We sincerely thank all the reviewers for their thorough feedback, and detailed comments. We will incorporate all feedback related to presentation (typos, stating benefits of variations, ending abruptly, etc.) in the draft. We'd like to start by addressing the feedback of reviewer 2:

- *If a selected participant fails to provide gradient update*: For clarity of the framework, we have not modelled various kinds of 'drop-outs' that could take place in such a system, e.g., dropping out after a random check-in, disconnecting after receiving the model from the server, etc. Our framework is designed such that the privacy guarantees will not degrade due to such drop-outs. One way to envision this is a time-based system, where the server stops waiting for some client after a predetermined amount of time, and proceeds with applying an all-zero model update with noise. Thus, the utility of the system will depend on such factors, but not privacy.
- *Assumptions of a trusted server and privileged communications*: The primary focus of our work is to ensure a model published to the world doesn't regurgitate private data, for e.g., a language model accurately completing the sentence "John Doe's credit card number is ...". Requiring less trust from the server is also an interesting problem for a distributed setup, and addressing this may be best accomplished by orthogonal techniques like Secure Multi-Party Computation. Moreover, in our framework each model update sent from a client device does obtain a local differential privacy (LDP) guarantee, which is what gets amplified for a central DP guarantee. The amplified guarantees do assume both the stated assumptions, however we would like to state that these assumptions hold for all existing amplification guarantees for a general distributed learning setting (namely, privacy amplification by sampling and shuffling).
- *No analysis of convergence; why optimal ... unlikely to work in NN training?*: We do state that the utility guarantees of our main protocol are *optimal for convex ERMs*, and we also provide a bound on the number of "dummy" updates in general. The method of DP-SGD is commonly used in practice for training deep NNs (without any formal utility guarantees), and our technique is modeled on that. Moreover, for non-convex settings, very little is known in general about optimality/convergence in the DP literature.
- *Threat by malicious clients could be higher due to local randomness*: The assumption of no malicious clients in our setup is required only by the "thrifty" updates version of the algorithm (Section 4.1), and the privacy guarantees degrade smoothly with the proportion of malicious users (similar to the analyses of privacy amplification via shuffling). For the other two versions of our algorithm (the main version in Section 3.1, and the sliding window version in Section 4.2), the amplified privacy of a client device is crucially dependent on that client and the server following the protocol, and thus, it won't degrade by the actions of malicious clients. In other words, the central DP guarantee provided is for all clients that follow the protocol along with the server.
- *Other variations could be compared with the approach in Section 3.1*: Since the submission deadline, we have worked out a comparison of the utility of the algorithm in Section 4.1 to the main algorithm in Section 3.1, in that for convex non-smooth losses and $m \ll n$, the main algorithm provides a better utility whereas the utility bounds are incomparable for smooth losses.

Addressing the feedback of reviewers 1 and 2 regarding empirical evaluation of random check-ins: There are various design choices to be made for modeling real-world device availability (e.g., diurnal variations, overlap in availability for small/large/global populations, device capabilities across the population, etc.) to enable running appropriate simulations, and thus we leave it for future work.

Addressing the feedback of reviewer 3:

- *Practicality of local u.a.r. slot selection, and availability for largely overlapping fixed-size windows*: The design choices for our framework are motivated by real-world applications, such as a client device can actually locally determine and participate in a u.a.r. slot in its own availability window (since it is available to participate during each instant of its availability window). Similarly, federated learning involving a population of a country, for instance, can be expected to have largely overlapping windows due to diurnal variations in client device availability. A sliding-window type behavior can be expected for learning from a global population (due to shifting time-zones for various sub-populations).
- *Slot R_j known to the algorithm?*: The amplified DP guarantees are central DP guarantees, which assume a trusted server, and thus R_j is known to the algorithm (though it cannot be released publicly for the amplification, similar to 'secrecy of the sample' in amplification due to sampling, or the random ordering being secret in amplification due to shuffling).
- *Difficulty for non-overlapping slots*: The privacy guarantees of our protocols do not depend on any overlapping structure of the slots, but the utility of a protocol will have a dependence. Thus, we analyze some variants that were motivated by practical applications, such as limited overlap of clients that is motivated by diurnal variations in training on local/national populations, and sliding window availability additionally motivated by shifting time-zones for training on a global population.