised anomaly detection models (e.g., autoencoders, LSTMs) to identify deviations from a node’s established normal behavior.

4. FL Model Adaptation for Phase 3 (TFF with Keras)

- **Multiple Models:** It might be necessary to train separate FL models for different Phase 3 tasks (e.g., one for arbitrator bias, another for predicting high-risk disputes, a third for node behavioral anomaly detection).
- **Model Input:** The `NUM_FEATURES` and `ELEMENT_SPEC` will vary for each model.

- **Model Architectures (`model_definition_phase3.py` - or multiple files):
  - **For classification tasks (bias, high-risk dispute):** The Keras Sequential model used in Phase 1/2 can be adapted. Consider increasing complexity if features are rich.
  - **For Time-Series Anomaly Detection:** Recurrent Neural Networks (RNNs), LSTMs, or GRUs are suitable. TFF can wrap Keras models using these layers. The input data would be sequences of features over time for each node.
  - **Unsupervised Models:** Autoencoders can be trained via FL to learn a compressed representation of "normal" behavior. High reconstruction error on new data indicates an anomaly.

5. TFF Implementation Sketch for Phase 3

Create a new subdirectory, e.g., `ChainFLIP_FL_Dev/tff_advanced_analysis/` . This might contain further subdirectories for each specific Phase 3 task (e.g., `arbitrator_bias` , `dispute_risk`).

1. **`data_preparation_phase3.py` (and/or task-specific data prep files):**

   - Implement logic to fetch and process data from `DisputeResolution.sol` and other relevant contracts.
   - Perform the advanced feature engineering (Section 3).
   - For time-series models, data needs to be shaped into sequences (e.g., `[num_samples, timesteps, num_features]`).
   - Define appropriate `ELEMENT_SPEC` for each model.

2. **`model_definition_phase3.py` (and/or task-specific model files):**

   - Define Keras models suitable for each task (classification, RNN/LSTM for time-series, autoencoders).
   - Wrap them using `tff.learning.models.from_keras_model` or, for more custom TFF models (like unsupervised ones), you might need to implement the `tff.learning.models.VariableModel` interface more directly or use TFF's functional model constructs.

3. **`federated_training_phase3.py` (and/or task-specific training files):**

   - The `build_weighted_fed_avg` process can still be used for supervised Keras models.
   - For custom unsupervised TFF models or more complex federated algorithms, you might need to define custom `tff.Computation` s using `tff.federated_broadcast` , `tff.federated_map` , `tff.federated_aggregate` , etc.

4. **`run_simulation_phase3.py` (and/or task-specific simulation files):**
    - Adapt the simulation loop for each Phase 3 task.
    - Interpretation of metrics will be task-dependent.

6. Integration with Admin Dashboard for Phase 3

- **New Dashboard Sections/Alerts:**
    - **Arbitrator Performance:** Display arbitrator metrics, bias scores, and flag potentially biased arbitrators.
    - **Dispute Risk Assessment:** Show predicted risk levels for ongoing or new disputes.
    - **Node Dispute Profile:** Provide a history of a node's involvement in disputes and their FL-derived dispute-proneness score.
    - **Behavioral Anomaly Alerts:** Notify admins of nodes exhibiting significant deviations from their learned normal behavior patterns.
- **Admin Actions:**
    - **Review Arbitrators:** Investigate flagged arbitrators, potentially leading to removal or retraining.
    - **Prioritize High-Risk Disputes:** Allocate more attention or experienced arbitrators to disputes predicted as high-risk.
    - **Counsel/Monitor Dispute-Prone Nodes:** Engage with nodes frequently involved in disputes.
    - **Investigate Anomalous Nodes:** Further scrutinize nodes flagged by time-series anomaly detection.

7. Smart Contract Considerations for Phase 3

- **`DisputeResolution.sol` Enhancements (Potential):**
    - Ensure detailed events are emitted for all critical stages of a dispute, including evidence submission (even if just CIDs) and arbitrator votes with timestamps.
    - Consider adding functions to query aggregated statistics about past disputes if feasible and useful (though this is usually better done off-chain by FL clients).
- **Timestamps:** Consistent and reliable timestamps for on-chain actions are crucial for time-series analysis.

8. Testing Phase 3

- **Complex Synthetic Data:** Generating data for Phase 3 is challenging.
    - **Arbitrator Bias:** Simulate arbitrators with predefined biases in their voting patterns for certain types of disputes or parties.
    - **Dispute Scenarios:** Create synthetic dispute data with varying characteristics (complexity, involved party reputations) and predefined outcomes (e.g., some clearly fraudulent, some legitimate).
    - **Time-Series Behavior:** Generate sequences of actions for nodes, with some nodes exhibiting sudden changes or drifting into malicious patterns over time.
- **Metrics:** Evaluate models based on their ability to correctly identify biased arbitrators, predict high-risk disputes, or flag anomalous behavioral sequences.

Phase 3 represents a significant step up in complexity, moving towards more nuanced and predictive insights. It will likely require iterative development and refinement of features, models, and simulation environments.