Thanks to Ariel Gabizon and Zac Williamson for collaborating on the post, and the authors of Marlin for highlighting the attack and its importance.

The attack

Cheon shows that if you're given g, g^α and g^{α^d}, where g is an element of a group of order p and $d \mid p -1$, then it's possible to find α in $2\left(\left\lceil\sqrt{\frac{p - 1}{d}}\right\rceil + \left\lceil\sqrt{d}\right\rceil\right)\cdot \left(\mathsf{Exp}_{\mathbb{G}}(p) + \log{p} \cdot \mathsf{Comp}_{\mathbb{G}}\right)$ operations, where $\mathsf{Exp}_{\mathbb{G}}(n)$ means the cost of one exponentiation of an element in $\mathbb{G}$ by a positive integer less than n amd $\mathsf{Comp}_{\mathbb{G}}$ means the cost to determine if two elements of $\mathbb{G}$ are identical. By assuming that $\mathsf{Exp}_{\mathbb{G}}(p)$ dominates $\mathsf{Comp}_{\mathbb{G}}$ and that the $\log{p}$ factor can be ignored when using a hash table, the cost formula can be simplified to be approximately $2\left(\left\lceil\sqrt{\frac{p - 1}{d}}\right\rceil + \left\lceil\sqrt{d}\right\rceil\right)\cdot \mathsf{Exp}_{\mathbb{G}}(p)$. The storage cost is $\max\left({\left\lceil\sqrt{\frac{p-1}{d}}\right\rceil}, \left\lceil\sqrt{d}\right\rceil\right)$ elements of $\mathbb{G}$.

For more intuition on how the attack works, check out Ariel's write-up.

Cheon uses Baby-step Giant-step as the main part of the attack, and it's possible to use Pollard's Rho instead.

When using Pollard's Rho algorithm, we can either use a large memory or a constant memory version, as mentioned in [3]. For the large memory version, i.e. which requires saving around $1.25\left(\sqrt{\frac{p-1}{d}} + \sqrt{d}\right)$ elements of $\mathbb{G}$, the expected number of evaluations (which roughly mean exponentiations) is $1.25\left(\sqrt{\frac{p-1}{d}} + \sqrt{d}\right)$. For the constant memory version, the expected number of evaluations is $3.09\left(\sqrt{\frac{p-1}{d}} + \sqrt{d}\right)$ and $1.03\left(\sqrt{\frac{p-1}{d}} + \sqrt{d}\right)$ comparisons.

The Marlin authors also noticed that if you're given g, g^α and g^{α^d} and h, h^α and h^{α^d} where g is a generator of $\mathbb{G}_1$

and h

is a generator of $\mathbb{G}_2$

, it’s also possible to use the pairing to transfer the problem into $\mathbb{G}_T$

: $e(g^{\alpha^m}, h^{\alpha^n}) = e(g,h)^{\alpha^{m+n}}$

.

The impact

This is particularly relevant for trusted setups that have been performed in the past and are being performed at the moment. Solving for τ

allows for the possibilty of breaking soundness.

1. [Zcash Powers of Tau - Sapling - BLS12-381]() - we have up until $g^{\tau^{2^{22} - 1}}$

in $\mathbb{G}_1$

and $g^{\tau^{2^{21}}}$

in $\mathbb{G}_2$

1. [AZTEC PLONK setup - BN254]() - we have $g^{\tau^{3 \cdot 2^{25}}}$

in $\mathbb{G}_1$

1. [Perpetual Powers of Tau - BN254]() - we have up until $g^{\tau^{2^{29} - 1}}$

in $\mathbb{G}_1$

and $g^{\tau^{2^{28}}}$

in $\mathbb{G}_2$

1. [Filecoin Powers of Tau - BLS12-381]() - we have up until $g^{\tau^{2^{28} - 1}}$

in $\mathbb{G}_1$

and $g^{\tau^{2^{27}}}$

in $\mathbb{G}_2$

Let’s take the biggest one to show the potential impact - Perpetual Powers of Tau. By the Cheon method with Pollard’s Rho, we can solve DLP in $\mathbb{G}_1$

for τ

in $1.25\left(\sqrt{\frac{2^{254}}{2^{28}}} + \sqrt{2^{28}}\right) \approx 2^{114}$

, so at most 2^{114}

exponentiations, or 114-bit security. For BN254, the impact is not severe, since there are other NFS-based attacks that lower the security to around [110-bit security](). You could also transfer the method to $\mathbb{G}_T$

, and get $1.25\left(\sqrt{\frac{2^{254}}{2^{29}}} + \sqrt{2^{29}}\right) \approx 2^{114}$

, but the operations in $\mathbb{G}_T$

are significantly more expensive.

For BLS12-381 setups, the impact might be more meaningful. The goal was to design a curve with 128-bit security, and the trusted setup lowers is. In the Filecoin parameters, this translate to $1.25\left(\sqrt{\frac{2^{255}}{2^{27}}} + \sqrt{2^{27}}\right) \approx 2^{114}$

, so at most 2^{114}

exponentiations.

This is also relevant to other projects which will perform a trusted setup:

1. Projects that are using curves mentioned in Zexe, such as Celo and possibly EYBlockchain
2. Coda that uses MNT4753 and MNT6753
3. Projects that are using curves mentioned in DIZK

Conclusion

Future projects that target 128-bit security should also consider this attack, which has become relevant because of the growing size of circuits.

This might also be a benefit of updatable setups, such as can be done for PLONK, Marlin and Sonic - you can estimate the amount of time it would take to solve for \tau

and make sure the SRS is updated before that.

References

[1] Cheon, Jung Hee. “Discrete logarithm problems with auxiliary inputs.” Journal of Cryptology 23.3 (2010): 457-476.

[2] Kozaki, Shunji, Taketeru Kutsuma, and Kazuto Matsuo. “Remarks on Cheon’s algorithms for pairing-related problems.” International Conference on Pairing-Based Cryptography. Springer, Berlin, Heidelberg, 2007.

[3] Bai, Shi, and Richard P. Brent. “On the efficiency of Pollard’s rho method for discrete logarithms.” Proceedings of the fourteenth symposium on Computing: the Australasian theory-Volume 77. Australian Computer Society, Inc., 2008.

[4] Chiesa, Alessandro, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS. Cryptology ePrint Archive, Report 2019/1047, 2019.

[5] Gabizon, Ariel, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. Cryptology ePrint Archive, Report 2019/953, 2019.